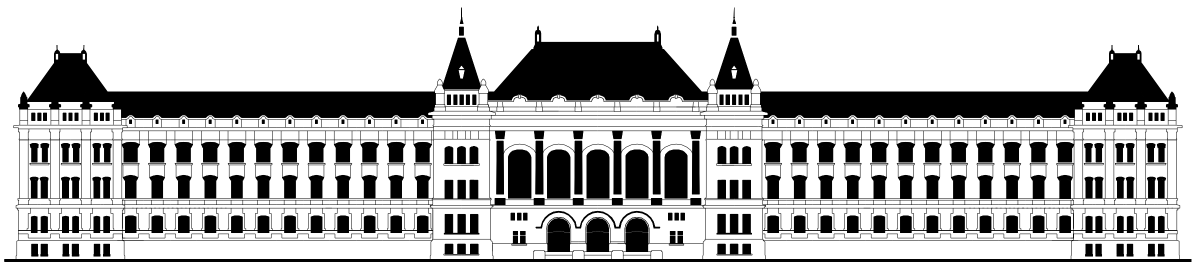
IEEE Hungary Section
Circuits, Systems and Computers Joint Chapter
Instrumentation and Measurement & Engineering in Medicine and Biology Joint Chapter

Proceedings of the 18th PhD Mini-Symposium

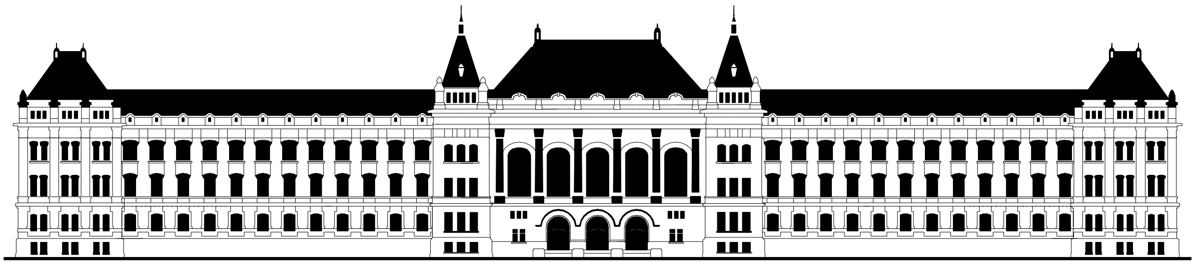
January 31–February 1, 2011.



Budapest University of Technology and Economics
Department of Measurement and Information Systems

PROCEEDINGS OF THE 18TH PHD MINI-SYMPOSIUM

JANUARY 31–FEBRUARY 1, 2011.
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:
Mihály BÁNYAI
István ENGEDY
Péter EREDICS
Ábel HEGEDÜS
Péter LACZKÓ
Gergely PALJAK

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

Sponsored by:
IEEE Hungary Section (technical sponsorship)
Schnell László Foundation
evopro Kft.
evosoft Hungary Kft.

ISBN 978-963-421-601-8

FOREWORD

This proceedings is a collection of the extended abstracts of the lectures of the 18th 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.

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. The lectures are at different levels: some of them present the very first results of a research, because most 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 eighteen-year period there have been shorter or longer cooperation between our department and some universities and research institutes. Some PhD research works gained a lot from these connections. In the last year the cooperation was especially fruitful with the Vrije Universiteit Brussel, Belgium; IRISA Rennes, France; IBM Budapest Center of Advanced Studies, Hungary; KFKI Research Institute for Particle and Nuclear Physics of the Hungarian Academy of Sciences, Budapest; Robert Bosch Kft., Hungary; National Instruments Hungary Software és Hardware Gyártó Kft., Debrecen; Innomed Zrt., Budapest, Nokia Siemens Networks Kft., Budapest, Hungary.

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 15, 2011.

Béla Pataki

Chairman of the PhD Mini-Symposium

EVOPRO INFORMATICS AND AUTOMATION LTD.

evopro Informatics and Automation Ltd. offers services in the field of industrial automation, engineering and manufacturing machines, drive technology, test automation, electrical installation and information technology all around the world.

Our professional quality provided by our engineer group is guaranteed by not only our references, but being certified as a Siemens Solution Partner – Automation, as a Microsoft Gold Certified Partner, as a Cisco Premier Certified Partner and as holder of the Drive Center Qualification. The company's processes are based on ISO 9001:2000 and 14001:2004 quality standards.

We have a wide range of services including engineering design and assembly of electrical systems, installment, training, supply of equipment, maintenance, support, consultancy, design and operation of info-communication networks, application development, unique and serial industrial machinery and building automation. It means that our activities cover everything from survey of demands up to 'turnkey' solutions.

To our benefit we perform testing projects in German and in Hungarian locations in cooperation with Siemens AG in order to help different system developments. These projects represent the newest versions and equipments of many Siemens Automation systems. The task of the Systemtest branch of evopro Ltd. is the final quality check of the product before marketing. During this procedure our engineers test the products through the aspects of the users. That is why it is extremely important that our professionals have wide experience in the fields of application and installation and they are aware of clients' requirements.

evopro Ltd. has considerable experiences and references in developing and constructing electrical systems, mechanical designs and software of specialized machines and full production line plants and facilities. Most of our projects require controlling, data and monitoring acquisition and application development based on industrial communication.
In the field of IT evopro Ltd. has experience in specific application developments, among our references we have sales and invoicing systems of high value. As a Cisco Certified and Microsoft Certified Gold partner we can guarantee the perfect network design, implementation and operation also in the field of individual application development.
Research activities of evopro focus on FPGA based hardware acceleration algorithms for biocomputing, development mobile applications on popular mobile platforms, developing measurement and embedded systems.
The Building Technology department is responsible for the design and construction of low voltage systems, such as fire alarm-, camera-, safety alarm- and access systems for office buildings, mansions, trade centres, industrial and agricultural facilities and public institutes.

15 years passed

and we have become the leading software development business

(according to the Budapest Business Journal)

We may be big, but our principles haven't changed!

Meanwhile we've got a new craze:

...and by now we are in a position to constantly offer job opportunities!

http://karrier. evosoft.hu/facebook

LIST OF PARTICIPANTS

Participant	Advisor	Starting Year of PhD Course
BÁNYAI, Mihály	STRAUSZ, György	2009
BERGMANN, Gábor	VARRÓ, Dániel	2008
CSURCSIA, Péter	KOLLÁR, István	2010
DEMIÁN, Tamás	PATARICZA, András	2010
ENGEDY, István	HORVÁTH, Gábor	2009
EREDICS, Péter	DOBROWIECKI, Tadeusz	2009
GALAMBOS, Róbert	SUJBERT, László	2010
GÉZSI, András	ANTAL, Péter	2009
HAJÓS, Gergely	ANTAL, Péter, DOBROWIECKI, Tadeusz	2008
HEGEDÜS, Ábel	VARRÓ, Dániel	2009
JUHÁSZ, Sándor	HORVÁTH, Gábor	2007
KOLLÁR, Zsolt	PÉCELI, Gábor	2008
LACZKÓ, Péter	FEHÉR, Béla	2009
MARX, Péter	ANTAL, Péter, DOBROWIECKI, Tadeusz	2010
OLÁH, János	MAJZIK, István	2009
ORBÁN, Gergely	HORVÁTH, Gábor	2009
PALJAK, Gergely	PATARICZA, András, KOVÁCSHÁZY, Tamás	2009
PÁLFI, Vilmos	KOLLÁR, István	2010
PECHAN, Imre	FEHÉR, Béla	2008
SÁRKÖZY, Péter	ANTAL, Péter, DOBROWIECKI, Tadeusz	2009
SCHERER, Balázs	HORVÁTH, Gábor	2007
SZATMÁRI, Zoltán	MAJZIK, István	2008
SZOMBATH, István	PATARICZA, András	2008
UJHELYI, Zoltán	VARRÓ, Dániel	2009
VÖRÖS, András	BARTHA, Tamás	2009

Program of The MINI-SYMPOSIUM

Conference Schedule

Time	January 31, 2011	Time	February 1, 2011
8:30	Conference Opening Opening Speech: Gábor Horváth Evopro presentation	8:30	Embedded and Intelligent Systems II.
8:45	Model-based Software Engineering I.	9:40	Measurement and Signal Processing II.
10:00	Embedded and Intelligent Systems I.	11:10	Embedded and Intelligent Systems III.
Lunch break			
13:00	Model-based Software Engineering II.		
14:30	Measurement and Signal Processing I.		

Sampling and Parameter Testing in Large IT Infrastructure Graphs

István SZOMBATH
Advisor: András PATARICZA

I. Introduction

Today, model-based techniques are rapidly appearing in various tools and approaches of IT system and service management. As a major element of this transition, more and more enterprise systems are being equipped with a central Configuration Management System that stores and maintains central configuration and structural models of physical and logical infrastructure components and their relationships. With the model of the IT infrastructure it is possible to validate the correctness of the infrastructure by performing analysis on the model itself. Models can be represented as typed graphs, thus methods from graph theory can be used to evaluate the IT infrastructure.

In the last decade the mathematical theory of networks i.e. graph theory has been one of the fastest developing areas of mathematics. Many new challenges emerged with the spread of large infrastructure networks (e.g. the Internet), that conventional graph theory methods cannot be dealt with. For instance, how to determine average neighborhood degree of computers in an IT infrastructure network, that is not know explicitly. Data needs to be collected by indirect methods like random local sampling. Mathematical background of large graphs can be found in [1].

A proof of correctness of technical solution for local sampling based estimation of global metrics is performed by means of large graphs theory. Sampling facilitates approximation of property and parameter, based on data from a small representative fragment, thus avoiding the necessity of global and complete measurements.

For example, it is possible to approximate the average neighborhood degree of computers and topological fault tolerance of a subsystem.

II. Graphs and Models and System Management

A. *Configuration Management Database (CMDB)*

A CMDB stores Configuration Items (CIs), their attributes and their mutual relationships with the intent to consolidate the configuration information stored in diverse tools into a single datasource with a harmonized data model. This way, relationships between CIs from different sources can be captured and tracked. For instance, a service level communication relationship can be explicitly mapped to network resources.

A properly maintained CMDB is essentially a runtime configuration model that can be readily utilized for the purposes of model-driven system management.

B. *IT Infrastructure Topology Model as a Graph*

The CMDB stores both the instance model and the metamodel of the infrastructure. Both models can be represented using conventional graph theory notations.

A metamodel describes the syntax of a modeling language. Formally, it can be represented with a type graph. Nodes of the graph are called classes. The classes may have attributes, properties of the specified class. Associations define relations between classes and are represented with an edge in the type graph.

An instance model describes a specific system defined in a modeling language as a well-formed instance of the metamodel. The instance model can be represented with a typed graph (typed over the metamodel). The nodes and edges are called objects and links; they are the instances of classes and associations, respectively.

For example, an instance model of an infrastructure can be represented as a graph, where nodes are the servers and edges are physical communication between the servers (we refer to this graph as the physical infrastructure graph). Another example is the transitive closure of the physical infrastructure graph (we refer to it as the logical infrastructure map). Note that if the physical infrastructure map is connected the logical infrastructure will be a full graph (clique). It is also possible to set edge weights in this graph to represent maximum bandwidth between nodes.

Typical IT Infrastructures are very large and complex, thus the typed graph of the physical and logical IT infrastructure model is huge. In this way many conventional graph algorithms fail due to the complexity.

C. Typical IT Infrastructure Patterns

The most important System Management task is discovery. Every other system management process uses the information on the IT infrastructure (e.g. topological dependency) populated into the CMDB by the discovery process. My previous work on discovery methods and CMDB population techniques can be found in [3].

In my I other previous work [2] I used flow based discovery methods (i.e. IP layer communication) to build and maintain the model of the physical and logical computer network in real time. The presented framework identifies typical IT infrastructure patterns automatically and tracks the matching set of these patterns also in real time. Thus for example, the framework can be used to track the matching of typical infrastructure pattern instances in the model, like 3-tier architectures. Furthermore the framework can export and import the model from and into any other CMDB solution (e.g. IBM TADDM), even in real time and incrementally, and any other discovery method can be implemented (as a plug-in).

III. Fundamentals of Large Graph Theory

Mathematical background of large graphs can be found in [1]. Large graphs are either graphs that are not known explicitly or for a practical reason we consider the number of nodes infinite. Two types of large graphs are distinguished:

- Dense Graphs are graphs where the number of edges are $O(n^2)$ where n is the (estimated) number of nodes,
- Bounded Degree Graphs or sparse graphs are graphs where every node has bounded number of neighbors (if d is the maximum degree of a node in, the graph the number of edges are at most nd).

Sparse graphs show similarities with the Internet and most network build upon. Thus for the practical point of view the theory of bounded degree graphs is more interesting. However theoretical results of dense graphs are more complete.

Algorithms of very large graphs do not use the graph in an explicit from, data is collected indirectly, for instance, by local sampling. To obtain a parameter of a large graph, a smaller sampled subgraph is studied.

A. Sampling a Graph

In the dense case, random n node of a graph are selected, then the random induced subgraph is taken. It is called subgraph sampling.

In the sparse case, neighborhood sampling is used. Random node $v \in V(G)$ is taken, and explore its neighborhood to a given depth r (radius). This graph is called the r-ball ($B_G(v, r)$).

Note that if a global boundary exists for the degree of nodes, for a given m the number of possible m-balls is finite. Also note that for every rooted graph F with bounded degree d there is a probability that the neighborhood sample returns F. Let this distribution denoted by $\rho_{G,m}(F)$.

B. Graph Sequences and Convergence

To introduce infinite graphs first graph sequences needs to be discussed. A graph sequence is a series of graphs; the easiest practical interpretation is for instance an evolving graph where edges and nodes are added to the graph continuously.

The sequence of graphs (G_n) with degrees uniformly bounded by d and $|V(G_n)| \to \infty$ is convergent if the neighborhood densities $\rho_{G,m}(F)$ converge for all m and all finite rooted graphs F [1].

Informally this means an evolving graph is convergent if the densities for all patterns are convergent. In another interpretation a graph is convergent if it contains p_i pattern with similar probability. For instance an evolving IT infrastructure model contains webservices, mailing services, and storage services with uniform (constant) probability.

C. Parameter and Property Testing

Parameter testing in dense case is function and used to estimate some parameter of a very large graph G based on a sample graph $G[S]$. The expectation is that if the sample is sufficiently large enough the estimated value approximates the parameter with large probability. Formally function $f : G \to \Re$ is testable if for $\forall\epsilon > 0$ $\exists k \in N$ that if G is a graph with at least k nodes and we select a set X of k independent uniform random nodes of G, then from a subgraph $G[X]$ induced by them we can compute an estimate $f(G[X])$ of f such that $P(|f(G) - f(G[X])| > \epsilon) < \epsilon$. This equivalent with that for every convergent graph sequence G_n, $f(G_n)$ also converge.

For example, the average degree of a p2p network (or any dense network) can be estimated by taking and observing a random induced subgraph. Furthermore the sequence of calculated parameters (on a convergent graph sequence) also converges.

Property testing in sparse case is when a true or false property needs to be determined for a given (large) graph ($G \in P$?) using neighborhood sampling method. Formally graph property P is testable for graphs G_d globally bounded by d if for $\forall\epsilon > 0$ there are integers $r = r(d, \epsilon)$ and $k = k(d, \epsilon)$ such that sampling k neighborhoods of radius r from a graph G with degree bounded by d we can compute YES or NO so that:

1. if the answer is NO, then $G \notin P$;

2. if the answer is YES, then at most $\epsilon|V(G)|$ edges can be changed in G to get graph in P.

An important theorem is that if a graph property is preserved by edge and node deletion and disjoint union, then it is testable for graphs with bounded degrees.

A practical example for sparse graphs (e.g. logical infrastructure model) looks as follows: by taking random nodes and observing the local neighborhoods properties can be estimated, like every service are fault tolerant (hasNonFaultTolerantService).

IV. Sampling and Measuring the Infrastructure

In my previous work I presented a framework that builds and maintains the model of the IT infrastructure mainly by observing IP traffic information. Also a library of typical IT infrastructure patterns is created. The framework tracks the instances of patterns (the matches of the patterns in the library) in the infrastructure model in real time.

This framework is used to track the distribution of library patterns and examine that a graph sequence (evolving model) is convergent or not. Furthermore, if it is convergent properties and parameters are estimated based on the sample.

The approach presented in this paper looks as follows:

1. Discovery of the IT infrastructure using flow parsing techniques,
2. Sampling points are chosen randomly, r-neighborhoods (radius) of sample points are explored,
3. Matching of typical IT infrastructure patterns are identified in the model,
4. Graph convergence of pattern distribution densities is examined,
5. Aggregation: instances of pattern matches may aggregated into a single node. In this way a layered model can be created, where the more aggregated layers contain significantly less nodes,
6. Properties and parameters are calculated if the model is convergent.

It is assumed that majority of the graph can be covered with patterns from the library i.e. typical IT infrastructure patterns. To examine the graph convergence a vector is created. The n-th element of the vector is the number of matches of the n-th typical IT infrastructure pattern in the IT infrastructure model. The graph sequence is convergent if the vector is convergent.

For example, the $(0.5, 0.3, 0.15, \ldots)$ (normalized) vector indicates that 50% of the nodes are clients, with 30% chance we may find non-redundant (simple) webservice, with 15% load-balanced or fault tolerant webservice, and the rest of the infrastructure cannot be covered with patterns from the library. If this vector is convergent through time the graph sequence is convergent.

Note that if (limit of the) convergence changes it is a sign. It may indicate malicious activity or emerging trends.

If the graph sequence is convergent properties and parameters can be approximated from sample. For example:

- average neighborhood degree of computers
- topological fault tolerance of a network or subgraph

V. Future Work

It is proofed that model of the infrastructure can be sampled, to approximate certain parameters and properties of the model. This provides a validation of the framework developed and implemented earlier. Thus (a relevant) model can be built and maintained based on incomplete observations.

Also limitations of this approach is examined: if the graph sequence is not convergent or parameter or property is not testable this method is not valid. Although it is also a vital information that a graph sequence is not convergent anymore. It may be a sign of malicious activity for instance. Non testable properties and parameters are also carry significant information. To calculate such a parameter full observation of the model is needed, but in an enterprise size network it is almost impossible.

Our main goal in the near future is to create more sophisticated and relevant parameters and properties that can be approximated from a sample, both in dense and sparse case. For example such metrics may be the distribution of degrees, to identify important servers.

Significant step may be to transform sparse physical infrastructure topology to a dense graph in order to approximate more sophisticated metrics. For example, by taking its transitive closure. Also graphical representation of graphs used by mathematicians (graphon) can be used to represent the IT infrastructure (e.g. to identify highly correlated subgraphs).

References

[1] L. Lovasz, "Very large graphs," *ArXiv e-prints*, Feb. 2009.

[2] I. Szombath, "Online infrastructure dependency detection and tracking," in *Conference of PhD Students in Computer Science*, p. 67, Szeged, 06/2010 2010.

[3] I. Szombath, "Investigating indirect dependencies in bipartite cliques of it infrastructure topology," Research Report RZ 3714, IBM Research Zurich, 2008.

TEST DATA GENERATION USING METAHEURISTICS

János OLÁH
Advisor: István MAJZIK

I. Introduction

Software testing is the process of evaluating the quality of the software under test by controlled execution, usually with the primary aim to reveal inadequate behavior or performance problems. Obviously, testing is an essential step of all software development models, however, writing tests is expensive, labor-intensive and time consuming, thus contributors often skimp on testing phase.

One of the most important task in a testing is test data generation. In software testing, test data generation is the process of identifying input data which satisfy certain criterion, e.g. coverage of selected program code. Test data generation is roughly equivalent with the simple case of test case generation, where the goal is to find an input sequence, that will drive the execution along a particular path in the control flow graph. Nevertheless, a test case is usually more than a simple input data, e.g. test case usually contain the desired output of the system under test.

Unfortunately, it's usually difficult to create realistic production data, especially early in the development stage, when discovering software bugs would be important in order to avoid expensive redesigns in later phases. Enormous effort have been made to overcome these difficulties. Through the last decades, several approaches have appeared for automatic software test data generation. In paper [1] authors divided these methods into three classes: random, path-oriented and goal-oriented automatic test data generation methods.

In this paper we present a novel automatic test data generation approach, which utilizes the idea behind model based test generation. In MBT, test cases are generated using the engineering models and formal specifications of the system under test, constructed during the planning phase (left side of figure 1 presents the general architecture of MBT). The models provide specific information for the test derivation algorithm, while formal specifications are precise, machine-readable mathematical descriptions of the expected behavior of the software under test (see [2] for details about MBT).

Test derivation is a crucial component in this approach, since the effectiveness of MBT depends on the level of automation it provides. In fully automated test case generation, models are usually translated to finite state automaton or transition system, and these are searched for possible executable paths, by model-checking, constraint satisfaction or SAT solving. An extensive survey of automated test generation is presented in [3].

II. Test data generation

In the previous section, we mentioned the classification of test data generation methods. As a remainder, test data generation is often formulated as identifying a program input which executes a given program element. Random methods obviously select input data by random selection. The advantage of these methods is the easy implementation and their generality, however in case of large state space they are very unlikely to identify proper input sequence.

Path-oriented methods reduce the test data generation to a path problem, where a path in the control flow is selected usually to trigger a selected program statement, and then the task is to generate input data to execute that path. Two popular methods in this category is symbolic execution and execution-oriented test data generation. This approach is best suited for programs with relatively small number of paths and simple control sequences, since their main disadvantage is the wastage of significant computation effort while identifying infeasible paths in the control flow graph.

In the goal-oriented approach, the path selection is eliminated, thus the goal is to find particular input data which trigger a selected statement in the program code. Methods using this approach monitor the program execution with the current input data, and classify branches according to their influence on execution of the desired branch.

The weakness of these approaches is the program analysis stage. Both path- and goal-oriented approaches require the analysis of program code, which can be complicated in case of large program code, partly implemented program with component stubs or legacy code. Furthermore, all introduced approaches handle complex data structures with difficulty, though most modern software application deal with large and complex input data.

III. Metamodel-based test data generation

Besides engineering models describing the structure and dynamic behavior of the system under test, complex input data of the program is usually modeled by the architect. Our proposed approach for test data generation uses this metamodel, in order to generate feasible test data. The metamodel describes the abstract syntax of a modeling language, thus in our case it defines the well-formedness rules of program data, and test data generation task can be interpreted as instance model generation.

This idea is inspired by model-based test generation, but instead of engineering models we use the metamodel of test data. However, this approach only suitable when goal of testing can be defined without engineering models. The architecture of the approach is shown on the right side of figure 1.

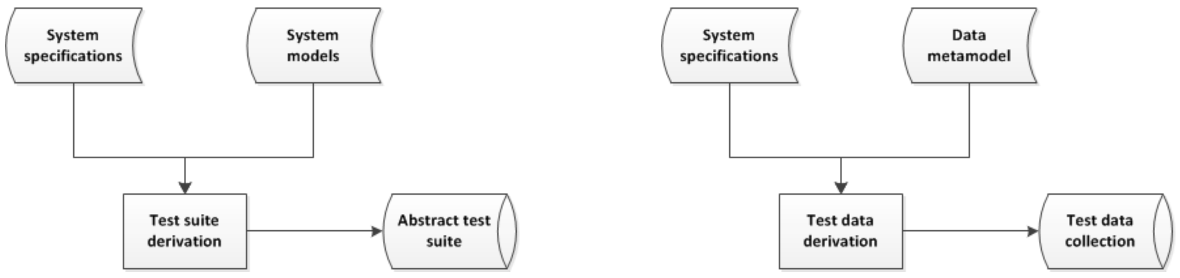


Figure 1: Left side of the figure presents traditional architecture of model-based software test generation, while the architecture of the proposed metamodel-based test data generation is shown on the right.

As it was already observed in section I., test derivation algorithm in MBT usually works on the transformed automatons of the original models. In this approach we propose the utilization of metaheuristics for test data generation.

A. *Metaheuristics*

Metaheuristics is the primary subfield of stochastic optimization applied for a very wide range of problems. However the term is somewhat misleading, optimization methods in this category apply some degree of randomness to find an optimal solution. Metaheuristics are advantageous in problems described as “I know when I see it”. In these problems there isn’t a proper algorithm to find the optimal solution, though we are able to score the quality of a selected solution and decide whether it’s optimal.

Metaheuristics exploit a heuristic belief about the space of candidate solutions. This means that similar solutions behave similarly, thus small changes in parameters will result in small changes in the quality of the current solution as well. Class of well-known basic metaheuristics contains single-state methods (i.e. hill-climbing, simulated annealing or tabu search) and population methods (i.e.

genetic algorithms and evolution strategy from the field of evolutionary computation, or particle swarm optimization from the class of swarm intelligence methods). Furthermore countless combinations and modifications are described in the literature. In the current paper we won't go into details of these methods, an exhaustive and up to date description of metaheuristics is presented in [4].

Crucial stage of applying metaheuristics is the procedure of evaluating the candidate solution, which is usually executed by an objective function. Since this determines the difference between candidate solutions, it must reflects the quality of a candidate according to the given problem. Thus objective function is always problem-dependent.

Another question when applying metaheuristics is how to update the candidate solutions. In traditional problems the candidate solutions are represented as vectors, thus updating is executed by random alteration of values in the vector. Obviously, this is again problem-dependent, furthermore implementation-dependent. The idea we applied in case of metamodel-based test data generation is introduced in the following section.

B. Metaheuristics for test data generation

According to the test data generation idea outlined above, our task is to generate feasible program input using metamodel of test data, thus test data generation task can be interpreted as instance model generation.

The quality of generated test data is measured by an objective function. In our approach, we utilize the information given in form of specifications to construct the objective function, hence score the quality of a candidate solution. Thus, a candidate solution is considered better, if it covers more specification (if multiple specifications are considered in the objective function).

The specifications address the behavior of system under test, and not the test data directly, therefore they do not always carry information regarding input data (typically non-functional specifications define behavior independent from the input data). Obviously we must select, and even process suitable specifications to construct the objective function.

Since our candidate solutions are represented by instance models, updating of a candidate can be executed by model transformation. Possible transformations of candidate solutions are defined by a set of model transformation rules. In every iteration of test data generation, an arbitrary number of rules are selected and executed. Obviously, these rules don't violate the well-formedness rules provided by the metamodel. Figure 2 presents the workflow of the proposed test data generation algorithm.

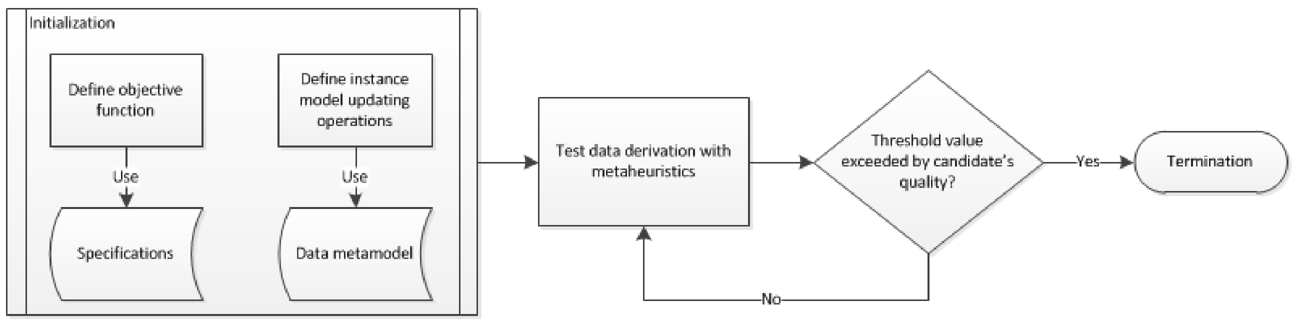

Initialization
Define objective function
Define instance model updating operations
Use
Use
Specifications
Data metamodel
Test data derivation with metaheuristics
Threshold value exceeded by candidate's quality?
Yes
No
Termination

Figure 2: Workflow of test data generation algorithm.

IV. A simple example

In this section we show an example application of the proposed metamodel-based test data generation approach. Consider a software under test responsible for controlling an autonomous robot. This software is clearly an agent program, since it perceives it's environment through sensors and acts upon that environment through effectors (definition of agent in [5]).

Let's suppose that our system under test has an architecture presented in figure 3. The agent program maintains a model of its environment called the context model, and the agents uses this model as input data in order to decide among possible actions.

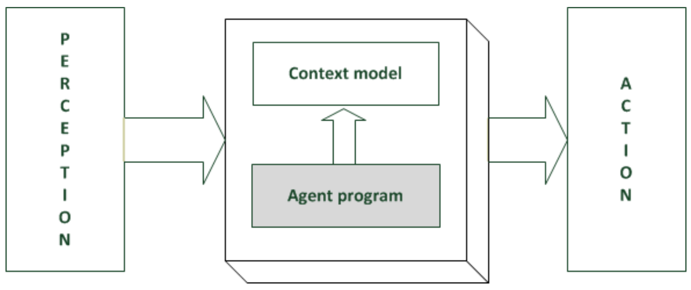


Figure 3: Architecture of the system under test.

Testing of the agent requires simulation of different environment settings, which can be performed by generation of context models, and injecting these test models into the architecture above in order to examine output of the agent program (feasible place of injection depends on the implementation).

Obviously this context model should have a metamodel, which describes the valid objects and relations within the environment of the agent. Our proposed test data generation algorithm could use this metamodel to generate complex test models.

Possible model transformations would include adding and erasing objects and relations from the metamodel, while an appropriate objective function could monitor, whether a necessary set of objects are present in the generated model (coverage criterion).

V. Conclusion and future work

In this paper we've introduced a novel metamodel-based test data generation method, which uses metaheuristic as test data derivation algorithm and constructs objective function from specifications.

In the outlined method, updating of candidate solutions represented by model instances is executed by model transformations. Unfortunately, constituting a set of transformation rules is labor intensive and time consuming. In the future, we plan to develop a method which automatically creates model transformation rules in run time, by partitioning the metamodel based on the information in specifications.

Beyond reducing the time needed to initialize a test data generation process, this solution would increase the universality of the approach, since this approach only requires formal specifications and formal metamodel to initiate test data generation.

References

[1] R. Ferguson and B. Korel, "The chaining approach for software test data generation," *ACM Trans. Softw. Eng. Methodol.*, 5:63–86, January 1996.

[2] D. Neto, A. C., R. Subramanyan, M. Vieira, and G. H. Travassos, "A survey on model-based testing approaches: a systematic review," in *Proceedings of the 1st ACM international workshop on Empirical assessment of software engineering languages and technologies: held in conjunction with the 22nd IEEE/ACM International Conference on Automated Software Engineering (ASE) 2007*, WEASELTech '07, pp. 31–36, New York, NY, USA, 2007. ACM.

[3] J. Rushby, "Automated Test Generation and Verified Software," in *Verified Software: Theories, Tools, Experiments - First IFIP TC 2/WG 2.3 Conference, VSTTE 2005, Zurich, Switzerland, October 10-13, 2005, Revised Selected Papers and Discussions*, vol. 4171 of *Lecture Notes in Computer Science*, pp. 161–172. Springer-Verlag, 2008.

[4] S. Luke, *Essentials of Metaheuristics*, 2009, Available at http://cs.gmu.edu/∼sean/book/metaheuristics.

[5] S. Russell and P. Norvig, *Artifical Intelligence. A Modern Approach.*, Pearson Education Inc., second edition, 2003.

Common Logic and its Dialects

Tamás DEMIÁN
Advisor: András PATARICZA

I. Introduction

Complex applications, like those in typical cyberphysical systems require information fusion of heterogeneous source domains, for instance of the physical, cyber and enterprise world. This fusion necessitates putting the modeling languages of the individual domains onto a uniform platform ensuring a seamless syntactic and semantic integration.
Common Logic (CL) is an ISO standard family of first-order logics since 2007 [1] which share a common abstract syntax and a precise model-theoretic semantic. Ontologies are the best means for concept integration from different fields. CL was made part of the OMG Ontology Definition Metamodel (ODM) [2] in order to offer ontologists with a means to represent constraints and rules with expressiveness exceeding in expressivity traditional Description Logics.

II. CL syntax

CL syntax (Fig. 1) has been deliberately kept small. This makes it easier to state a precise semantics and to place exact bounds on the expressiveness of subsets of the language.
A language is a CL *dialect* if there is a mapping between its syntax and the CL abstract syntax. This mapping provides the full model-theory of CL for that dialect. Dialects need not correspond exactly to CL abstract syntax, these are referred as *sub-dialects*. Dialects can also extend the CL syntax by the use of *irregular sentences*. The ODM [2] specifies the transformations from OWL, RDF(S) and OCL to CL. The CL standard defines three concrete model representation syntaxes: CLIF (CL Interchange Format, successor of KIF), XCL (XML based) and CGIF (Conceptual Graph based). Controlled English (CE), a human-readable formal language is also a dialect of CL [3] as well as SQL, Prolog, Datalog, RuleML and languages without formal semantics like UML diagrams and concept maps.

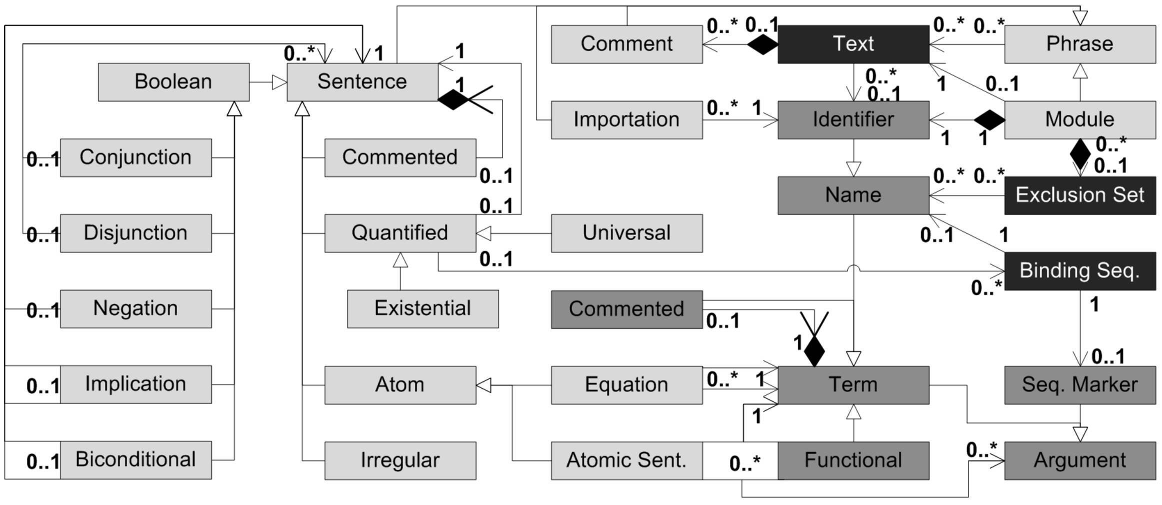


Figure 1: Metamodel of the Common Logic

Term denotes an entity of the domain (or universe). A term is either a name (string) or a functional term. **Identifier** is a name explicitly used to identify a module or piece of common logic text.

Text is a top-level component. A text is a collection of phrases. It may be identified by a name. A **Module** consists of a name, and a text called the body text. A module may optionally have an exclusion list of names whose denotations (i.e., list of variables) might be excluded from the local domain of discourse (for scoping purposes). **Importation** re-asserts the content of a text.

Phrase is a syntactic unit of text. It is either a comment, or a module, or a sentence, or an importation, or a phrase with an attached comment. Truth values are assigned to phases, not to sentences only.

Atom is either an equation with two terms or an atomic sentence (relation, predicate). Atomic sentences are similar in structure to functional terms. Arguments do not have any signature.

Sequence markers (S)(outside of the domain) can stand for arbitrary finite sequence of arguments. E.g. $\forall s \in S : r(s)$ is equivalent with an infinite (RE) conjunction: $r() \wedge r(x_1) \wedge r(x_1, x_2) \wedge \ldots$ ($\forall i \in \mathbb{N}^+ : x_i \in$ domain). CL with sequence markers is therefore not compact. However, if we think of such phrases to act as axiom schemata, then the logic remains first-order.

III. CL semantics

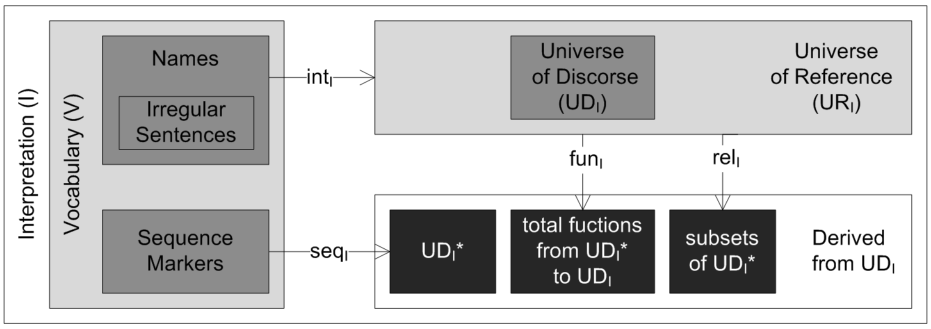


Figure 2: Semantic of the Common Logic

The domain (or universe) of discourse (UD_I) is the set over which quantifiers range (Fig. 2). The domain of reference (UR_I) is the UD_I set possibly with non-discourse names. The model-theoretic interpretation (I) of a vocabulary (V) (set of names and sequence markers) is a set UR_I with a distinguished non-empty subset UD_I and four mappings: rel_I, fun_I, int_I and seq_I for the formal interpretation of the relations, functions, names and sequence markers respectively.

We gain great expressive power by mapping extensions from entities in UD_I rather than from textual (syntactic) objects, e.g. quantification over relations, functions on relations, etc. Irregular sentences are treated as names and int_I includes a mapping to their truth values. A dialect with extra semantic conditions is a semantic extension. We can map the classes, relations and the functions of any formalism to rel_I, rel_I and to fun_I mappings respectively in a straightforward way.

Information fusion of different Domain Specific Languages is an union over their universes (then refreshing rel_I, fun_I, int_I and seq_I) resulting an unified interpretation. The fusion in a common CL framework facilitates a joint interoperation of different domain models. Typical task includes their co-validation and verification from general (completeness, consistency) and application specific aspects.

IV. Conclusion

This paper presents the essence of Common Logic. CL is able to serve as a model-theoretic background for any kind of knowledge representation especially on the Web. CL is an emerging technology still without a robust tool support and with unexploited potentials e.g. in model transformations.

References

[1] *CL standard: ISO/IEC 24707:2007(E)*, 2007, Available at standards.iso.org/ittf/PubliclyAvailableStandards/c039175_ISO_IEC_24707_2007(E).zip.

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

[3] J. F. Sowa, *Controlled English*, 2004, www.jfsowa.com/clce/specs.htm and www.jfsowa.com/logic/ace.htm.

ADAPTIVE BAYESIAN SEQUENTIAL STUDY DESIGN

Gergely HAJÓS
Advisors: Péter ANTAL, Tadeusz DOBROWIECKI

I. Introduction

The relative scarcity of the results of genetic association studies (GAS) prompted many research directions and hypotheses. To address the multiple testing problem computer intensive statistical methods became widespread and as an ultimate solution genome-wide association studies (GWAS) had appeared.

In the case of partial genome association study (PGAS) contrary to GWAS only partial information is available about the genome of the participants. In PGAS, we attempt to discover from the subsequent measurements of well-selected blocks of variables the relevant genetic factors for a given target set with interim analyzis and meta-analyzis of the available aggregated data sets in order to interpret and guide further measurements (see Fig. 1). The phases are shown in Fig. 1, starting with the GWAS layer and the application of gene prioritization systems for the subjective, knowledge-rich initiation of our pruning process.

One of the main bottlenecks in genetic association studies today are the required large sample size and complex models (with larger computational resources). In the paper we jointly address both issues within context of sequential PGAS: we apply a Bayesian, model-based meta-analysis in using an adaptive study design for pruning the variables. This can ensure large sample size within a fixed budget. We evaluate typical sequential pruning policies in association studies based on interim Bayesian meta-analysis. The approximation of one-step look ahead of expected value of experiments was also investigated with a full Bayesian approach. Our application domain is the investigation of genetic background of asthma using PGASs.

II. Background

The objective of the *sequential study design* (SSD) is to retrieve statistical information from data collected sequentially, given a utility and cost with a budget constraint for the data collection. A possible application is to check the effectiveness of drug treatment or to find association between genetic factors and diseases, etc. In the field of sequential study design generally the standard hypotheses testing approach is used [1]. The aim is to collect the minimum amount of data necessary to make a decision between null and alternative hypotheses, since one wants to minimize the costs of measurements performed in the study design. In every step of the sequential study design the tests of hypotheseses are performed on a set of samples, if one of the hypotheseses is accepted the study design is stopped and a decision is made. If a decision cannot be made due to lack of enough information the study design continues and more data is collected [2, 3].

Instead of applying the standard statistical approach we present in this paper a multivariate extension based on Bayesian networks. In every step of the study design we first approximate the utility of the computed results based on the available data, second we predict the future data based on the available data using Bayesian model averaging over Bayesian networks. With the help of future data we predict the utility of the continuation of the study design. If the predicted utility of continuation is higher than the utility based on the available data then the sequential study design is continued, otherwise stopped and the last computed results are reported.

Besides selecting the sample size to minimize the cost, in this paper we present an *active learning* approach to reduce the number of variables in every step of the sequential study design [4]. Since the algorithm in every step narrows down the set of the investigated variables, in the subsequent step just the last (i.e. the narrowest) set is used for further analysis. In this way in every subsequent step less and less measurements are performed.

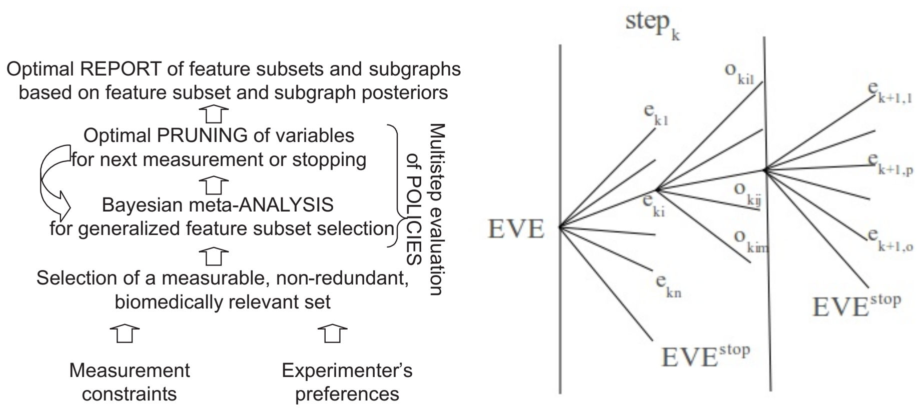

Optimal REPORT of feature subsets and subgraphs based on feature subset and subgraph posteriors
Optimal PRUNING of variables for next measurement or stopping
Bayesian meta-ANALYSIS for generalized feature subset selection
Multistep evaluation of POLICIES
Selection of a measurable, non-redundant, biomedically relevant set
Measurement constraints
Experimenter's preferences
step_k
EVE
e_k1
e_ki
e_kn
o_kil
o_kij
o_kim
e_k+1,1
e_k+1,p
e_k+1,o
EVE^stop
EVE^stop

Figure 1: Left: The phases of sequential study design. Right: Expected value of experiment.

III. Methods

A. *Calculation of expected value of experiment with multilevel analysis*

In each step of a SSD researchers set an experiment. We can say researchers in the step k choose experiment e_{1i} from set of possible experiments $(e_{k1}, e_{k2}, \ldots, e_{kn})$. Each experiment has an outcome o_{kij} with probability $p(o_{kij})$. The *expected value of experiment* (EVE) in step k is defined by $EVE(e_{ki}) = \max(EVE^{stop}(exp), \sum_i \sum_j p(o_{kij})EVE(e_{k+1,j}))$. Where EVE^{stop} is the utility of stopping the experiment with the actual results. The expected value of an experiment in step k depends on the expected value in step $k+1$. In this paper we want to maximize the EVE by selecting the optimal set of variables for experiment (see Fig. 1).

We assume that there is a special set of target variables, and the goal is the identification of an optimal set of relevant variables, and their interactions (for an overview of feature subset selection (FSS) problem, see e.g. [5]). The goal of the analogous Bayesian FSS can be defined as the computation of the posteriors for the pair wise relations, relevant sets, and interactions, which can be formalized as the posteriors for Markov Blanket Membership (MBM), Markov Blanket set (MBS), Markov Blanket subgraph (MBG) [6].

Respectively, we assume that utility function for the model can be additively decomposed into three parts, specifically for the MBM, MBS, and MBG levels. Note that given the utility function and the posterior over the model space in step i, the expected utility of reporting a structural model $\hat{M}$ is computable. The model with maximal utility can be determined as:

$$M^* = \arg\max_{\hat{M}} E_{p(M|D_{<i})}[U(\hat{M}|M)].$$

where M is a model with probability $p(M|D_{<i})$, and $U(\hat{M}|M)$ is the utility of $\hat{M}$ in case of M.

We evaluate typical policies in association studies based on interim Bayesian meta-analysis, and also the performance of a one-step look ahead approximation of the expected value of the experiments in the full Bayesian approach. Our application domain is the investigation of the genetic background of asthma using PGAS, where the costs of sample collection and genotyping are considerable, and a multivariate approach is essential due to complex, weak interactions behind multifactorial diseases.

B. *Bayesian sequential study design and variable pruning*

Since beside the selection of the number of samples, in case of variable pruning we also narrow down the number of the variables, we define the following: a prior $p(M)$ for the generative models; variable set S_i, where S_i contains only variables present in step i due to the reduction of variables; a corresponding likelihood $p(D_i|M)$ for the ith step and data set D where D_i represents the data set narrowed down to variable set S_i with the samples N_i collected in step i; a set of actions, continuing sequential study design consisting of both the selection of S_{i+1}, N_{i+1} or reporting actions (stop experiment and report the last computed model); a context-free, e.g. timeless, cost $C_{N_i}^{S_i}$ of measuring (observing) D_i. $D_{<i}$ represents the data set narrowed down to variable set S_i with all the samples collected in steps $< i$ ($N_{<i} = \sum_{j=0}^{i-1} N_j$).

In the optimal Bayesian approach, at step $1 < i$ one possibility is to stop and to report the optimal maximal utility model M^* with utility

$$U_i^{\text{report}} = E_{p(M|D_{<i})}[U(M^*|M)] - \sum_{j=1}^{i-1} C_{N_j}^{S_j}. \tag{1}$$

The other option is to continue by selecting the next, optimal experiment defined by the selection of N_i, with utility

$$U_i^{\text{cont}} = U(N_i) - \sum_{j=1}^{i-1} C_{N_j}^{S_j}, \tag{2}$$

where $U(N_i)$ denotes EVE. In case of a decision problem with finite horizon, backward induction can be applied to calculate Eq. 2. The exponential number of the potential future subsequent data makes however the estimation of these expectations computationally prohibitive (for evaluation of the value-of-information, see [7, 8]). The one-step approximation of $U(N_i)$ is as follows

$$U(N_i) \approx E_{p(D_i|D_{<i})}[E_{p(M|D_{\leq i})}[U(M^*|M)]], \tag{3}$$

which means that after the first step, the optimal Bayesian decision, (reporting M^* or continuing with measuring N_i) can be determined by comparing U_i^{report} to U_i^{cont} (see [9]). Note that the framework of Markov decision processes is not directly applicable to this context, because of the dynamic state space.

IV. Results

During the evaluation of the pruning algorithm (Section B.), results showed that the main bottleneck was the underlying Bayesian FSS algorithm (Section A.). We found that the Markov chain Monte Carlo (MCMC) based multilevel analysis (see [10]) does not converge to the real posterior distribution in the planned settings. The applied MCMC uses multiple chains with different heat temperature [11]. Our aim was to validate the MCMC algorithm and find proper temperature parametrization for stable simulation and fast convergence.

For testing convergence of the MCMC algorithm the multiple chain based Gelman-Rubin R score (see [12]) was used. For multiple parametrization five independent parallel simulations were performed.

The single chain Geweke's Z score (see [13]) was used to test convergence of confidence of the MCMC simulations. Z score was calculated for MBM and MBS features, our results show that over 5×10^6 step of burn-in, the calculated Geweke score of most of the features fall into the acceptable region (for illustration, see Fig. 2).

Regarding the confidence scores we face the multiple testing problem (MTP): the confidence of the simulation is estimated based on the confidence score of numerous (dependent or independent) features, thus we have to apply a correction, e.g. the Bonferroni correction. In the case of MBGs an extreme case of MTP occurs, because the typical number of MBG features is extremely high.

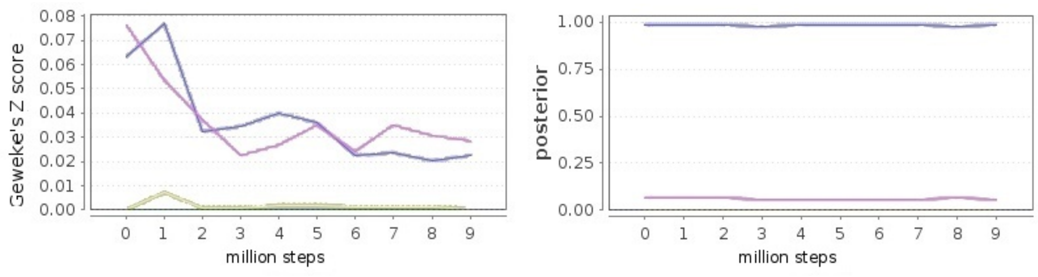


Figure 2: The horizontal axis: the simulation steps after burn-in. Left: Geweke's score of three features with the same parametrization. Right: The posteriors of the same features (one is constant zero).

V. CONCLUSION

In the paper we investigated the decision support for sequential partial genetic association studies, which are essential methods to ensure high sample size for more targeted statistical analysis after GWAS-based explorations. During the convergence of confidence investigations we determined a default parametrization for temperature settings and burn-in length. The MCMC algorithm was improved, the simulation is stable with the proper settings. Further research needed to find the utility function for one-step look ahead and calculate non-myopic look ahead.

References

[1] I. R. Konig and A. Ziegler, "Group sequential study designs in genetic-epidemiological case-control studies," *Hum. Hered.*, 56:63–72, 2003.

[2] D. J. Spiegelhalter, K. R. Abrams, and J. P. Myles, *Bayesian Approaches to Clinical Trials and Health-Care Evaluation*, John Wiley ans Sons, 2003.

[3] S. M. Berry, B. P. Carlin, J. J. Lee, and P. Muller, *Bayesian Adaptive Methods for Clinical Trials*, Chapman and Hall, 2010.

[4] J. Li, "Prioritize and select snps for association studies with multi-stage designs," *Journal of Computational Biology*, 15(3):241–257, 2008.

[5] Y. Saeys, I. Inza, and P. Larranaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, 23(19):2507–2517, 2007.

[6] P. Antal, A. Millinghoffer, G. Hullám, C. 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.

[7] H. D., E. Horvitz, and B. Middleton, "An approximate non-myopic computation for value of information," in *Proc. of the 7th Conf. on Uncertainty in Artificial Intelligence (UAI'91)*, pp. 101–107. Morgan Kaufmann, 1991.

[8] W. Liao and Q. Ji, "Efficient non-myopic value-of-information computation for influence diagrams," *International journal of approximate reasoning*, 49:436–450, 2008.

[9] J. M. Bernardo, *Bayesian Theory*, Wiley & Sons, Chichester, 1995.

[10] P. Giudici and R. Castelo, "Improving Markov Chain Monte Carlo model search for data mining," *Machine Learning*, 50:127–158, 2003.

[11] G. Altekar, S. Dwarkadas, J. P. Huelsenbeck, and F.Ronquist, "Parallel metropolis coupled markov chain monte carlo for bayesian phylogenetic inference," *Bioinformatics*, 20(3):407–415, 2004.

[12] A. Gelman and D. B. Rubin, "Inference from iterative simulation using multiple sequences," *Statistical Science*, 7(4):457–511, 1992.

[13] J. Geweke, "Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments," in *Bayesian Statistics 4*, J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, Eds., pp. –. Clarendon Press, Oxford, UK, 1992.

HAPLOTYPE- AND PATHWAY-BASED AGGREGATIONS FOR THE BAYESIAN ANALYSIS OF RARE VARIANTS

Peter SARKOZY
Advisor: Peter ANTAL

I. Abstract

The statistical analysis of rare variants from new generation sequencing methods has become a central challenge. We discuss the single gene aspect of "equivalent pathway degrading variants" with similar functional effects, which arises from the transcription to post-translation chain, and its multivariate pathway aspect arising from cascades and modules. We propose a stochastic aggregation for incorporating uncertain knowledge, and describe and evaluate a method for the Bayesian analysis of uncertain data using Bayesian networks.

II. Introduction

New generation sequencing methods are rapidly changing the landscape of the research of common diseases, tumorgenetics, and immunogenetics, some already advocate the era of the "common disease-rare variants" [6]. Thus the discovery and use of rare genomic variants with strong effects became a central challenge.

A haplotype is a combination of alleles or SNPs that are within close proximity on a chromosome, and are transmitted together. Since humans contain a set of maternal and paternal chromosomes, each haplotype is present on both, and the indication of which strand a haplotype is from is marked as its phase. Biological pathways are a number of linked biochemical steps, with a start and an end point; e.g. metabolic pathways or signaling pathways. A variation in the DNA (SNP, insertion, deletion, etc.) can be called a rare variant (RV) if it is present in less than 5% of the population.

At first, we present a unified approach to common and rare variants in common disease. Second, we catalog information sources for the univariate, gene-centered aggregation of variants, starting from transcription regulation to post-translational modifications. We also overview information sources for the multivariate, gene-gene/protein-protein associations and pathway based aggregation of variants. Next we demonstrate a method for the Bayesian analysis of uncertain data in the haplotype analysis of asthma.

III. Common and rare variants in common diseases

After decades of research of the rare variants of rare diseases, common variants (CVs) became central in the research of complex, common diseases. The slower-than-expected progress and partial results of the corresponding genome wide association studies (GWAS) resulted in heavy criticism, [6] concludes the failure of "the common disease (CD)-common variant (CV) hypothesis". Furthermore, it argues for a systems biology based evaluation of RVs in CDs.

The single nucleotide polymorphism (SNP) distribution in the human population is as follows: in a human population with 10^9 members, with 10^2 new germline SNPs per person "every point mutation compatible with life is likely present" while for a pair of genomes CVs give most of the variability. A CD is typically related to the combination of various degradations of multiple pathways, whereas the degradation of pathways is caused by both RVs and CVs. In fact, it is probably a tenable hypothesis that the effect strength of CVs and RVs are comparable, meaning that the current CVs are formed mainly by random drift, and proportionally there is no difference in coverage and functional role in pathway degradation. This implies that the majority of variants with strong effect are rare and we can discuss classes of "equivalent pathway degrading variants".

It is worth considering the multivariate extension of the earlier calculation in order to evaluate the asymptotic version of the CD-CV hypothesis; which states the asymptotic statistical sufficiency of CVs. Despite the presumed equivalence of CVs and RVs, taking into account the above mentioned aggregated minor allele frequency (MAF) for RVs, the chance of observing the causes (or proxies) of effected pathway patterns drops exponentially in a given patient when seeing only the CVs. To put it in other words, the loss of power of CVs for detecting CDs is a major problem escalating with their multifactorial nature, but this leaves the question of the asymptotic sufficiency of CVs for mapping CDs open.

IV. Aggregation of Rare variants

The weak or non-existing linkage of RVs limits the use of haplotypes or chromosomal regions for aggregations. However this property also limits the possibility of discovery of non-functional associations.

In the univariate, gene-centered approach RVs can be aggregated along the transcriptional regulations to post-translational modifications chain as follows: transcription factor binding sites, miRNA binding sites, splice-regulatory element binding sites, and phosphorylation and glycosylation related variations and conserved regions.

In the multivariate, pathway degrading approach RVs can be aggregated w.r.t. pathway knowledge bases and gene-gene/protein-protein associations.

In each case both nominal and quantitative variables V'_i can be induced based on the original sets of variables $\underline{V}$ using deterministic transformations $V'_i = f_i(\underline{V})$. Typically, there is considerable uncertainty over these transformations, and it is more practical to expect a Bayesian transformation, in which each deterministic transformation has a prior distribution $p(F_i = f_i)$. This implicitly defines a conditional distribution for stochastic mapping (assuming discrete F_i for simplicity)

$$p_i(V`_i|\underline{V}) = \sum p(f_i)f_i)(\underline{V}), \qquad (1)$$

V. Haplotype based aggregation

Haplotype block selection can be performed automatically with several freely available tools, as well as based on expert opinions. The PHASE [8] software can utilize certainty data generated in the previous steps (e.g. image processing, quality control) of the analysis, and output posteriors for the haplotype blocks. Aggregating multiple SNP's into haplotype blocks will result in increased cardinality per variable (since an SNP can only have 3 different states), with many low sample count rare haplotypes. We can handle this effect by implementing the following disease models for each block:

- Dominant disease model: Presence of a specific haplotype on either chromosome can increase risk. This is the general disease model used in most studies
- Recessive disease model: Both chromosomes must have a specific haplotype in order for it have an effect
- Merging of rare haplotypes

The method of maternal and paternal haplotype analysis using sample doubling can be described as follows. We separate the maternal and paternal chromosomes from each sample, because direct joining of the two would cause an undesirable cardinality increase. We also can sample the posterior haplotype distributions produced by PHASE. Next we create two samples from each data point, by splitting the data along the two chromosomes. Because the phase of the aggregations with relation to each other is unknown, we will generate multiple data files, each with a different, randomly generated global phase, and use the advantages of Bayesian model averaging to average out the effect of unknown global phases

VI. Pre-processing vs. post-processing aggregation

There are various numerically and statistically motivated transformation techniques in the data preprocessing phase, such as normalization, standardization, and dimensionality reduction. These methods can be very valuable in RV aggregation, although the discrete nature of the data excludes many standard solutions. However the real challenge is to incorporate prior knowledge in RV transformation.

The incorporation of such priors in data analysis is already common place, although not in the data preprocessing phase and not for detecting interactions, but in the post processing phase (see Table 1.), such as in the Gene Ontology (GO) annotation analysis or in the Gene Set Enrichment Analysis (GSEA) (for Bayesian aggregation in the post processing phase, see [2].

Aggregation types		Pre-processing	Post-processing
Chromosomal	Based on haplotypes or gene regions	With PHASE or direct joining	Sub Markov Blanket sets
Functional	Aggregating downstream of the functional pathway, or even across the entire pathway.	Along the transcriptional regulations to post-translational modifications chain	GO annotation, GSEA

Table 1: Methods of aggregation along different dimensions. Columns represent pre- and post-processing aggregation, while rows represent variable subset selection methods.

VII.Analyzing uncertain data

Because of uncertainty in RV transformation and aggregation, the analysis of uncertain data is a central theme in the analysis of rare variants. We will concentrate on the Bayesian statistical framework for the analysis, particularly because of its ability to incorporate priors and aggregate posteriors [2].Assuming a distribution over possible datasets, various approaches are as follows

1. using only the most probable data set,
2. using multiple data sets with high probability,
3. Monte Carlo data-averaging in Bayesian model-averaging (MC-DA-BMA).

The Bayesian averaging over model properties is done using Metropolis-Hastings algorithms (M-H) [2]. To avoid multiple burn-in in case of multiple data sets, we can mix data-averaging and model-averaging in a joint Metropolis-Hastings scheme, in the M-H-within-Gibbs, which is a hybrid of M-H and Gibbs sampling, the Gibbs sampling steps and the M-H steps can follow each other successively [3]. The Gibbs sampling steps can be used to generate uncertain and missing values, then using the completed data set a structure can be sampled in the M-H step.

VIII. Results

We performed transformations in a partial genetic association study in asthma, both in the gene-centered and in the pathway-centered approach. Here we illustrate our Bayesian network based methods for the Bayesian analysis of uncertain data for haplotypes in the gene-centered approach.

Because of computational reasons such separation of blocking, phasing (haplotype inference [1]) and data analysis is a practical choice followed in many systems (see e.g. HapScope [5], HAPLOT [4], GEVALT [7]). We similarly follow this decomposition, in which the biomedical expert specifies the blocks a priori (corresponding to unrelated chromosomal regions), then the PHASE method is applied for each block [8] to generate a maximum a posteriori distribution over phased genotyped data, and finally we apply our Bayesian model-based approach [2].

We applied the Monte Carlo data averaging in Bayesian model averaging (MC-DA-BMA) method in a genetic association study in asthma research investigating 56 SNPs, 15 genes in chromosome 11 and 14 (settings: burn--in: 1000000, step: 5000000, 10 random datasets from phasing distribution, see Table 2.).

Region Name	Average MBM posterior	Average MBM Posterior (ML)	Variance of MBM posterior
FRMD6 START HT1	0.638754	0.643055	0.0946
FRMD6 START HT2	0.723333	0.657325	0.0250
AHNAK HT 1	0.160561	0.152456	0.0187
AHNAK HT 2	0.611516	0.589275	0.0058

Table 2: BNF maximum likelihood and the averaged results on chromosome 11 and 14. Where HT is the haplotype.

IX. Conclusions

We focused on the Bayesian statistical framework for the analysis, particularly because of its ability to incorporate priors and to aggregate posteriors [2]. We summarized various types of aggregations of rare variants, proposed and implemented a stochastic aggregation scheme, which can be used both in the pre-processing and post-processing phases. We implemented various sampling schemes to cope with uncertain data sets using parallel computing information resources (for availability, see http://mitpc40.mit.bme.hu:8080). We demonstrated these methods for the Bayesian analysis of uncertain data in the haplotype analysis of asthma.

To cope with challenges of the analysis of RVs it is worth noting that beside the aggregation of the effects of RVs along pathways, their effects can be aggregated along multiple quantitative traits (QT) as well, for which new statistical methodologies are needed. QT analysis truly avoids some serious problems of case-control studies, such as binary oversimplification and population confounding. These requirements are neatly covered by a causal network analysis. Furthermore the Bayesian statistical framework offers many advantages for this analysis, such as the use of background knowledge as prior and in the posterior interpretation, meta-analysis, and automated correction for multiple testing. The application of RVs from NGS using a (Bayesian, causal) network analysis may define a third phase in genetic research after linkage analysis and association analysis.

References

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

[2] 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.

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

[4] Kidd K. K. Gu S, Pakstis AJ. Haplot: a graphical comparison of haplotype blocks, tag snp sets and snp variation for multiple populations. Bioinformatics, 21(20):3938-3939, 2005.

[5] William L. Rowe, Jinghui Zhang. Hapscope: a software system for automated and visual analysis of functionally annotated haplotypes. Nucleic Acids Research, 30(23):5213-5221, 2002.

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

[7] Ron Shamir Ofir Davidovich, Gad Kimmel. Gevalt: An integrated software tool for genotype analysis. BMC Bioinformatics, 8(36):2105-2112, 2007.

[8] 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.

Multivariate Bayesian Analysis of Genome-Wide Association Studies

András GÉZSI
Advisors: Péter ANTAL, Csaba SZALAI

I. Introduction

The completion of the Human Genome Program and the HapMap Project has made it possible to identify an informative set of > 1 million single nucleotide polymorphisms (SNP) across the genome that can be used to carry out genome-wide association studies (GWAS). These experiments performed to date exhilarated numerous new, replicable associations between common genetic variants and different phenotypic features (eg. complex diseases) [1].

Despite the promising results it is widely accepted that the effect of variations in the human genome is largely unknown [2]. The ”weak” results can be explained by several factors: the design of experiments do not exploit the available biological background knowledge; testing one SNP at a time does not fully realize the potential of GWA studies to identify multiple causal variants, which is plausible scenario for many complex diseases; the experiments usually ignore the environmental effects on diseases. To overcome these problems the using of multivariate methods is beginning to spread, which try to discover the probabilistic and causal relations of the joint effects of weak factors while modeling gene-gene and gene-environmental interactions.

The relative scarcity of the results of genetic association studies prompted many research directions, such as the use of the Bayesian framework [3]; and the use of complex models, such as Bayesian networks (BN), which can learn non-transitive, multivariate, non-linear relations between target and explanatory variables, treat multiple targets and allow scalable multivariate analysis [4]. Because their applicability in genome-wide association studies is hindered by the high computational complexity (approx. 10^6 variables and 10^4 number of samples), we implemented a two-phased combined method, in which a Bayesian univariate method filters the variables for the subsequent deep analysis for interactions.

We present results about the performance of this two-step approach using a realistic GWAS dataset.

II. Methods

A. *Generating the artificial dataset*

We demonstrate the results on an artificial data set generated as follows:

(1) We simulated a case-control dataset containing 10000 random samples with HAPGEN [5]. This program can handle markers in linkage disequilibrium and can simulate datasets over large regions, such as whole chromosomes. We used the publicly available files that contain the haplotypes estimated as part of the HapMap project, and the estimated fine-scale recombination map derived from that data (HapMap rel#22 - NCBI Build 36 (dbSNP b126)). The generated dataset contained 33815 SNPs, these SNPs defined the genetic background of our samples.

(2) We created manually a Bayesian network which modeled a realistic disease. The model contained 10 variables that were relevant with respect to the case-control target variable. The parametrization of the Bayesian network was set to be similar to typical complex diseases found in GWA studies [1, 6] (odds ratio (OR) between 1.1 and 1.5, minor allele frequencies above 0.01). The model contained variables that had no direct effect on the target variable, only through an other variable (pure interactions). Using this model, we generated 10000 random samples (5000 cases and 5000 controls) by

Gibbs-sampling.

(3) We integrated the two datasets as follows: we paired each sample from the (2) dataset with a randomly chosen sample from the (1) dataset. We concatenated the two rows to create a new row which hence contained $10 + 33815 = 33825$ variables. We used the resulting dataset to test our methods. Therefore the variables relevant to the target variable were known before the completion of the analysis, so we could easily measure the performance of our methods. The aim of our investigation was to identify all the relevant variables with respect to the target variable.

B. *Univariate Bayesian test used for filtering the variables*

To filter the variables we implemented a univariate Bayesian test [3] which was based on calculating Bayes Factors (BF). Let θ_{het} denote the logarithm (base e) of the OR between the heterozygote and the common homozygote. Let θ_{hom} denote the logarithm of the OR between rare and common homozygotes. The null hypothesis is:

$$H_0 : \theta_{het} = \theta_{hom} = 0$$

The general alternative, H_1, is that at least one of θ_{het} and θ_{hom} is non-zero. If we consider a precise alternative, for example:

$$H_1 : \theta_{het} = t_1, \theta_{hom} = t_2$$

in which t_1 and t_2 are known, then:

$$BF = \frac{P(data|\theta_{het} = t_1, \theta_{hom} = t_2)}{P(data|\theta_{het} = 0, \theta_{hom} = 0)} \tag{1}$$

However, in practice the values of θ_{het} and θ_{hom} under H_1 are unknown, and computing the numerator of the BF requires averaging over possible values for t_1 and t_2, which are weighted by their plausibilities before the association data were observed. These weights are referred to as the prior distribution for θ_{het} and θ_{hom} under H_1. Specifying a prior distribution for θ_{het} and θ_{hom} under H_1 proceeds by first selecting one or more genetic models. These genetic models mean implicit restrictions on the possible values of θ_{het} and θ_{hom}. In case of an additive model, $\theta = \theta_{hom} = 2 * \theta_{het}$; in case of a dominant model $\theta = \theta_{het} = \theta_{hom}$; and in case of a recessive model $\theta_{het} = 0$ and $\theta = \theta_{hom}$, so in all cases there is the only a single effect size parameter (θ).

C. *Filtering the variables*

We calculated the Bayes Factor between the case-control status and every variable from the 33825 variables of the final dataset. We sorted the variables by the associated Bayes Factors and chose the best 10, 20, 50, 100, 200 and 300 variables with the highest BFs. We performed the multivariate Bayesian analysis on these reduced datasets.

D. *Bayesian Multi-level Analysis*

The multivariate, multi-level Bayesian analysis was performed as it was previously described in [4].

We modeled the joint probability distribution of the variables of the reduced datasets with Bayesian networks. In Bayesian probabilistic networks the relevant variables with respect to the target variable form the Markov boundary of the target variable. Therefore our main task is a feature selection problem; the calculation of the posterior probability of the Markov boundary structural feature averaged over the possible network structures:

$$P(f|data) = \sum_G f(G)P(G|data) \tag{2}$$

where $f(G)$ is 1 if the feature holds in network structure G and 0 otherwise. The number of BN structures is super-exponential in the number of random variables in the domain; therefore this summation have to be approximated by considering only a subset of possible structures. We used Markov Chain Monte Carlo methods; we defined a Markov chain over structures whose stationary distribution was the posterior $P(G|data)$, we then generated samples from this chain, and used them to estimate (2).

III. Results and discussion

In the first step, the artificial GWAS dataset, that contained 33825 variables before filtering, was filtered with the univariate Bayesian test described at II. B. resulting 6 different datasets. The datasets contained 10, 20, 50, 100, 200 and 300 variables and 10000 samples each. Next, we performed the multivariate, multi-level Bayesian analysis described at II. D. on the first 500, 1000, 5000 and 10000 samples of the various datasets. We calculated for every variable the posterior probability of the structural feature, that the variable is in the Markov boundary of the target variable.

Figure 1 shows the posterior probability of Markov boundary membership of the truly relevant variables in different model sizes (variable counts) depending on the sample sizes. Increasing the sample size, the posterior probability of the truly relevant SNPs would also increase; so it can be generally said, that the more samples we had, the more probable that a truly relevant variable would be claimed by our methods as relevant. As it can be clearly seen in Figure 1, there were relevant variables (whose relevance was known before as we had set them to be relevant while generating the datasets) whose posterior probability were zero, even if we used 10000 samples. The reason of that is quite clear: variables that were in pure interaction with the target variable could have been excluded in the filtering step.

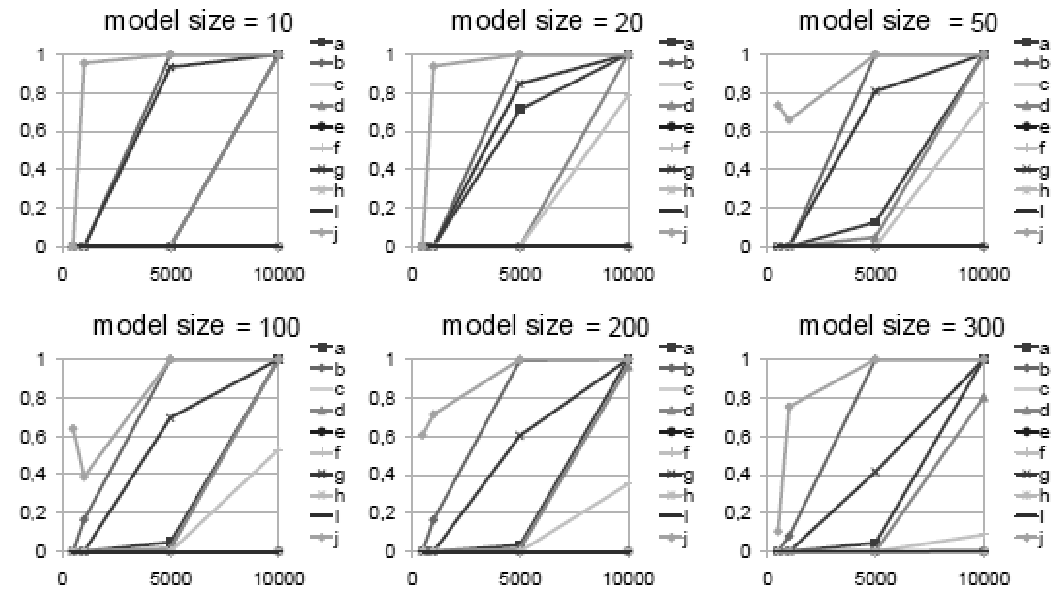


Figure 1: The posterior probability of Markov boundary membership of the truly relevant variables in different model sizes depending on the sample sizes. x-axis: sample size; y-axis: posterior probability; the count of variables in the datasets (the size of the models) from left to right, top to bottom: 10, 20, 50, 100, 200, 300.

Figure 2 shows the aggregated sensitivity and positive predictive values depending on the data sizes. Sensitivity values were calculated as follows: what proportion of the truly relevant variables' posterior probability exceed threshold 0.5. Positive predictive values were calculated as follows: what proportion of the variables claimed by the method to be relevant (posterior probability > 0.5), were truly relevant.

It can be clearly seen, that increasing the sample size, sensitivity and positive predictive values are increasing as well. Using 10000 samples, the method claimed at least 5 from the 10 truly relevant variables to be relevant in each model sizes. Using 500 samples the method found at most 1 variable; the biggest difference was at 5000 samples, where the best parametrization (model size = 20) yielded 4, the worst parametrization (model size = 300) yielded only 2 truly relevant variables.

The positive predictive values are generally increasing till 5000 samples. Using 10000 samples the smaller model sizes (10, 20 and 50 variables) yielded further increase, the bigger model sizes (100, 200 and 300) yielded minor decrease. Using 10000 samples the smaller the model size, the higher the positive predictive values are.

We also calculated the standard χ^2 test of association as a reference method (generally used for association testing). On both sides of Figure 2 a dashed line shows the results of this method on 10^{-5} significance level.

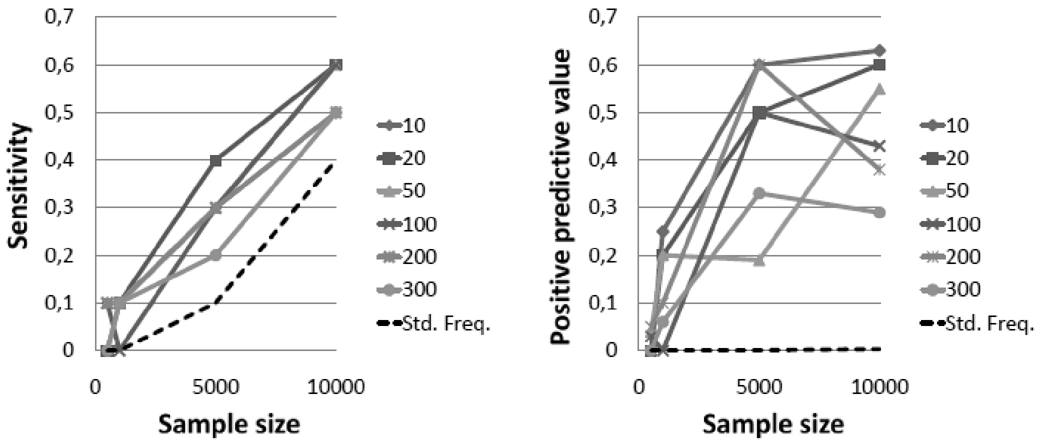


Figure 2: The sensitivity and positive predictive value of the method presented in the article. Left: Sensitivity: what proportion of the truly relevant variables' posterior probability exceed threshold 0.5. Right: Positive predictive values: what proportion of the variables claimed by the method to be relevant (posterior probability > 0.5), were truly relevant. On both sides a dashed line shows the results of the standard χ^2 test on 10^{-5} significance level.

IV. Conclusion

Probabilistic models are already widely applied tools in expression data analysis, in pedigree analysis, in linkage and association analysis. In the paper we presented the learning characteristics of a two-step Bayesian method in genetic association studies for systematically varying sample size and number of variables.

References

[1] W. T. C. C. Consortium., "Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls.," *Nature*, 447(7145):661–78, 2007.

[2] M. T. et al, "Finding the missing heritability of complex diseases," *Nature*, 461(7265):747–53, 2009.

[3] M. Stephens and D. Balding, "Bayesian statistical methods for genetic association studies," *Nature Review Genetics*, 10(10):681–690, 2009.

[4] P. Antal, A. Millinghoffer, G. Hullám, C. 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.

[5] C. C., A. Spencer, Z. Su, P. Donnelly, and J. Marchini, "Designing genome-wide association studies: Sample size, power, imputation, and the choice of genotyping chip," *PLoS Genetics*, 5(5), 2009.

[6] C. S. Ku, E. Y. Loy, Y. Pawitan, and K. S. Chia, "The pursuit of genome-wide association studies: where are we now?," *Journal of Human Genetics*, 55:195–206, 2010.

BAYESIAN APPROACH OF THE FEATURE SELECTION PROBLEM IN CASE OF CONTINUOUS TARGET VARIABLES

Peter MARX
Advisors: Peter ANTAL, Tadeusz DOBROWIECKI

I. Introduction

The analysis of the relevance of variables in high-dimensional regression problems is an important and difficult issue. In several problems, we have got man variables but scarce dataset. Because of the relative scarcity of the data there is no dominant model in the frequentist framework. Similarly in the Bayesian framework using an overall model including all the variables and their interactions results in a flat a posteriori distribution over the model parameters, additionally the Monte Carlo simulations are typically not tractable and it is hard to interpret the results. A more promising approach is to consider submodels with subsets of the predictor variables and to estimate the a posteriori probability of the inclusion of a predictor and its corresponding strength parameters. In this paper, I will give a short review of Bayesian variable selection with regression models. In the first part, I will present some important properties, which leads to different algorithms. In the second part, I will briefly describe the Stochastic Search Variable Selection (SSVS) algorithm.

II. Important properties

There are a few algorithms which use Bayesian tools to build a regression model. The main difference between them is how they define the prior distribution and the weights in the posterior model. In this section, I will describe some important properties in variable selection, following O'Hara and Sillanpää [1].

The sparseness is one of the most important properties of a model. The model complexity is related to the sparseness, but it cannot be said, that a sparser model is better in all cases. In a regression model, the easiest way to achieve the sparseness is to give a prior probability to every variable. This probability defines the chance of inclusion of a variable in the posterior model.

Bayesian variable selection method is based on assuming a normal prior distribution on the regression parameters. The variance of the distribution usually is a constant, but we can extend the model by estimating the variance as in case of SSVS. If we estimate the variance of normal prior, it helps tuning the parameters, because in the regression model the coefficient depends on the variance. In a heterogeneous problem, the variance can be set differently for all regression variables, which allows heterogeneity, furthermore local characteristics are included in the model.

Other important property of these models is how many parameters have to be set, because it can exponentially increase the computational demands.

III. Stochastic Search Variable Selection

In this paper, I give a brief description of stochastic search variable selection. SSVS was developed by George and McCulloch [2] for feature subset selection over regression models, considering a canonical regression setup [2][3]:

$$\mathbf{Y} = \beta_0 + \boldsymbol{\beta}\mathbf{X} \qquad (1)$$

The algorithm helps to put the full regression problem in a Bayesian statistical framework. Let us introduce an indicator variable $\boldsymbol{\gamma}$ **:** if $\gamma_j = 1$ then x_j is included in the regression model, if $\gamma_j = 0$ then

x_j is not included in the regression model. The SSVS algorithm defines the prior of regression coefficients β_j as follows: [2]

$$P(\beta_j \mid \gamma_j) \sim (1-\gamma_j) N(0, \tau_j^2) + \gamma_j N(0, c_j^2 \tau_j^2) \tag{2}$$

$$p_j = P(\gamma_j = 1) = 1 - P(\gamma_j = 0) \tag{3}$$

In Eq. (3) p_j defines the probability of inclusion for xj in the model. In Eq. (2) it can be seen easily that the coefficients are computed from different normal distributions.

The illustration of the sampling process is as follows: if a variable x_j is not included in the model then the coefficient of this variable comes from a normal distribution with 0 mean and τ_j^2 variance, if x_j is included in the model, the coefficient of the variable is defined by the second distribution on the right side in Eq. (2).

We have to choose τ_j^2 positive, but small enough to estimate the coefficient 0 in Eq. (2) The user has to set c_j^2 parameter large enough, so that the coefficient can be estimated as a nonzero value. With this prior configuration if $\gamma_j = 1$ β_j comes from a normal distribution with high variance, so x_j will be in the posterior model with higher probability, but β_j is small enough x_j also can be excluded from the model. If $\gamma_j = 0$ then β_j is small enough so x_j will be in the posterior model with low probability, but there is also a chance of inclusion for x_j.

These two parameters have to be set for each variable by the user, but several automated and semi-automated methods are known to define these parameters precisely, or George and McCulloch [2] showed a manual parameter setting which gives a really good approximation. These methods will not be considered in this paper.

The most significant gain of stochastic search variable selection is that the algorithm doesn't compute exhaustively the posterior probability for all 2^p subsets. It uses Gibbs sampling - which contains a search in variable space – so in the sample, the variables with higher probability will appear with higher frequency. This makes the SSVS very efficient in case of huge datasets [2]. The user specified parameters help to tune the algorithm and make it more problem-specific.

IV. Conclusion and summary

The variable selection problem is not solved, however, the Bayesian approach presented some good results. Bayesian methods are more and more popular because they coherently quantify the remaining uncertainty dictated by the data. In this paper, I briefly described an algorithm which can be applied in various cases, also with huge amount of data. I am implementing this algorithm in BUGS [4] (Bayesian inference Using Gibbs Sampling), and R [5], because BUGS is a good environment for Bayesian statistics. This implementation will handle the quantitative trait analysis in the GenaGrid [6] project complementing the current methods targeting discrete outcome variables.

References

[1] R. B. O'Hara, M. J. Sillanpää, "A Review of Bayesian Variable Selection Methods: What, Which and How," *Bayesian Analysis*, 4(1):85-118, 2009.

[2] E. I. George, R. E. McCulloch, "Variable Selection Via Gibbs Sampling," *Journal of the American Statistical Association*, 88(423):881-889, 1993.

[3] M-H. Chen, Q-M. Shao, H. G. Ibrahim, *Monte Carlo Methods in Bayesian Computation,* Springer, 2001.

[4] Ioannis Ntzoufras, *Bayesian Modeling Using WinBUGS*, Wiley, 2009.

[5] Ross Ihaka, Robert Gentleman, "R: A Language for Data Analysis and Graphics," Journal of Computational and Graphical Statistics, 5(3):299-314, 1996

[6] The GenaGrid Consortium, [Online] Available: www.genagrid.com

SUPPORTING THE TESTING OF AUTONOMOUS SYSTEMS USING CONTEXT ONTOLOGIES

Zoltán SZATMÁRI
Advisor: István MAJZIK

I. Introduction

This paper is based on my previous research on the conceptual comparison of the classical modeling technologies (e.g. UML, EMF) and ontology based modeling. Both of these modeling approaches are intended for describing domain-specific knowledge using high-level models, called metamodels and constructing domain models based on these metamodels [1].

Ontologies expressed in description logic formalism [2] provide a way to represent knowledge bases in a well-structured and expressive way. Ontologies consist of terminologies (TBox) and model instances (ABox). TBox represents the metamodel, it describes the concepts, its relationships and properties. ABox collects the model elements that are TBox-compliant instances. In other words terminology is a metamodel-like "dictionary" to describe a model while model instances store the knowledge of an object.

Last year I presented a framework that supports ontology based assessment of development toolchains. In this paper I propose a method that is based on ontology models and ontology metrics, that can be used for model based test generation for testing autonomous systems.

II. Testing of agent based systems

The concept of agents is a very popular one, which has its origin in the mathematical model of concurrent computation, called *actor model* [3]. After the invention of software agents, they spread across software industry as they have proven useful in many fields from email clients to autonomous robots. In textbook [4], agent is defined as *anything, that can be viewed as* ***perceiving its context*** *through sensors and* ***acting upon that context*** *through effectors.* The agent can be described with an **agent function** in abstract mathematical form, and the implementation of the function called **agent program**. The goal of this work is supporting ontology based testing of the agent programs.

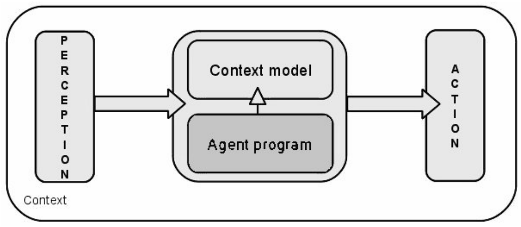


Figure 1: Architecture

In the rest of this paper I will concentrate on those agents, that can be installed in some kind of *spatial situation*, where they can move and execute different actions. Furthermore, I won't deal with the testing of perception and action execution, but only with the testing of the agent program which implements the agent function.

The reference architecture is shown in figure 1. The agent program is the **system under test**. This set up is very similar to the arrangement that authors use in [4], when defining the connection between an agent and its context.

A. Motivation

The main challenge in test generation is to avoid ad-hoc testing and support test generation based on measurable coverage metrics and well-defined method.

During the behaviour testing the focus is on testing the implementation of the the agent program.

The input is provided by the **perception module**, that provides the current situation and the changes of the context. Based on the internal rules and goals the **agent program** makes a decision and generates the input for the **actuator**.

The control of such an agent based system includes reasoning, learning and adaptation to evolving context. The testing of this function is supported by the **context model**, that stores the knowledge about the context for the agent itself. This model should describe all the things, events in the agent's context that are relevant for the control algorithms.

The context model can include **context patterns**. These patterns represent different aspects of test goals, focusing on complex combinations of things and events. Test goals can include complementary or in some cases conflicting patterns. To support test generation based on these patterns definition of **coverage metrics** and based on these metrics an **intelligent algorithm** is needed.

In order to support these approach I propose an ontology based description of the domain. A **context ontology** will be presented that supports the description of the context elements, and based on this ontology the context patterns and coverage metrics can be defined.

B. Context ontology

The **context ontology** is a domain-specific description of the objects and events in the agent's context. A **context model** is constructed based on this ontology and describes the following aspects of the context:

- The **static objects** that can be found in the context will be modeled using an ontology concept hierarchy, or in other words as a dictionary based taxonomy. The hierarchy can be used for defining coverage metrics of a context model in case of testing.
- Every object could have some **properties** (e.g., location), that are modeled using Data properties or Object properties, that are base elements in an ontology TBox model.
- The **relations between concepts** will be also utilized when test coverage is analyzed.
- **Dynamic changes**, and events should also be modeled using this ontology. I defined an extension for the ontology, that allows modeling dynamic behaviour. It describes changes (e.g. appears, disappears, moves) of objects, properties or relations. Using this aspect, based on the static model a dynamic context can be described.

OWL2 ontology language is used to represent the context models, which language provides $\mathcal{SROIQ}(\mathcal{D})$ expressive power.

In most cases (e.g. in case of autonomous robots) the context model describes a real world that consists of 3D objects and dynamic behaviour of these objects. For demonstration and visualization purposes the context model can be represented in a visualization language (e.g. X3D is an open-standard format, that is able to describe 3D scenes and objects).

C. Test model

The testing is based on a **behaviour specification** of the control, that are relations between input (context) and output (actions). The specification consists of a set of scenarios that represent behaviour. Such scenarios can be created manually, even on the basis of informal requirements, using the elements of the context model and action model. For this purpose a scenario language is used. [5]

The **test data** is a **context model** that is conforming to the context ontology. The **test case** contains every necessary information to execute a test on the **system under test**. It contains the agent's **initialization**, the **workload** that should be executed by the system under test, the description of the **expected behaviour** and the **test data**.

The efficiency of the testing (a set of test cases) can be measured using **coverage metrics** defined in the following way:

- *Coverage of the ontology* measures the coverage of the concepts defined in the context ontology, considering the hierarchical taxonomy. It can take advantage of the hierarchy of the ontology.

- *Coverage of context patterns* measures the coverage of various patterns defined using the concepts of the ontology.

Besides testing the correctness of behaviour, the testing of its robustness can also be addressed. Robustness is an attribute of dependability and it is used to measure the degree to which a system operates correctly in the presence of **exceptional inputs** or **stressful environmental conditions**. In this case the expected input attributes of the agent program are taken from the context model based on the range definitions of the concept attributes or based on the extreme values defined in the ontology constraints.

D. Test generation

In the proposed algorithm, **evolutionary strategy** will be used to generate test data for black-box software testing. Using evolutionary strategy means using **genetic operators** - selection and mutation - in order to generate better solution in every iteration of the algorithm. [6]

The genetic algorithm is based on an initial set of test data (**initial population**), that will be manipulated using the **genetic operators**. These operators can select some elements from the population, can modify an element (mutation) and can add new generated element to the population in order to achieve the goal of the algorithm. The set of the allowed operators and the size of the initial population depend on the used genetic strategy. The goal of the algorithm can be defined using **metrics** that can be measured on the the instances or on the whole population.

The steps of the presented algorithm are the following:

- **Initialize the test generation** with an initial context model based on *context patterns* derived from the scenarios.
- **Complete measures on context models** in order to determine the next step of the algorithm.
- **Assign execution probability** to the genetic operators based on the set of measures in order to balance the different test generation aspects.
- **Make some modification on context models** using genetic operators to get new context models.

During the implementation of the algorithm the main challenge is the connection between the context ontology, metrics and genetic operators.

The definition of the **metrics** used by the algorithm is based on the context ontology. These metrics measure how good is the generated context model in the view of the scenario being tested. In my approach the metrics will be defined on the basis of context patterns, they describe the required objects and properties. These patterns will be implemented using **graph patterns**.

The **genetic operators** are model manipulation steps and will be implemented as **model transformation rules**. The execution semantics of transformation rules provides the nondeterminism that is required by the genetic algorithms.

The final step is making **connection between metrics and genetic operators**. On the basis of the set of metrics the algorithm can get a set of measures on the generated test data. Depending on the measures different **execution probability** will be assigned to the different transformation rules, and the transformation engine can choose one transformation rule based on the probabilities. The connection between patterns (related metrics) and transformation rules (operators) will be described using **test generation rules**.

III. Forklift example

All the presented test generation steps (metrics, model transformation rules) can be implemented using **model manipulation** technologies. In the following this implementation will be presented using a Forklift example based on industrial use cases proposed in the R3-COP project [7].

Cooperating forklifts are Laser Guided Vehicles (LGVs) that physically move the goods and act as the link between the different machines within a warehouse environment. LGVs are controlled by a central unit which plans their path constructed on the basis of the warehouse map. Each forklift knows its own position calculated using a laser positioning system and the velocity using the wheel encoder information. The context recognition is based on stereo cameras. The actions are defined as *put up things, move to a position and put down things*.

The proposed **scenario** is the following: The forklift should move an object from one position to an other one and during the operation it should not collide other objects.

First, the **context ontology** of the domain should be constructed. A simplified ontology of the forklift domain is shown on figure 2. There can be seen the **general concepts** (*position*, *dimension*, *static and dynamic objects*) and the specific elements **inherited from the use case definition** (*forklift, pallet, shelf, good*).

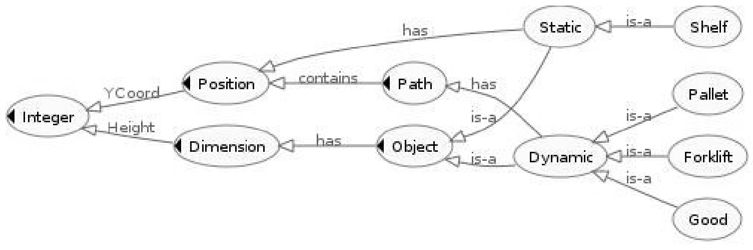
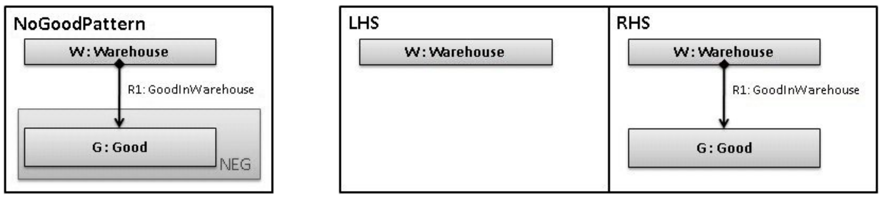


Figure 2: Simplified context ontology

The existence of simple pattern shown in figure 3a is used as a metric in test generation. (If the scenario prescribes to move some object to a different position then there should be a movable object in the context model). This pattern describes, that there is no *good* object in the model.

In figure 3b an example rule is shown, that places a new good object in the context model. This sample rule can be executed more than once, in order to generate goods in the test context.



(a) Sample pattern (b) Sample transformation rule

Figure 3: Forklift test generation examples

IV. Conclusion

I presented a method that extends the test generation algorithm described in [6] with context ontologies. Namely I proposed an implementation of the genetic algorithm based test generation that is supported by **ontology based context description and coverage metrics**.

The construction of the patterns, genetic operators and test generation rules requires domain-specific knowledge. Accordingly, most of these elements can be constructed by domain experts on the basis of the scenario model and context ontology.

References

[1] B. Henderson-Sellers, "Bridging metamodels and ontologies in software engineering," *Journal of Systems and Software*, 84(2):301 – 313, 2011.

[2] S. Bechhofer, "Owl web ontology language reference. w3c recommendation," Feb. 2004.

[3] William D. Clinger, "Foundations of Actor Semantics," Tech. Rep., MIT Artificial Intelligence Laboratory, 1981.

[4] S. Russell and P. Norvig, *Artifical Intelligence. A Modern Approach.*, Pearson Education Inc., second edition, 2003.

[5] Z. Micskei and H. Waeselynck, "The many meanings of uml 2 sequence diagrams: a survey," *Software and Systems Modeling*, pp. 1–26, 2010, 10.1007/s10270-010-0157-9.

[6] János Oláh, "Test data generation using metaheuristics," in *Proc. of the 18th PhD Mini-Symposium*, 2011.

[7] "R3-COP: robust and safe mobile cooperative autonomous systems," URL: https://www.artemis-ju.eu/projects.

GRAPH TRIGGERS AND INCREMENTALLY EVALUATED QUERIES OVER EMF MODELS

Gábor BERGMANN
Advisor: Dániel VARRÓ

I. Introduction

As model management platforms are gaining more and more industrial attraction, the importance of techniques for processing large models is also increasing. A further challenge is coping efficiently (in terms of computational performance and required manual effort) with the constantly evolving nature of models. A model management platform can greatly support the declarative processing of models with the following three features:

- efficiently evaluated declarative model queries,
- change-driven [1] reactive behaviour triggered by changes captured using model queries,
- and declaratively defined model manipulation operations.

Scenarios where such features offer great benefits include model transformation (especially live synchronization [2]), code generation, domain specific behaviour simulation and model validation.

A leading industrial modeling ecosystem, the Eclipse Modeling Framework (EMF [3]), provides different ways to query the contents of models. These approaches range from (1) the use of high-level declarative constraint languages (like OCL [4]) to (2) a dedicated query language [5] resembling SQL, or, in the most basic case, (3) manually programmed model traversal using the generic model manipulation API of EMF. However, industrial experience shows scalability problems of complex query evaluation over large EMF models, taken e.g. from the automotive domain, especially when confronted with evolving models.

To address these issues in the context of EMF, a novel solution for model queries was developed in cooperation with several colleagues. EMF-INCQUERY [6] automatically provides efficient incremental evaluation, without manual coding, to cope with the evolving nature of models. This paper presents an overview of EMF-INCQUERY, then investigates the possibility of applying similar techniques to provide declarative model manipulation and a triggering mechanism.

II. Background: EMF and GT

The Eclipse Modeling Framework (EMF [3]) provides automated code generation and tooling (e.g. notification, persistence, editor) for Java representation of models. EMF models consist of an (acyclic) containment hierarchy of model elements (EObjects) with crossreferences – some of which may only be traversed by programs in one direction (unidirectional references). Additionally, each object has a number of attributes (primitive data values).

EMF uses Ecore metamodels to describe the abstract syntax of a modeling language. The main elements of Ecore are the following: EClass, EAttribute and EReference. EClasses define the types of EObjects, enumerating EAttributes to specify attribute types of class instances and (unidirectional) EReferences to define association types to other EObjects. Some EReferences additionally imply containment.

Example: Figure 1(a) shows a simple example metamodel for a State Machine. The two EClasses are “Machine” and “State”. Each Machine contains States through a containment EReference called “states”, and one of them is marked by the “current” EReference to denote the current State of the Machine. Possible transitions are indicated by “next” EReferences between States.

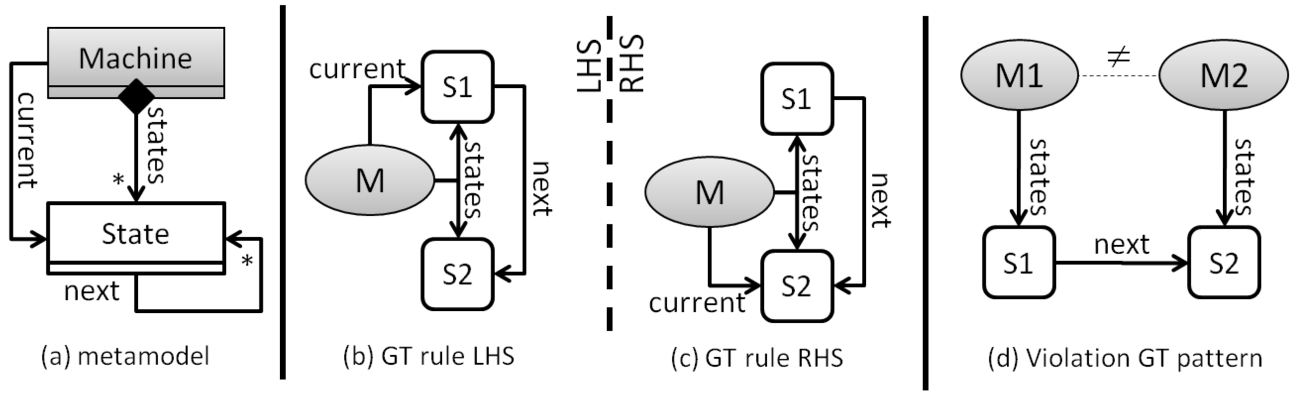


Figure 1: State Machine example

Graph patterns [7] constitute an expressive formalism used for various purposes in Model Driven Development, such as defining declarative model transformation rules, defining the behavioral semantics of dynamic domain specific languages, or capturing general purpose model queries including model validation constraints. A graph pattern (GP) represents conditions (or constraints) that have to be fulfilled by a part of the instance model. A basic graph pattern consists of structural constraints prescribing the existence of nodes and edges of a given type (or subtypes, subject to polymorphism). Some languages (e.g. VIATRA2 [7]) include a way to express negation, resulting in an expressive power equivalent to first order logic [8]. Attribute constraints are also a typical feature. A match of a graph pattern is a group of model elements that have the exact same configuration as the pattern, satisfying all the constraints.

Graph patterns can form the basis of declarative model queries, where the pattern spacifies what arrangement of elements is sought after, not how or where to find them. In an EMF context, each node in the pattern represents an EObject (EMF instance object), and the type of the node identifies the EClass of the object. This feature is useful to select only those model elements that conform to a certain type. Furthermore, the pattern nodes are connected by directed edges, annotated by an EReference type (or containment), to express how these elements reference each other. Finally, attribute constraints filtering and comparing the attributes of these elements can also be added.

Example: Figure 1(b-d) show three simple graph patterns over graph models that conform to the metamodel depicted in Figure 1(a). In particular, the pattern in Figure 1(b) identifies a Machine M and two of its States $S1$ and $S2$, where $S1$ is the current state and $S2$ is one of the possbile successor states. The pattern in Figure 1(c) is similar, but $S2$ is the current state and $S1$ is one of its predecessors. Finally, the pattern in Figure 1(d) captures two different Machines $M1$ and $M2$, with states $S1$ and $S2$ respectively, where $S2$ is marked as a successor of $S1$.

Graph transformation (GT) [9] provides a high-level rule and pattern-based manipulation language for graph models. Graph transformation rules can be specified by using a left-hand side (LHS) graph pattern determining the applicability of the rule, and a right-hand side (RHS) graph pattern which declaratively specifies the result model after rule application. Elements that are present only in (the image of) the LHS are deleted, elements that are present only in the RHS are created, and other model elements remain unchanged.

Example: The GT rule formed from Figure 1(b) as LHS and Figure 1(c) as RHS describes how the state machine can advance its state. The LHS captures the current state and a successor state, which are needed to fire the rule. When the rule is applied on a match of the LHS, it is substituted with the image of the RHS, meaning that $S2$ is marked as the current state from that time on.

An important performance issue associated with large models is that they have to be reconsidered again and again after each small change, causing a significant overhead. Incrementality is therefore a valuable property of model processing mechanisms in the context of evolving models. In case of GT-based approaches, most of the computational complexity is associated with graph pattern matching, thus incremental pattern matching (INC) [10] is a promising way to address the performance problems.

INC techniques rely on a cache which stores the results of a query explicitly. The result set is readily available from the cache at any time without additional search, and the cache is incrementally updated whenever (elementary or transactional) changes are made to the model. As results are stored, they can be retrieved in constant time, making query evaluation extremely fast. The trade-off is increased memory consumption, and increased update costs (due to continuous cache updates).

III. EMF-INCQUERY

The aim of the EMF-INCQUERY approach is to bring the benefits of graph pattern based declarative queries and incremental pattern matching to the EMF domain. The advantage of declarative query specification is that it achieves (efficient) pattern matching without time-consuming, manual coding effort associated to ad-hoc model traversal. While EMF-INCQUERY is not the only technology for defining declarative queries over EMF [4, 5], its distinctive feature is incremental pattern matching. Thanks to this matching technique, EMF-INCQUERY has special performance characteristics suitable for scenarios such as on-the-fly well-formedness validation. See [6] for measurements revealing how EMF-INCQUERY can be several orders of magnitude more efficient than other approaches.

Additionally, some shortcomings of EMF are mitigated by the capabilities of EMF-INCQUERY, such as cheap enumeration of all instances of a certain type, regardless of where they are located in the resource tree. Another such use is the fast navigation of EReferences in the reverse direction, without having to augment the metamodel with an EOpposite (which is problematic if the metamodel is fixed, or beyond the control of the developer).

EMF-INCQUERY provides an interface for each declared pattern for (i) retrieving all matches of the pattern, or (ii) retrieving only a restricted set of matches, by binding (a-priori fixing) the value of one or more pattern elements (parameters).

In both cases, the query can be considered instantaneous, since the set of matches of the queried patterns (and certain subpatterns) are automatically cached, and remain available for immediate retrieval throughout the lifetime of the EMF ResourceSet. Even when the EMF model is modified, these caches are continuously and automatically kept up-to-date using the EMF Notification API. This maintenance happens without additional coding, and works regardless how the model was modified (graphical editor, programmatic manipulation, loading a new EMF resource, etc.).

IV. Graph Triggers over EMF

EMF-INCQUERY is useful in itself to provide an expressive and efficiently evaluated query language. However, the incrementality of the result set opens up the possibility of considering graph changes as events. As proposed with co-authors in previous work [2] outside the context of EMF, changes of the model graph, as captured on an arbitrary granularity defined by a graph pattern, can drive the execution of Graph Triggers. There are two basic kinds of triggers: those that are activated by the appearance of a match of a graph pattern, and those that are fired on a disappearance (a language for capturing more precise conditions of trigger guards is proposed in [1]). A trigger is thus specified by this guard condition and a reaction that can be defined by an arbitrary sequence of Java code.

Example: The graph pattern in Figure 1(d) identifies a violation of the well-formedness of the state machine model, in which a “next” edge leads to a different Machine. By registering a Graph Trigger that is activated on the appearance of this graph pattern and notifies the modeling expert of the mistake, the domain-specific modeling environment can benefit from on-the-fly well-formedness checking.

To reflect the changes in the match set during a sequence of model manipulation, a device called delta monitor continuously and incrementally maintains such a difference. This makes the implementation of the trigger engine a matter of regularly checking the delta monitors and firing triggers when necessary. The timing of this check must be chosen such that the incremental pattern matcher already reflects a consistent state of the model. In case of EMF Transactions, the triggers are fired at the pre-commit phase, so that they can modify the model.

The usefulness of such a service was demonstrated by applying it in multiple ways. As a demonstration of live transformation (pioneered in [2]) over EMF, triggers facilitated an on-the-fly bidirectional synchronization between two views of the same model in two different modeling formalisms of the SecureChange EU research project. A second likely usage, similar to the example given above, is on-the-fly checking of structural well-formedness properties, as demonstrated in [6]. As a third usage, change itself was used as a source of information (in addition to the current state of the model) in the same SecureChange case study [1].

Finally, the possibility of declarative model manipulation over EMF was also investigated, although the implementation on top of EMF-IncQuery remains future work. The natural choice of formalism to apply on EMF would be GT rules, investigated in e.g. [11]. The main challenge is the hierarchical nature of the EMF Resources, implying that new elements can only be created in a container element, and that element deletions cause the removal of the entire containment sub-tree, which can cause inconsistency if e.g. the same GT rule connects a new edge to a deleted element. To address these issues, [11] imposes some consistency restrictions on GT rules, and these conditions assume complete knowledge of the Ecore metamodels, which may be unrealistic.

V. Conclusion

The proposed technology applies academically researched model transformation features (namely incremental pattern matching and graph triggers) in the industrial platform of EMF; benefits manifest in use cases including live model synchronization and structural well-formedness checking.

References

[1] G. Bergmann, I. Ráth, G. Varró, and D. Varró, "Change-driven model transformations: Taxonomy and language," *Journal of Software and Systems Modeling*, 2010, Submitted.

[2] I. Ráth, G. Bergmann, A. Ökrös, and D. Varró, "Live model transformations driven by incremental pattern matching," in *Theory and Practice of Model Transformations*, vol. 5063/2008 of *Lecture Notes in Computer Science*, pp. 107–121. Springer Berlin / Heidelberg, 2008.

[3] The Eclipse Project, *Eclipse Modeling Framework*, http://www.eclipse.org/emf.

[4] The Eclipse Project, *MDT OCL*, http://www.eclipse.org/modeling/mdt/?project=ocl.

[5] The Eclipse Project, *EMF Model Query*, http://www.eclipse.org/modeling/emf/?project=query.

[6] G. Bergmann, Á. Horváth, I. Ráth, D. Varró, A. Balogh, Z. Balogh, and A. Ökrös, "Incremental evaluation of model queries over EMF models," in *Model Driven Engineering Languages and Systems, 13th International Conference, MODELS'10*. Springer, Springer, 10/2010 2010, Acceptance rate: 21%.

[7] D. Varró and A. Balogh, "The Model Transformation Language of the VIATRA2 Framework," *Science of Computer Programming*, 68(3):214–234, October 2007.

[8] A. Rensink, "Representing first-order logic using graphs," in *International Conference on Graph Transformations (ICGT), LNCS 3256*, pp. 319–335. Springer, 2004.

[9] H. Ehrig and et al., Eds., *Handbook on Graph Grammars and Computing by Graph Transformation*, vol. 2, World Scientific, 1999.

[10] G. Bergmann, A. Ökrös, I. Ráth, D. Varró, and G. Varró, "Incremental pattern matching in the VIATRA model transformation system," in *Graph and Model Transformation (GraMoT 2008)*, G. Karsai and G. Taentzer, Eds. ACM, 2008.

[11] E. Biermann, C. Ermel, and G. Taentzer, "Precise semantics of EMF model transformations by graph transformation," in *MoDELS '08: Proceedings of the 11th international conference on Model Driven Engineering Languages and Systems*, pp. 53–67. Springer-Verlag, 2008.

Criteria Evaluation-driven State Space Exploration of Graph Transformation Systems

Ábel HEGEDÜS
Advisor: Dániel VARRÓ

I. Introduction

Model transformation is a common technique in Model Driven Engineering to design, analyze and simulate various kinds of models. In case of model analysis, forward transformations usually carry out an abstraction to enable efficient formal validation. However, mapping the information gathered from validation back to the original models (i.e. *back-annotation*) is a challenge due to the abstraction gap between the source and target languages.

Graph transformation (GT) [1] is a mathematical formalism which provides a rule-based manipulation of graph models. GT rules are defined as two graph patterns, where application of the rule is replacing an occurrence of the first pattern in the graph with the second pattern. *Graph transformation system* (GTS) is defined as graph, where nodes are instance graphs and edges are the applications of GT rules. Thus, it describes the reachable instance graphs from a given host graph. GTS are frequently used at many application areas (e.g. creating models or modeling the behavior of systems). For example, in [2] graph transformations are used for software architecture reconfigurations and the reconfiguration actions of services will be captured by a graph transformation system. Fig. 1 shows a simple reconfiguration where a faulty service (*Do*) having a backup (*St*) fails over to it's standby service, that becomes active (*Ac*).

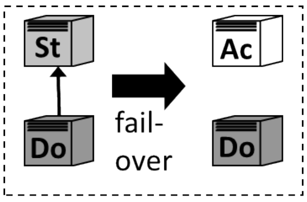


Figure 1: Service reconfiguration example

Place-Transition (P/T) net is a formalism often used to describe the dynamic behavior of systems, P/T nets are defined as bipartite graphs, where nodes can be either places or transitions. Tokens placed in places represent the state of the system, while transitions represent state changes by removing and adding tokens to connected places. The abstraction of GTS, e.g. as Place-Transition (P/T) nets, are often used for termination analysis [3], optimization [4], verification [5, 6] or finding errors in the implementation (debugging) [7].

However, the results of these analysis methods are usually not *execution paths* (ordered sequence of rule applications or transition firings) for the GTS but a more abstracted information, such as an *occurrence vector* ($\overline{v}_o$) containing only the number of transition executions instead of their exact order.

In order to successfully retrieve the rule application sequence (execution path or trajectory) on the GTS-level, the state space of the GTS is explored using the information collected by back-annotating the analysis results. Our goal is to efficiently identify a feasible execution path of a GTS corresponding to a given occurrence vector (if such path exists). During the exploration of the state space, possible execution paths are examined to check their compliancy with the analysis information.

II. State Space Exploration of Graph Transformation Systems

In the service reconfiguration example, the initial (or host) graph of the GTS represents the current configuration of services. Furthermore, there are certain Quality of Service (QoS) requirements about the desired configuration (e.g. the number of available services) and *global constraints* which must be satisfied at all times (e.g. the maximum amount of services) that can be described as graph patterns. The state space exploration of the GTS is performed in order to find a sequence of manipulations which leads to a state satisfying QoS requirements (*goals*) and each intermediate state satisfies the constraints.

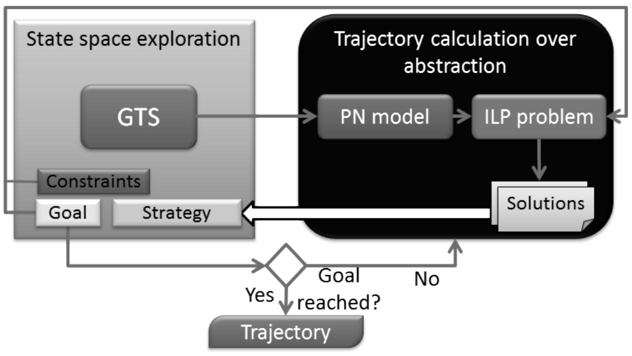


Figure 2: Overview of the approach

Occurrence vector-based search strategy. In [4], the computation of an optimal rule application sequence is performed by encoding the P/T net abstraction (detailed in [3]) of the GTS (along with the desired *goals* and *global constraints*) into an integer linear programming (ILP) problem (illustrated in Fig. 2). The *solution* of this problem is a candidate *transition occurrence vector* ($\overline{v}_o$), which counts the number of rule applications. Since the abstraction does not guarantee that this vector corresponds to an executable *trajectory*, its feasibility should be checked on the GTS-level. However, in the original approach, $\overline{v}_o$ was used in the GTS state space *exploration strategy* by only allowing occurrence vector compliant execution paths to be explored. Therefore, it did not help in selecting the most promising execution path or cutting the search on a given path when it is guaranteed to be infeasible. Note that an alternative approach could use backward searching with planning techniques from artificial intelligence if the final state is precisely known, but in our case the goals hold only partial information about the state.

Selection and cut-off criteria for search strategy. In this paper, I propose additional techniques, which use the occurrence vector as a hint, to guide the state space exploration to further increase the performance of the algorithm. The main features of these new techniques are (a) using the rule (or transition) *dependency graph* (G_d) computed from the GTS (or P/T net) [8] to have a global view on the effects of rule applications; (b) defining *selection criteria* on the applicable rules (transitions) at a given state; and (c) defining *cut-off criteria* on the paths. Criteria defined in both (b) and (c) depend on G_d and the application numbers for the rules (transitions) in $\overline{v}_o$. A more detailed description of criteria types is given in the following:

- *Cut-off criteria (Cr^{cut})* inspect the current state of the dependency graph and return a boolean result which is true if further exploration of the current branch cannot lead to a goal state with a compliant trajectory. In this case, the exploration continues from an other state instead of executing a GT rule in the current state.
- *Selection criteria (Cr^{sel})* take the dependency graph and define an ordering of the enabled GT rules. A given GT rule r_i is placed before an other rule r_j, if the execution of r_i is more promising, based on the criteria and the current state, than the execution of r_j. The ordering is used instead of selecting the most promising rule since the exploration may lead to a cut-off on the most promising branch. In this case the next GT rule in the ordering is executed.

III. Criteria Definition for Dependency Graphs

In this section the main concepts regarding graph transformation rule dependency and transition occurrence vectors are described which are relevant for the definition of selection and cut-off criteria for the state space exploration. Next, the building blocks which are used to construct arbitrary criteria are introduced, followed by criteria examples usable for the search strategy.

Definitions. Assume that there is a GTS including a set of GT rules (r_i) and an *initial graph* (G_I). Furthermore, as a result of critical pair analysis [9] there is a dependency graph (G_d, illustrated in Fig. 3) of the rules, where each r_i is a node (n_i) and there is a directed arc from n_i to n_j if r_j has sequential dependency on r_i (i.e. the application of r_i may affect the match set of r_j). Note that there may be arcs in both direction between two nodes. In this paper, n_i► refers to the set of nodes which have sequential dependency on n_i, while ◄n_i refers to nodes on which n_i has sequential dependency (both sets illustrated for n_c in Fig. 3).

Finally, the candidate transition occurrence vector is a solution of the state equation in the P/T net, where $\overline{v}_o[i]$ is the number of times that r_i is applied during the execution. During the state space exploration, the number of times r_i has been applied in a given path is stored in the *application vector* ($\overline{v}_a$) as $\overline{v}_a[i]$. Furthermore, an execution path of the state space exploration is *compliant* with $\overline{v}_o$ if $\overline{v}_a \leq \overline{v}_o$. Throughout the paper I use the difference between $\overline{v}_o[i]$ and $\overline{v}_a[i]$ ($\overline{v}_o[i] - \overline{v}_a[i]$) as the *remaining application number* of r_i ($\#_i$, illustrated in Fig. 3 at the bottom-left of each node).

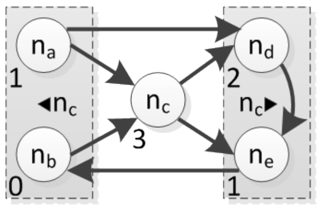


Figure 3: Example dependency graph

Building blocks for criteria. In our approach, the criteria are constructed using starting point identifiers and a well-defined set of operators (i.e. *building blocks*), which can represent navigation over the graph edges, numerical and logical functions between subcriteria, ordering of results and quantifiers. Starting points behave as *operands* and create a criteria together with an operator. The resulting criteria (called a *subcriteria*) can be also an operand to create more complex criteria. Throughout the paper I refer to an operator as *enclosing* for the operands which it is combined with.

Example criteria for guided search. Using these definitions, the following cut-off and selection criteria examples can be defined for the search strategy:

Permanently disabled rule (Cr^{cut}_{Pdr}): When there is a disabled rule r_i which still has to be applied based on $\overline{v}_o$, but the application of any rule in ◄n_i would violate result in a non-compliant path, the current path can be cut.

Maximum forward-dependant application path (Cr^{sel}_{Mfd}): Among the applicable GT rules at any given state of the exploration, the one with the most (transitively) dependant rule applications should be executed first. The selection is based on calculating the effect of each applicable rule using G_d.

IV. Evaluation of Criteria

The definition of the criteria and the evaluation algorithm is separated in the approach since different algorithms are suitable for different classes of dependency graphs. For example a graph without bidirectional edges should be handled differently than one including many of those. Similarly, loops and strongly connected components in the dependency graph call for more sophisticated algorithms.

A simple algorithm which may serve as a basis for advanced ones is described in the following:

First, one of the starting points (S) is selected from the criteria to decide upon which rules will the evaluation iterate through. Next, each node, that satisfies S, is evaluated after each other by iteratively applying the enclosing operators.

Navigation operators return reachable nodes in the graph that are not included in the already visited nodes and the $\#_i$ of each n_i is summed and stored as a partial result (R_p). Numerical and logical operators are applied as implied by their definition and result in R_p and boolean values (R_b), respectively.

Finally, ordering operator places the current node in the appropriate position in the list of evaluated nodes while quantifier operators decide whether Cr^{cut} is satisfied based on R_b (i.e. if Cr^{cut} is satisfied, the branch is cut). The final result of Cr^{sel} therefore is an ordered list of nodes (corresponding to enabled GT rules), where the first node in the list is deemed the most promising for execution. It is important to note that the exploration strategy is often constructed using several criteria, where the combination of Cr^{cut} is trivial (i.e. the branch is cut if any of Cr^{cut} is satisfied), while the combination of Cr^{sel} requires careful consideration.

Evaluation example. The evaluation of cut-off and selection criteria is illustrated on the dependency graph in Fig. 3. Assume that r_a is enabled and r_c is disabled in the current state. The Cr^{cut}_{Pdr} is evaluated on n_c, first the algorithm navigates backward to nodes n_a and n_b and finds that $\#_a$ is not zero, therefore the criteria is not satisfied and the branch is not cut.

Then the Cr^{sel}_{Mfd} is evaluated on n_a, as a result of forward navigation, n_c and n_d are visited and $\#_c$ is summed up with $\#_d$ in R_p (5). Since the navigation is transitive, n_e is also visited in the next step, and $\#_e$ is added to R_p (6). Next n_b is visited, followed by n_c, but that is already visited, so the transitive navigation stops, and $R_p = 6$. If n_e is also enabled, the same criteria is evaluated to $R_p = 5$ (after visiting n_b, n_c and n_d), therefore n_e is placed after n_a in the result list.

Once r_a is applied ($\#_a$ becomes 0) and r_c is still disabled, the evaluation of Cr^{cut}_{Pdr} will return true, as both $\#_a$ and $\#_b$ are zero. Thus the branch is cut and a different branch should be explored.

Conclusion. The state space exploration of GTS appears as a relevant issue in several application areas, however it remains a challenging problem. In this paper the back-annotation of analysis results in the P/T net abstraction of GTS and the inherent dependencies of GT rules are utilized for driving the exploration by evaluating arbitrarily defined cut-off and selection criteria. The application of the presented techniques can increase the efficiency of the state space exploration. Note that the complexity and scalability of the approach are not discussed in the paper, although early results indicate that the approach is usable for meaningful cases.

Ongoing implementation efforts extend upon the CSP(M) framework [10] to provide criteria evaluation-driven state space exploration, while future work includes investigating the usage of graphs created from critical pairs, incorporating goals and constraints into the graphs and the definition of sophisticated criteria and evaluation algorithms.

References

[1] G. Rozenberg, Ed., *Handbook of Graph Grammars and Computing by Graph Transformations: Foundations*, World Scientific, 1997.

[2] L. Baresi, R. Heckel, S. Thöne, and D. Varró, "Style-based modeling and refinement of service-oriented architectures," *Journal of Software and Systems Modelling*, 5(2):187–207, 2006.

[3] D. Varró, S. Varró-Gyapay, H. Ehrig, U. Prange, and G. Taentzer, "Termination Analysis of Model Transformations by Petri Nets," in *Proc. Third International Conference on Graph Transformation (ICGT 2006)*, A. Corradini, H. Ehrig, U. Montanari, L. Ribeiro, and G. Rozenberg, Eds., vol. 4178 of *LNCS*, Natal, Brazil, 2006. Springer.

[4] S. Varró-Gyapay and D. Varró, "Optimization in Graph Transformation Systems Using Petri Net Based Techniques," *ECEASST*, 2, 2006, Selected papers of Workshop on Petri Nets and Graph Transformations.

[5] B. König and V. Kozioura, "Counterexample-Guided Abstraction Refinement for the Analysis of Graph Transformation Systems," in *Tools and Algorithms for the Construction and Analysis of Systems*, H. Hermanns and J. Palsberg, Eds., vol. 3920 of *LNCS*, pp. 197–211. Springer Berlin / Heidelberg, 2006.

[6] L. Baresi and P. Spoletini, "On the Use of Alloy to Analyze Graph Transformation Systems," in *Graph Transformations*, A. Corradini, H. Ehrig, U. Montanari, L. Ribeiro, and G. Rozenberg, Eds., vol. 4178 of *LNCS*, pp. 306–320. Springer Berlin / Heidelberg, 2006.

[7] M. Wimmer, G. Kappel, J. Schoenboeck, A. Kusel, W. Retschitzegger, and W. Schwinger, "A Petri Net Based Debugging Environment for QVT Relations," in *Automated Software Engineering, 2009. ASE '09. 24th IEEE/ACM International Conference on*, pp. 3 –14, 16-20 2009.

[8] T. Mens, G. Kniesel, and O. Runge, "Transformation dependency analysis - a comparison of two approaches," in *LMO*, R. Rousseau, C. Urtado, and S. Vauttier, Eds., pp. 167–184. Hermès Lavoisier, 2006.

[9] R. Heckel, J. M. Küster, and G. Taentzer, "Confluence of Typed Attributed Graph Transformation Systems," in *In: Proc. ICGT 2002. LNCS*. Springer, 2002.

[10] Á. Horváth and D. Varró, "CSP(M): Constraint Satisfaction Problem over Models," in *Model Driven Engineering Languages and Systems, 12th International Conference, MODELS 2009, Denver, CO, USA, October 4-9, 2009. Proceedings*, A. Schürr and B. Selic, Eds., vol. 5795 of *LNCS*. Springer, 2009.

DEF-USE ANALYSIS OF MODEL TRANSFORMATION PROGRAMS WITH PROGRAM SLICING

Zoltán UJHELYI
Advisor: Dániel VARRÓ

I. Introduction

Model transformations, used for various tasks, such as formal model analysis or code generation are key elements of model-driven development processes. As the complexity of developed model transformations grow, ensuring their correctness becomes increasingly difficult. Nonetheless, error detection is critical as errors can propagate into the target application.

Various analysis methods are being researched for the validation of model transformations, such as model checking [1] or static analysis [2] based approaches. However, there are further methods supporting the development and validation of traditional programming languages, and their application to model transformation programs can raise the maturity of the technology.

Program slicing [3] techniques are used to calculate parts (called *slices*) of a program that (potentially) affect the values computed at a selected point of the program. The generated slices can be used in several ways: (1) they can be simply displayed to the developer as a filtered version of the program; (2) as the slices represent a small, cohesive unit of the program, various types of analysis can be executed on them instead of the entire program, resulting in performance and/or precision gain; and (3) as the dependencies between program statements are explored to generate the slices, this information can be re-used in different analysis techniques.

The current paper presents a slicing approach for model transformation programs based on *dependence graphs*. The generated slices are then used to perform *def-use analysis* to check for *uses of undefined variables* and *unused variables* (approach 3). We demonstrate this approach using the transformation language of the VIATRA2 model transformation framework [4] over a Petri net simulator transformation, however it is applicable to other languages and transformations as well.

Petri nets are bipartite graphs with two disjoint set of nodes: *Places* and *Transitions*. *Places* can contain an arbitrary number of *Tokens*, that represents the state of the net (marking). The simulator changes this state by a process called *firing*: from every input *Place* of a *Transition* a *Token* is removed (if there is none to remove, the *Transition* must not fire), then to every output *Place* a *Token* is added.

II. Overview of the Approach

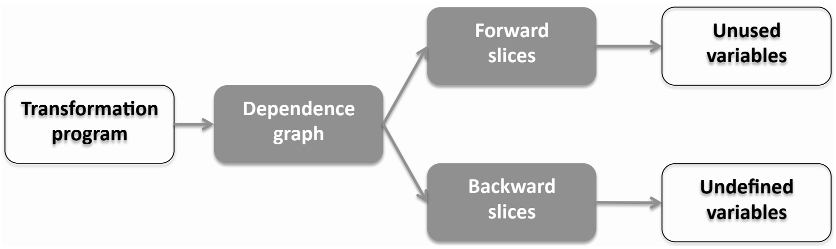


Figure 1: Overview of the Approach

Our analysis process is depicted in Figure 1. At first, the transformation program is read, and the dependencies between the program statements are collected into a *system dependence graph* (SDG) [5].

These graphs are labeled graphs, their nodes represent the program statements, while the edges describe the various dependency relations between them, such as control, data or call dependencies. The calculation of these graphs is detailed in Section III.

The SDG can be used to calculate two different kind of slices: forward and backward slices.

The *backward slicing problem* of a program can be summarized as follows: for a selected program statement we are looking for the (superset of the) statements the selected statement depends on. Such slices can be calculated by traversing the SDG: starting from the node of the selected statement every statement has to be reached, that the selected statement depends on (both directly or indirectly).

Backward slices can be used to detect uses of *undefined variables*, that are read before having been defined. If the backward slice of a statement using a variable does not contain a definition (or assignment) for the used variable, it should be reported as error. When the slice consists of multiple execution paths, every execution path is checked for a definition. A special case of these undefined variables is the *reading of output parameters*: a parameter of a called procedure that is supposed to be returned is undefined before it is set.

Similarly, the *forward slicing problem* can be summarized as follows: for a selected program statement we are looking for the (superset of the) statements depending on the selected statement. These slices can also be calculated by the traversal of the SDG: the dependencies have to be traversed in the other direction then in case of backward slices.

Unused variables can be detected by looking for the forward slices of variable definitions. In case the slice does not contain at least a single statement reading the variable, the variable is unused. In case of multiple execution paths we check whether the variable is used in at least a single path. The *writing of input parameters* is a special case of unused variables: changes to an input parameter are lost after rule termination.

III. Calculating Dependence Graphs in Model Transformation Programs

The transformation language of VIATRA2 [4] uses the formalism of *graph transformation* (GT) to describe elementary transformation steps between graph models, and uses abstract state machines (ASM) as control structure to build complex transformation programs. The dependence graphs of the GT and ASM rules have to be calculated differently, as GT rules offer a declarative description of transformation steps with loosely defined control flow, while ASM rules define a rich, precisely specified one. In the following we detail the calculation of such dependence graphs for both parts.

A. The Dependence Graph of ASM Rules

ASM rules provide similar control structure as imperative programming languages, making their dependence graph basically the same as the one described in [5]: the statements of the transformation program are represented as nodes, while control, data and call dependencies are depicted as arcs between them.

ASM rules may have directed parameters: incoming parameters are initialized when the rule is called, while outgoing parameters are expected to be initialized by the rule. These dependencies are presented in the dependence graph by incoming/outgoing data dependency arcs.

Block rules, such as `forall` or `let` rules define local variables that have to be considered as dependencies as well. As the language permits having the same variable name to be defined in multiple blocks, the analysis must distinguish between these instances - for this reason the dependence graph uses different variable instances for these elements.

Example 1 *Figure 2 displays an ASM rule of the Petri net simulation program together with its dependence graph. The rule first calls the* isTransitionFireable *pattern as a condition, then applies the* removeToken *GT rule for all places that are results of the* inputPlace *pattern. Then this is repeated similarly for the* outputPlace *pattern and the* addToken *GT rule.*

In the dependence graph solid lines represent call and control dependencies, while jagged lines represent data dependencies - their labels describe the corresponding variable.

The dependence graph describes how the variables are connected. It is important to note, that although the Pl *variables in the two* `forall` *rules have the same name, but describe different variables - these variables are called Pl_1 and Pl_2 respectively.*

The called patterns and GT rules are omitted from the graph for readability reasons - only the calls and entry points are displayed. In order to calculate the required slices, these dependence graphs are also to be calculated.

```
rule fireTransition(in T) = seq {
 // graph pattern call in a condition
 if (find isTransitionFireable(T))
 seq {
  forall Pl
   // graph pattern call
   with find inputPlace(T, Pl)
   // GT rule application
   do apply removeToken(T, Pl);
  forall Place
  // graph pattern call
   with find outputPlace(T, Pl)
  // GT rule application
   do apply addToken(T, Pl);
 }
}
```

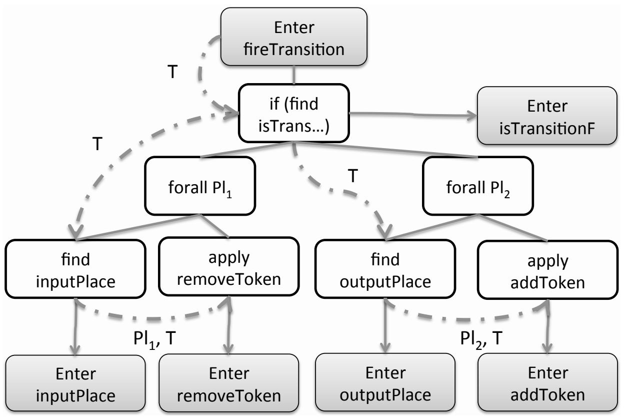


Figure 2: The Dependence Graph of the fireTransition ASM Rule

B. *The Dependence Graph of Graph Transformation Rules*

Graph transformation provides a high-level, declarative rule and pattern-based manipulation language for graph models. GT rules can be specified using a precondition graph (pattern) to decide applicability, and a postcondition graph (pattern) to specify the result model after rule application. To achieve this, the rule application removes all elements, that are only present in the precondition graph, creates all elements, that are only present in the postcondition, and leaves every other element unchanged.

Data dependencies are easy to calculate inside graph patterns, as the pattern graph is described with a series of pattern variable definitions. If such a definition uses another pattern variable, a data dependency is identified. However, when a pattern is called, its incoming parameters might or might not be initialized. Initialized pattern variables depend on the pattern call, but are not modified, while uninitialized parameters are calculated during the pattern matching process. This means, unlike ASM rules, data dependencies of a graph pattern cannot be evaluated completely without looking at its callers.

This external information is available when calculating the dependencies of the caller GT rule: pattern variables of the precondition pattern are initialized, if they are incoming parameters of the rule (explicitely specified in the syntax), while pattern variables of the postcondition pattern are initialized if they were initialized in the precondition.

Example 2 *Figure 3 displays a token manager GT rule of the Petri net simulator with its dependence graph. Its precondition pattern (called* LHS*) consists of a single* Place*, while its postcondition (called* RHS*) consists of a* Place *and a* Token *connected by a* tokens *relation.*

The LHS *pattern does not have any internal dependency, while the* tokens *relation in the* RHS *pattern depends on both variables* Pl *and* To*. As the* Pl *variable is an incoming parameter of the rule, it is an external dependency of the* LHS *pattern. Similarly the* RHS *pattern depends on the* Pl *variable from the* place *pattern, however, the* To *variable is initialized inside the pattern.*

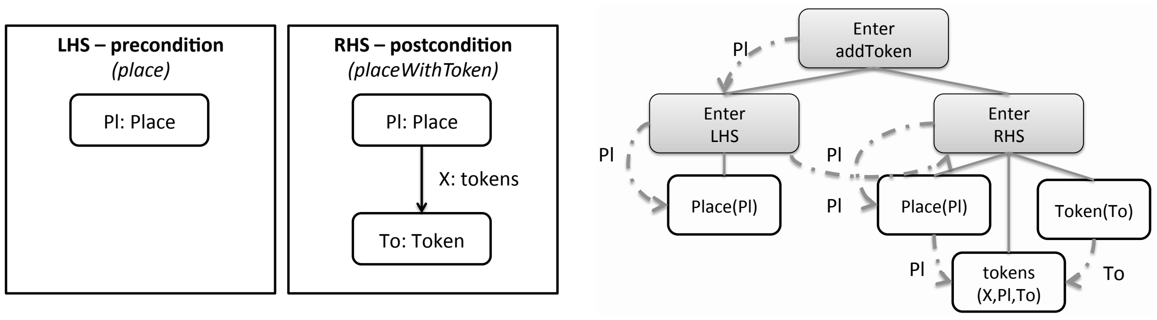


Figure 3: The Dependence Graph of the addToken GT Rule

In case of graph patterns and GT rules it is possible that their application creates modifications in the (input) model, that are not directly visible in transformation program variables. Next to the internal and external data dependencies mentioned before, it is also possible to establish indirect dependencies between graph patterns and GT rules. As the rules look for a match for their input models, and alter them, the applicability of other rules might change as well without any explicit connection between the rules in the transformation program.

A way to detect these dependencies is to consider each model element of the (input) models as a variable of the transformation program. However, as the models can be complex, abstractions are needed for this approach to be feasible. The use of type system provides such an abstraction [6]. This way, each model element can be represented by a type variable, and this type variable could be used to represent possible dependencies between model elements.

IV. Conclusion

In this paper we presented an approach to create slices of model transformation programs based on dependence graphs, and used the slices to perform a def-use analysis: detect undefined variables and unused variable definitions statically.

For the future we plan to find other uses of the generated slices. By displaying the slices to the transformation developer it is possible to get a better understanding of the structure of the transformation program. Similarly, the slices may be used to calculate the set of statements relevant to some condition (e.g., expressed as a graph pattern).

In order to produce smaller slices we plan to use a more precise dependency calculation. To achieve this we will investigate further model abstractions and compare their slice sizes to the one generated using the type system abstraction. The use of dynamic slicing also promises reduced slice sizes since next to the program structure it also uses the input variables to filter out impossible dependencies.

References

[1] A. Rensink and D. Distefano, "Abstract Graph Transformation," *Electronic Notes in Theoretical Computer Science*, 157(1):39–59, May 2006.

[2] J. Bauer and R. Wilhelm, "Static Analysis of Dynamic Communication Systems by Partner Abstraction," in *Static Analysis*, pp. 249–264. Springer Berlin / Heidelberg, 2007.

[3] D. Binkley, S. Danicic, T. Gyimóthy, M. Harman, Ákos Kiss, and B. Korel, "A formalisation of the relationship between forms of program slicing," *Science of Computer Programming*, 62(3):228–252, Oct. 2006.

[4] D. Varró and A. Balogh, "The Model Transformation Language of the VIATRA2 Framework," *Science of Computer Programming*, 68(3):214–234, October 2007.

[5] S. Horwitz and T. Reps, "The use of program dependence graphs in software engineering," in *Proceedings of the 14th international conference on Software engineering*, pp. 392–411, Melbourne, Australia, 1992. ACM.

[6] Z. Ujhelyi, "Static type checking of model transformation programs," in *Graph Transformations*, vol. 6372 of *Lecture Notes in Computer Science*, pp. 413–415. Springer Berlin / Heidelberg, 2010.

ITERATIVE DECISION FEEDBACK EQUALIZATION FOR FBMC SYSTEMS

Zsolt KOLLÁR
Advisor: Gábor PÉCELI

I. Introduction

The introduction of Cognitive Radio (CR) triggered a new interest for researching alternatives of Orthogonal Frequency Division Multiplexing (OFDM) multicarrier systems [1]. In this paper we focus on the Filter Based Multicarrier (FBMC) systems as a candidate competing with OFDM in CR scenarios. OFDM is a widely adopted modulation scheme. Despite its many advantages it has drawbacks which must be taken into consideration during design. An important aspect is its spectral properties, especially the out of band radiation which is considered moderate in case of OFDM, but in this respect FBMC has better performance. The transmission data rate will be also higher due the fact that FBMC does not apply a cyclic prefix (CP). On the other hand, the channel equalization is more complex compared to OFDM due to the inter symbol interference (ISI) caused by the multipath channel. In this paper we focus on channel equalization for FBMC and OFDM systems. The basic problem of equalization for FBMC systems is presented in [2]. First we briefly describe the two modulation schemes. In Section II. we describe the baseband signal model for the transceiver chain with the multipath communication channel. Then we introduce the channel equalization schemes that we intend to analyze in four subsections: first the basic per subchannel Zero Forcing (ZF) and Minimum Mean Square Error (MMSE), then the modified MMSE which minimizes the ISI and the averaged MMSE technique. In the last subsection, we present a new decision feedback equalization technique. Finally we verify them via bit error rate simulations and we assess their performance by comparing them to OFDM systems.

A. OFDM

In this section we give only a short description of the OFDM system [3]. First, the binary information is mapped to constellation symbols X. The time domain samples of an OFDM symbol are generated via IFFT. Then, the CP is added to the OFDM symbol to form the transmitted signal s_n.

B. FBMC

FBMC systems are derived from the orthogonal lapped transforms and filter bank theory [4]. The block diagram of an FBMC transmitter can be seen in Fig. 1. Similar to OFDM the bits are first mapped to complex constellation values X, then the real parts are modulated by a cosine filter bank where only the even-index subbands are used and the imaginary parts are modulated by a sine filter bank where only the odd-index subbands are modulated. An offset of half of the symbol overlapping $N/2$ duration is applied to the output of the sine filter bank – similar to offset quadrature amplitude modulation technique. The basic structure of the filter banks can be also seen in Fig. 1. First the frequency domain data is spread over M subcarriers forming a subband, then it is filtered by a prototype filter of the k^{th} subband $F_k(z)$ which is designed so that it fulfils the Nyquist criteria. In FBMC applications these filter bank structures are implemented in a computationally efficient manner using an N-IFFT and a polyphase network [4]. The filter bank output provides a symbol length of $N * M$ samples. In order not to lose data rate they will overlap with a factor M – due to the Nyquist criteria, the symbols can be separated in the receiver and a perfect reconstruction is possible. For example if $M = 4$ then 4 FBMC symbols overlap. In Fig. 2 the signal structure of FBMC can be seen. The FBMC signal is given for an overlapping factor of $M = 4$. The resulting transmitted signal is the sum of overlapping FBMC symbols generated by the filter banks.

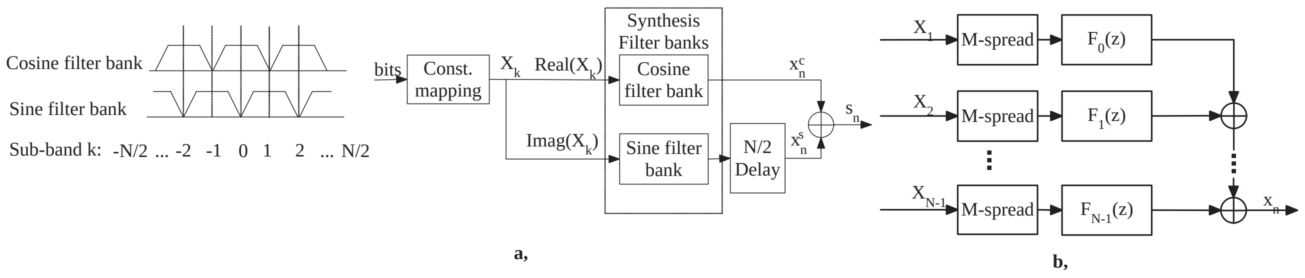
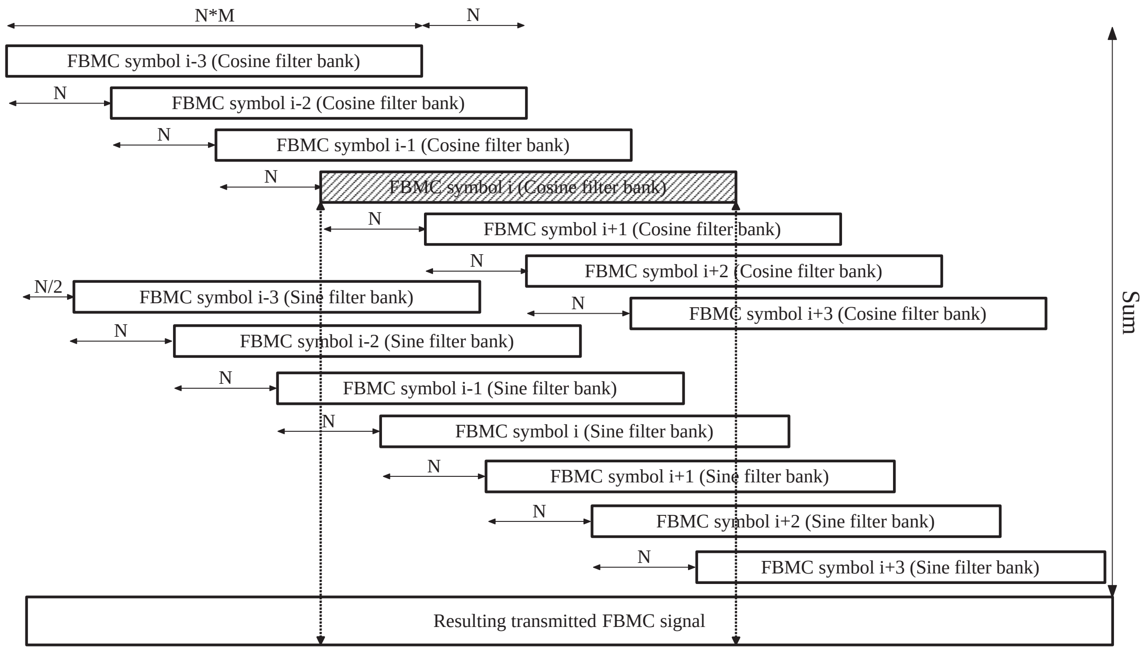


Figure 1: (a) Block diagram of the FBMC transmitter, with the spectral structure of the cosine and sine filterbanks and (b) Basic structure of the filter bank with an oversampling ratio of M.



N*M
N
FBMC symbol i-3 (Cosine filter bank)
FBMC symbol i-2 (Cosine filter bank)
FBMC symbol i-1 (Cosine filter bank)
FBMC symbol i (Cosine filter bank)
FBMC symbol i+1 (Cosine filter bank)
FBMC symbol i+2 (Cosine filter bank)
FBMC symbol i+3 (Cosine filter bank)
N/2
FBMC symbol i-3 (Sine filter bank)
FBMC symbol i-2 (Sine filter bank)
FBMC symbol i-1 (Sine filter bank)
FBMC symbol i (Sine filter bank)
FBMC symbol i+1 (Sine filter bank)
FBMC symbol i+2 (Sine filter bank)
FBMC symbol i+3 (Sine filter bank)
Sum
Resulting transmitted FBMC signal

Figure 2: The structure of the transmitted FBMC signal with an overlapping radio of $M = 4$.

II. Baseband transceiver chain

In this section we describe the applied baseband model for the transceiver chain. The discrete received signal r_n can be expressed as $r_n = s_n * h_n + w_n$, where x_n, h_n and w_n are the samples of the transmitted signal, channel filter and Additive White Gaussian Noise (AWGN) respectively. We will use this model when dealing with the equalization algorithms, where the samples of the channel impulse response are Rayleigh distributed and $\sum_{n=0}^{L-1} |h_n|^2 = 1$ is valid, where L is the length of the channel impulse response. In case of OFDM systems, if the CP is longer than the channel impulse response, after removing the cyclic prefix we can write for an OFDM symbol

$$Y_k = X_k H_k + W_k, \qquad k = 0 \dots N-1, \tag{1}$$

where Y_k is the N-FFT of the r_n, belonging to one OFDM symbol. X_k, H_k and W_k are also an N-FFT of the signal x_n, h_n and w_n respectively. For FBMC systems the frequency domain description is more complicated due to the missing CP leading to ISI from the neighboring symbols. One of the effects of this ISI is that FBMC systems will require different equalization strategies.

III. Channel equalization

A. ZF and MMSE

Zero forcing is known to be the simplest method for channel equalization in the frequency domain. We simply assume that the received noise is zero in equation (1), so the transmitted complex constellation value on the k^{th} subcarrier can be simply calculated as $\hat{X}_k^{\text{ZF}} = Y_k/H_k$. The MMSE technique gives a better result if we have taken the noise also into account. The problem of ZF occurs if H_k is small, the noise values will be also amplified. The equalization coefficient for the k^{th} subcarrier is calculated according to [3] as

$$1/H_k^{\text{MMSE}} = H_k^*/(|H_k|^2 + \frac{N_0}{E_s}), \tag{2}$$

where N_0 is the noise power and E_s is the signal power. It can be seen that with small $\frac{N_0}{E_s}$ values the MMSE solution is equal to the ZF. In case of FMBC system the equation (2) has to be modified according the ISI if we use a per subcarrier equalization scheme similar to [2] as

$$1/\hat{H}_k^{\text{MMSE}} = H_k^*/\left(|H_k|^2 + \frac{N_0 + I}{E_s - I}\right) \tag{3}$$

where I is the power of the ISI, for which we present the following equation: $I = E_s \sum\limits_{n=0}^{L-1} \frac{n}{N}|h_n|^2$. So we can write for the MMSE estimate $\hat{X}_k^{\text{MMSE}} = \frac{Y_k}{\hat{H}_k^{\text{MMSE}}}$.

B. Minimized ISI MMSE

Observing equation for the ISI more closely, we have also concluded that the ISI can be minimized by moving the observation window along all possible positions of the channel impulse response to minimize the following equation

$$\min_{\Delta n}\{I(\Delta n)\} = \min_{\Delta n}\left\{E_s \sum_{n=0}^{L-1} \frac{|n-\Delta n|}{N}|h_{|n-\Delta n|}|^2\right\}, \quad \Delta n = 0 \ldots L-1. \tag{4}$$

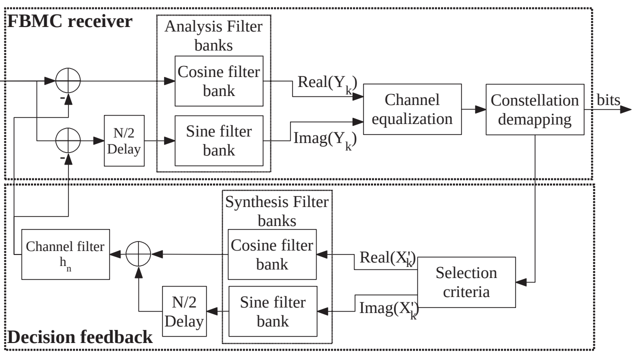
After finding the sample value Δn which minimizes equation (4), the observation window – where we perform the channel equalization – has to be moved by Δn samples and also the channel impulse response has to be circularly shifted, respectively as $\hat{X}_k^{\text{MIN}} = \frac{Y_k(\Delta n_{\min})}{\hat{H}_k^{\text{MMSE}}(\Delta n_{\min})}$.

C. Averaged MMSE

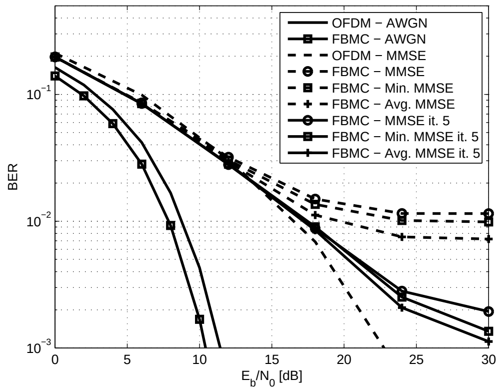
To further minimize the ISI we introduce the idea of the Averaged MMSE equalizer. The averaged MMSE is driven by the idea that the ISI can be also considered as a noise, which can be eliminated by averaging. So based on the idea of moving the observation window, we perform the demodulation and MMSE equalization for each Δn positions of the possible L observation windows and then we average all complex modulation values belonging to the same subband $\hat{X}_k^{\text{AVG}} = \frac{1}{L}\sum\limits_{\Delta n=0}^{L-1} \frac{Y_k(\Delta n)}{\hat{H}_k^{\text{MMSE}}(\Delta n)}$.

D. Iterative decision feedback equalization

In this section we introduce a new iterative decision feedback scheme where the most reliable complex constellation values are fed back after the decision to minimize the ISI in the received signal. This decision feedback scheme shown in Fig. 3(a). The basic idea is to regenerate the ideal transmitted signal, but only those subbands which are reliable. The idea is visualized in Fig. 2: If we want to make a decision for the shaded i^{th} FBMC symbol of the cosine filter bank, then we reconstruct the surrounding symbols $i-3, i-2 \ldots i+3$ which overlap with it (both sine and cosine) based on the selection criteria. Then, during the decision on the i^{th} FBMC symbol the ISI of the known neighboring symbols can be subtracted, reducing the noise steaming from the ISI, leading to better performance.



(a) The FBMC receiver with decision feedback.



(b) BER of the channel equalizers over SNR normalized to E_b/N_0.

The selection criteria is defined based on the constellation diagram, a confidence interval is taken around each constellation point and the complex demodulation values which are inside this interval are considered as reliable.

IV. Simulation results

The simulated bit error rate (BER) for the proposed 3 MMSE equalization schemes can be seen in Fig. 3(b). The applied simulation parameters for both OFDM/FBMC system are the following: $N = 64/64, N_c = 48/48, M = 1/4, CP = 16/0, L = 16/16$. We applied a 16-QAM for modulation. For comparison we have also plotted the results for AWGN channel. It can be observed that OFDM outperforms at higher SNR value (SNR $>$ 12 dB) the FBMC system when only an MMSE equalizer is applied. When introducing the minimized MMSE and the averaged MMSE, a small performance gain becomes apparent for large SNR values for FBMC. The BER results for the iterative decision feedback technique are also depicted in Fig. 3(b). The BER results for the 5. iteration step is given. It can be observed that the averaged MMSE performs the best, the minimized MMSE has a similar result and finally the original MMSE has the worst BER.

V. Conclusion

In this paper we investigated channel equalization schemes for FBMC systems which were compared to the results of CP-OFDM systems using MMSE equalization. Modifications of the MMSE equalization technique suited for FBMC distorted by ISI were presented and iterative decision feedback scheme was introduced, which has a much better performance compared to the simple MMSE methods. The results show that FBMC is a good candidate to compete with OFDM systems over multipath channels.

References

[1] Zs. Kollár and P. Horváth, “Modulation schemes for cognitive radio in white spaces,” *Radioengineering*, 19(4):511–517, Dec. 2010.

[2] T. Ihalainen, T. H. Stitz, M. Rinne, and M. Renfors, “Channel equalization in filter bank based multicarrier modulation for wireless communications,” *EURASIP Journal Appl. Signal Process.*, Jan 2007.

[3] J. van de Beek, . Edfors, M. Sandell, S. Wilson, and P. Borjesson, “On channel estimation in OFDM systems,” *Proc. IEEE Veh. Technol. Conf.*, 2:815–819, July 1995.

[4] P. P. Vaidyanathan, *Multirate systems and filter banks*, Prentice-Hall Inc., 1993.

SIMULATION OF ERRORS IN FLOATING-POINT CALCULATIONS

Vilmos PÁLFI
Advisor: István KOLLÁR

I. Introduction

In digital signal processing, quantization errors have two main sources: A/D conversion, and roundoff in the computer, usually a digital signal processor. Therefore, to properly characterize the data processing chain, both need to be investigated. The usual problem is, however, that analytic results are rare. Most of these are of statistical nature, and they are based on the noise model of roundoff: for fixed-point, the roundoff is noise-like, with variance $LSB^2/12$, it is independent of the signal, and it is usually white. Floating-point error is less regular, but also has general approximation rules: its variance is about $\mathrm{var}\{\nu\} \approx 0.180 \cdot 2^{p} \cdot \mathrm{var}\{x\}$, with p the precision and x the signal ([1], page 257, Eq. (12.24)), and it is uncorrelated with the signal, and it is white. These properties are usually all valid, but careful experimentation is needed because they are usually only approximations with no guarantee. However, experimentation is not easy: tools are often not available, and if available, they need to be cross-checked because even small deviations from the theory can cause noticeable errors.

II. Possibilities for simulation

To simulate single-precision floating-point number representation, the following six methods were used:

- Calculations using variables of class “single” in Matlab,
- Using a roundoff simulation toolbox in Matlab,
- Using an object-based finite precision toolbox in Matlab,
- Using a DSP simulator,
- Using a DSP itself for calculations,
- Using variables of class “float” in C++ language.

All six have been tried, and achieved by proper settings that all of them yield exactly the same results. This will be discussed below. During the calculations the use of an extended-precision accumulator was avoided because one of the main aims was to achieve exactly the same simulation results.

III. Simulation results

We will look beyond the noise model: investigate roundoff errors which deviate from noise. The examinations are built up of simulations. The roundoff error of the Fast Fourier transform is analyzed for a clean sine wave input, because it has a rather regular pattern. According to [3] in this case for fixed-point the roundoff error is expectable to deviate from noise, so we investigated its behavior also for floating-point arithmetic.

The pattern of the error was checked for almost coherently sampled sine wave. This means that the sampling frequency was slightly shifted from coherence. Figure 1 shows the error on 125 different frequency shift values.

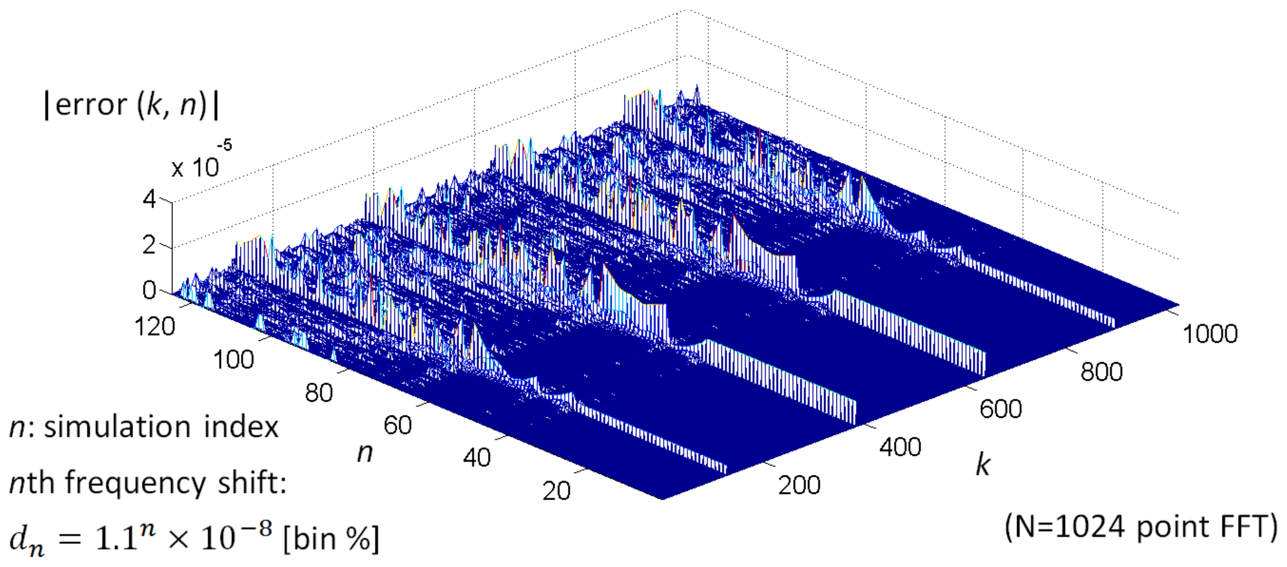


Figure 1. Error's pattern on different frequency shift values.

Thanks to a very useful feature of Roundoff toolbox in Matlab every rounding operation (for example after a multiplication or addition) can be replaced by adding independent white noise to the exact value. This way our results can be compared with results assuming the noise model.

In the following calculation the noise model of roundoff is used. Let us find the value X for which the probability is P=99.9% that every peak of the roundoff error is smaller than X. This X value can be seen in Fig. 2 as the horizontal line. In a very unfortunate situation 15 dB loss in the spurious-free dynamic range can be experienced when compared to the noise model. This means that the noise model of roundoff is not acceptable in this special case.

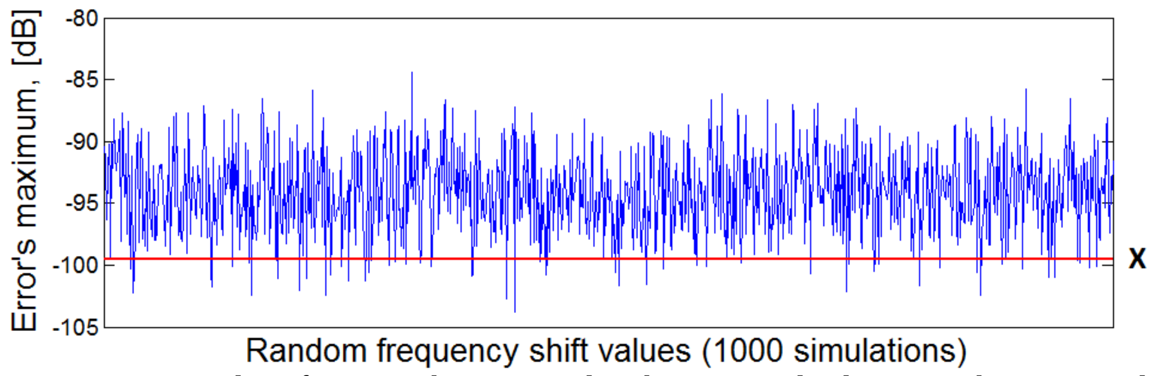


Figure 2. Maximum value of error peaks compared to the estimated value using the noise model.

IV. References

[1] B. Widrow and I. Kollár (2008), "Quantization Noise: Roundoff Error in Digital Computation, Signal Processing, Control, and Communications," Cambridge University Press, Cambridge, UK. Home page: http://www.mit.bme.hu/books/quantization/

[2] Weinstein, C. J., "Roundoff Noise in Floating Point Fast Fourier Transform Computation," IEEE Transactions on Audio and Electroacoustics, Vol. 17, No. 3 ,pp. 209-15, 1969

[3] Pálfi Vilmos, Kollár István, Roundoff Errors in Fixed-Point FFT. In: WISP'2009, IEEE International symposium on Intelligent Signal Processing. Budapest, Hungary, 2009.08.26-2009.08.28. (IEEE) pp. 87-91. Paper W1B5. http://mycite.omikk.bme.hu/doc/74336.pdf

BIAS OF MODEL FITTING BY LEAST SQUARES ESTIMATION

Péter Zoltán CSURCSIA
Advisor: István KOLLÁR

I. Introduction

The aim of this short paper is to present the non-trivial problems of the most used 'best fitting' parameter estimation: the Least Squares (LS) method.

Let us get a short summary of ordinary LS. First, to make it obvious, look at Fig. 1.

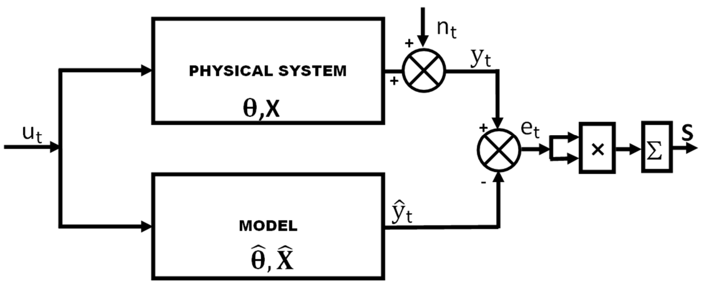


Figure 1: Measuring the error signal and the cost function between physical system and model

Our physical system (or process) is observed and represented by regressand variable y_t as a scalar measured signal, where t=1,2,..,N and N is the number of samples. In most cases the index t represents the time instant of the measurement.

The regressors, all known variables, for instance the data of input signal are in the p-size vector ($\mathbf{x_t}=[x_{t1}\ x_{t2}\ \ldots\ x_{tp}]^{\mathbf{T}}$). The model has one unknown coefficient θ_i per explanatory variable in a p-size vector $\boldsymbol{\theta}$ ($\boldsymbol{\theta}=[\theta_1\ \theta_2\ \ldots\ \theta_p]^{\mathbf{T}}$) and it is important that the parameters must be linear in $\boldsymbol{\theta}$**.** In the model we use the following estimation for y_t:

$$y_t \sim \hat{y}_t = \hat{\theta}^T \mathbf{x_t} \tag{1}$$

To measure the observation error e_t (noise) i.e. difference of $(y_t - \hat{y}_t)$(see the Fig. 1) we can create the (Euclidean) distance or cost function called S:

$$S = \sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{2}$$

To use matrix algebra, we have to collect all the measured samples y_t into an N-vector **Y**, all the vectors $\mathbf{x_t}$ into an N×p matrix **X**. Let us put Eq. (1) in matrix-from into Eq. (2), then we have to find the minimum value of S. Using the gradient of S we can find the minimum value of $\boldsymbol{\theta}$:

$$\hat{\theta}_{LS} = \arg\min_{\theta} S = [X^T X]^{-1} X^T Y = X^+ Y\,. \tag{3}$$

The $\hat{\theta}_{LS}$ is called the ordinary least squares estimate of $\boldsymbol{\theta}$. One of the conditions of the existence of this exact mathematical solution is that the inverse of $[\mathbf{X}^{\mathbf{T}}\mathbf{X}]$ must exist or in other words, the columns of **X** must be linearly independent. The $\mathbf{X}^+$ is called generalized pseudoinverse of Moore-Penrose.

II. Problems

Up to this point we have assumed that our system or model does not have any feedback and noise-effect, namely if we use LS in the general case we can find the best parameter estimation simply.

However, in a case of dynamical system we have to make allowance for feedback and inherently, the modified equation.

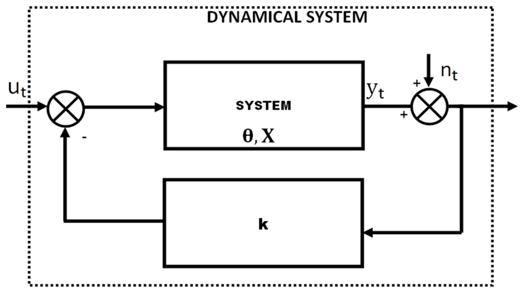


Figure 2: A dynamical system

The modified N×p matrix **X** that represents the known variables by vectors after Fig. 2. is the following (of course it is only a simple example in case of strong feedback):

$$X = \begin{bmatrix} \cdots & \cdots \\ y_t & u_t \\ \cdots & \cdots \end{bmatrix} \approx \begin{bmatrix} \cdots & \cdots \\ y_t & -ky_t \\ \cdots & \cdots \end{bmatrix}. \tag{4}$$

After Eq. (4) we have to realize that our matrix **X** cross-correlated between regressors, i.e. the stronger the feedback, the more correlated regressors, which can cause problems with linear independence.

Another problem is the effect of (observation) noise n_t,(as N-size vector in **n**), provided it is un-correlated, zero-mean, the LS estimate is therefore unbiased (for instance it can be an AR-process).It is the best and the regressors are deterministic and contain constant component.

When **X** is partly or wholly random, the bias has to be averaged over X as well as e and becomes for bias:

$$\text{bias} = \underset{U,n}{E}\{[X^T X]^{-1} X^T n\} \tag{5}$$

For example let us consider a first-order discrete dynamical system. In Eq. (5) $v_t=n_t+a_1n_{t-1}$ will represent the effect of zero-mean, uncorrelated noise in time t and its constant variance σ^2. In this case for instance we can have the following equation for y_t:

$$y_t = -ay_{t-1} + bu_{t-1} + v_t. \tag{6}$$

In Eq. (5) we can notice that the regressor y_t is correlated with v_t through n_{t-1}. If the input **u** is independent of **n**, we can get the following expected value:

$$E\{y_{t-1}v_t\} = a_1\sigma^2. \tag{7}$$

It means that our first-order system has bias and it leads to some problems in modeling.

III. Summary

There are a lot of advantageous properties of using LS in statistics and model fitting e.g. it is very simple to use, from the linear estimators it is the best unbiased.

In the most common case we do not have any problems with the calculation, nevertheless we have to consider problematic cases as well for instance:

- different noise-effects through the feedback,
- non linearity in **θ**,
- ARMA-systems, etc.

References

[1] J. P. Norton, An Introduction to Identification, Academic Press, London, ISBN 0-12-521730-7, 1986
[2] C. W. Helstrom, Probability and Stochastic Processes for Engineers, Macmillan Coll Div., 1991

HEART SHADOW COMPENSATION

Sándor I. JUHÁSZ
Advisor: Gábor HORVÁTH

I. Introduction

Chest radiographs can help the detection of lung cancers, tuberculosis (TB) and other lung diseases. The early detection of cancer is very important as it significantly increases the chance to cure the disease. Global screening is a relatively cheap way to detect lung cancers and TB. Such screening programs generate a vast amount of pictures to be analyzed by experts. This is where computer-aided detection (CAD) can help the work of radiologists.

CAD systems are not reliable enough to do the work alone at the moment. The current goal is only to create a system that can help the detection and increase the accuracy of examination by searching suspicious areas (region of interest, ROI) or by enhancing the visibility of the picture at the darker regions.

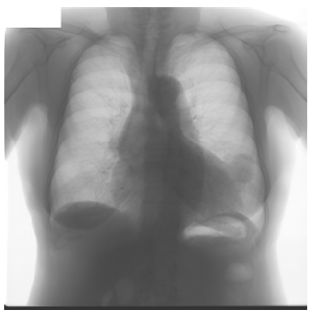


Figure 1.a

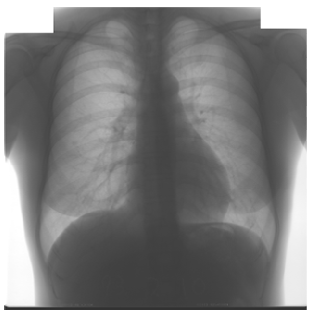


Figure 1.b

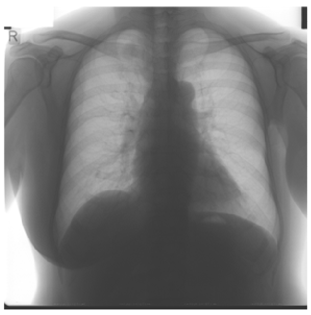


Figure 1.c

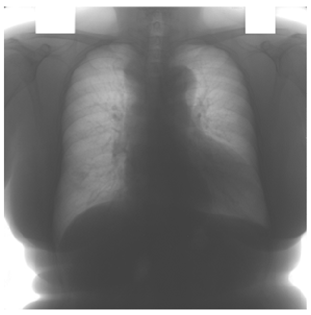


Figure 1.d

Figure 1: Examples of different heart shadows

The first step in such an evaluation would be the delineation of the organs' boundaries. Searching the contour of the lung, the heart, the clavicles and the ribs determine the area of processing and the raw data for image enhancing. The lung contour has diagnostic value without further processing too as it can show cardiomegaly and pneumothorax.

The next step is image enhancing. It contains techniques to make the image less noisy, to correct the brightness of the image and to reduce the shadows of other organs. Promising results have been reached at the field of rib and clavicle shadow compensation [4]. Heart shadow compensation has not been solved yet.

Finally the detection of lung diseases gives regions that doctors should closely check. Nodule detection and texture analysis are used at this stage. Nodule detection can profit from image enhancing [3]. Nodules can be seen better without the ribs and the false positive number can be decreased as the intersection of ribs will disappear.

Figure 1 gives a few examples of chest radiographs from the JSRT database [2]. Heart shadows show great variety in shape, size, position and darkness. The contours also vary from sharp to diffuse. Image 1.a is very difficult for several reasons: there is a nodule next to the heart in the left lung, a potential threat for contour detection; air bubbles are visible under the diaphragm making its border less prominent; the heart shadow is bigger than average and the shadow of the breasts adds another shadow layer with new edges. Image 1.b shows the typical heart shape and size. Image 1.c is an example where the border of the heart is not a strong edge rather a diffuse continuation into the area of the lung. Image 1.d shows an above average sized heart shadow.

The heart shadow is usually dark enough to make the detection of lung nodules in the same area almost impossible, thus heart shadow compensation can help both nodule detection algorithms and doctors. The goal is to make the area of the dark region more visible without destroying its inner structure. We must avoid creating artificial shadows and keep the anatomical structure.

II. Delineation of the heart

Shadow compensation algorithms usually need the contour of the object as an input. The precision of the contour is very important because if the calculated contour goes inside or outside the real contour then the compensation will create artifacts: darker and brighter structures along the border of the heart. This will create false positive hits in nodule detection or reduce the area that can be analyzed reliably. Figure 2 shows an example of such areas. The dotted line is a bad heart contour segment. After the shadow compensation new structures will appear on the image: a lighter than the average background area where the calculated contour was too big and a darker area where it was too small. Note the pecked line that is the border of the aorta descendens. We should determine that boundary too because the aorta descendens has a strong additional shadow inside the heart shadow. For the shadow compensation the dark area around the heart is used. This is not the exact anatomical border of the heart. The goal of the compensation is to make the lung area more visible everywhere where it is shadowed by something.

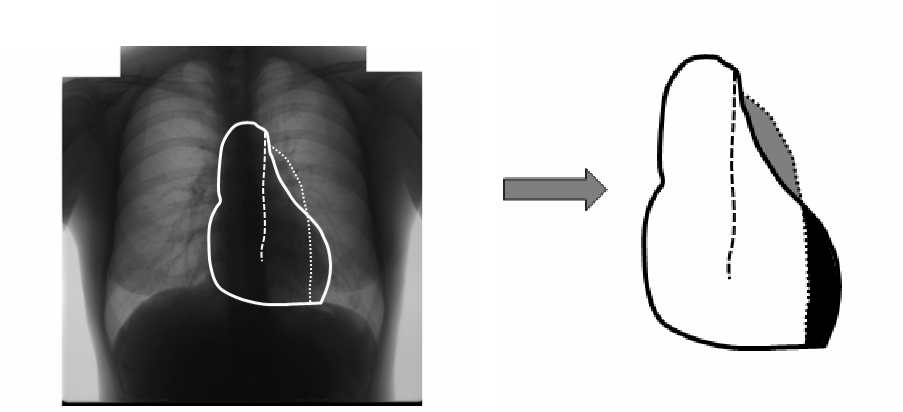


Figure 2: Effect of bad contour after the shadow compensation

A Active Shape Model (ASM)

ASM was used successfully for lung contour detection earlier. It can determine the heart boundaries the same way. It is able to generate complex contours, not only a closed loop, so it can be used to simultaneously determine the inner contours of the heart, the heart boundary and the lung boundaries as well. The algorithm is very fast. The only drawbacks are that its result is usually not accurate enough for the shadow elimination and it needs a good initial contour to guarantee that the algorithm converges to the right contour. See [1] and [4] for details.

B Active Appearance Model (AAM)

The AAM algorithm not only determines the contour of the object but generates the inner texture too. This type of combined search can lead to more robust solutions. Unfortunately the price of robustness is the relative slowness of the algorithm and the contours are far from the subpixel accuracy. See [1] for a comparative study of several contour detecting algorithms.

C Multistep Algorithms

For precise contour detection the examples above are not enough. We have to make sure that the algorithm is robust and accurate at the same time. A multistep method is used to guarantee this:

- First we calculate the approximate boundary box of the heart. This is useful because the position and size of the heart have great variability. The boundary box can be acquired using the position of the diaphragm and the result of an ASM for the lung contour detection which is quite reliable.
- After we have a good initial position we can use ASM to determine the heart boundary with little error.
- A final step is used to make the boundaries as close to the real contours of the shadow as possible [5]. We build a database of the contour's normal vectors' directions. The average direction of each contour normal vector is calculated using a training set of images. Edges are searched in the neighborhood of each contour point along the direction of the average normal vector. Within a range we move the contour to the strongest edge found. Outlier contour points are replaced using interpolation.

III. Heart shadow compensation techniques

Heart shadow compensation can never be perfect as we don't know the fine structure of the heart and in some cases the shadow is too dark, only noise is left after the compensation. But a relatively good image intensity correction can be made in most cases.

A Compensation using stripes

A simple model of the heart shadow intensity can be made if we assume that the intensity at a location is the function of only the distance from the heart contour. This is approximately true for many images. We only have to determine this intensity for every stripe. Figure 3 shows how we divide the image into regions. We generate stripes parallel to the left side of the heart's contour. A polynomial fit is made to the intensity of the pixels along curves parallel to the ribs outside the heart contour. These functions are used for extrapolation to estimate the intensity under the heart. The difference of this estimation and the real intensity is created by the heart's shadow. The average intensity is calculated in each stripe and the stripes' intensities are multiplied to the necessary level. The modifications are blurred to avoid strong stripes in the resulting image.

This technique is useful only for partial compensation. It can only be used on the left side of the heart shadow to the left of the mediastinum.

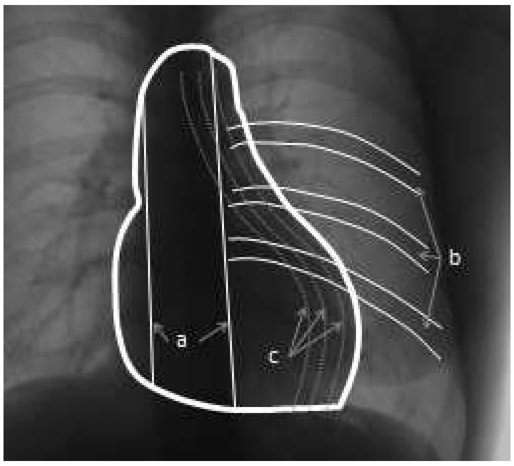


Figure 3: Areas used during the compensation. a shows the mediastinum boundaries, b indicates the ribs and c points to equidistant contours

B Compensation using models with global parameters

The previous model had many parameters. Each stripe had a unique average intensity that had to be calculated. A different approach is to use fewer global parameters inside the heart shadow. We can use the distance of the heart contour and the distance of a certain rib as coordinates. 1D and 2D polynomials can be fit to this intensity surface. The parameter fitting can be made with well-known algorithms.

The advantage of this kind of fitting is that there won't be artificial jumps in the corrected image, the polynomial guarantees the smoothness.

Conclusion and future work

The frameworks of heart compensation algorithms were introduced. In the future these will be optimized using a large image database.

References

[1] B. van Ginneken, M. B. Stegmann and M. Loog, "Segmentation of Anatomical Structures in Chest Radiographs using Supervised Methods: A Comparative Study on a Public Database," Medical Image Analysis, 10(1):19-40, 2006.

[2] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera, K. Doi, "Development of a digital image database for chest radiographs with and without a lung nodule: receiver operating characteristic analysis of radiologists' detection of pulmonary nodules," American Journal of Roentgenology 175, 71-74, 2000.

[3] G. Orbán, Á. Horváth, and G. Horváth, "Lung Nodule Detection on Rib Eliminated 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. 363–366. Springer Berlin Heidelberg, 2010, 10.1007/978-3-642-13039-7 91.

[4] 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.

[5] K. Katus, "Szívárnyék körülhatárolása mellkasröntgen felvételeken," BSc thesis, G. Horváth (advisor), Budapest University of Technology and Economics, 2010.

ARTIFICIAL NEURAL NETWORK BASED LOCAL MOTION PLANNING OF A WHEELED MOBILE ROBOT

István ENGEDY
Advisor: Gábor HORVÁTH

I. Introduction

The navigation system of a mobile robot must handle numerous well separable subtasks for proper operation. These tasks are the localization, motion planning and mapping.

In the motion planning problem the tasks are to determine the path from one point to another one (called target), and to control the motion, to make the robot follow the previously computed path. Thus motion planning is often discussed as two separate subtasks, the path planning and the movement.

A Path Planning

Path planning is the procedure which plans the path from the current location to the end point, of which the robot will have to go through. Obviously, the map of the intermediate area is needed to be known, so the obstacles and unreachable areas could be avoided. In case of metric maps usually the discretization of the area is needed, to be able to use graph-based algorithms in path planning [1]. One commonly used algorithm that works on finite graphs is the well-known A* algorithm [2].

There are also soft-computing methods for path planning on metric maps; some of them use artificial neural networks (ANN-s). They are used for a couple of reasons, like they are computationally efficient, they do not require an exact mathematical description of the problem, they better tolerate noisy input data, and they can adapt to changing environments.

ANN-s are usually trained with analytical or unsupervised training algorithms for path planning. This includes Hopfield-network in [3], self organizing map (SOM) and dynamic wave expansion neural network (DWENN) in [4]. However there are supervised learning solutions as well, like this one.

B Movement

The movement is the procedure of keeping the robot on the path. This could be complex, because the movement of the robot must meet various physical constraints. The most common solution is to use various controllers, such as P, PD or PID controllers.

The movement control could be carried out using soft computing methods, including ANN-s. Some of these methods are able to simultaneously perform the path planning and the movement problems too [2].

II. Former Results

In our former work [5], we have shown a solution of the mobile robot motion planning problem. The robot was a nonholonomic car-like robot. The target of the robot was a predefined position and orientation in the working area. There were also static and moving obstacles that the robot had to have to avoid. We presented an ANN based mobile robot controller for obstacle avoidance. The ANN was trained with the classical backpropagation through time (BPTT) training approach. We showed that additional constraints can be taken into consideration through regularization, and how this is used for avoiding obstacles. It was also shown that real time online training is also possible.

Although the online training is a good solution to adapt to changing environment, it has also a couple of drawbacks. The most serious one is the increased need of computational power. Another problem with it, that it is not capable of remembering the already learned movements, if a new movement in a new situation is trained.

There was another problem in our former work: the description of the configuration space (coordinates of the robot and target) were relative to an absolute coordinate system. This way the ANN had to learn the same movements in different positions.

III. Modified Navigation Method

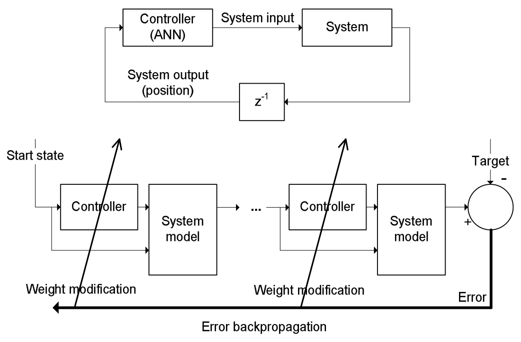
In the modified navigation method the controller ANN is still trained with the BPTT method, the output is the same as it was before, but the input is slightly different. Now the state of the robot is described with the relative position and orientation of the target to the robot. The description of the state is also changed to a polar coordinate system description. The input of the ANN is extended with a polar occupancy grid map of the obstacles in front of the robot.

A The BPTT method and obstacle avoidance

The BPTT could be used for robot navigation, as D. Nguyen and B. Widrow showed it [6], “Fig. 1.a” where the controller of the robot is an artificial neural network and the whole control loop is opened during the training. In this situation not only ANNs take place in the chain, but numerous models of the system, which are not needed to be trained, but the error must be propagated back through them.

To make the BPTT method able to take account of the obstacles, a potential function can be defined based on the location and the size of the obstacles. To actually repel the robot away from the obstacles, this function must be used to extend the cost function of the Delta-rule. The goal of the weight modification is not only to minimize the error at the end of the simulation chain, but also to minimize the potential of the path, to get the robot the farthest from the obstacles.

The regularization term could be added to the error during the backpropagation at the appropriate places through the simulation chain, and the weights of the ANN can be modified using the usual way as it can be seen in “Fig. 1.b”.



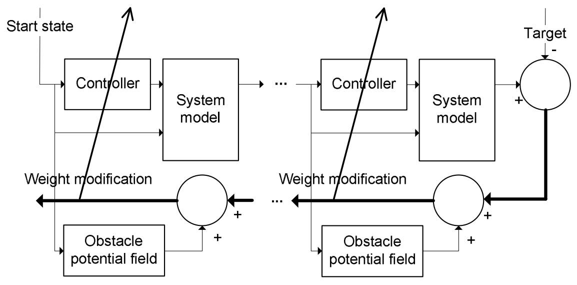


Figure 1. a) (left) BPTT in use of training in a control loop, b) (right) Adding the obstacle potential to the error

B Robot state and obstacle description

The state of the robot is described with its relative position to the target in a polar coordinate system. As it is shown in “Fig. 2.a” this state description contains the following values: α, the angle between the direction the robot is facing and the direction of the line pointing to the target from the robot, d, the distance of the robot and the target, and β, the angle between the direction the robot is facing and the target orientation.

Describing the robot state this way, the target of the training, where we wish the neural network controller to guide the robot, is the origin of the coordinate system. So we want to train the neural network that at the end of the unfolded chain, the state of the robot will be zero in all three values. This way numerically the state is the error we want to minimize.

The obstacles, similar to the target, are described in the polar coordinate system of the robot. Each obstacle has two values, its distance to the robot (*b*), and the angle between the direction of the robot and the line pointing from the robot to the obstacle (*β*). In "Fig. 2.b" it is shown how these coordinates can be converted into the distance and angle between the target and the obstacle, which is needed in the backpropagation part of the BPTT training.

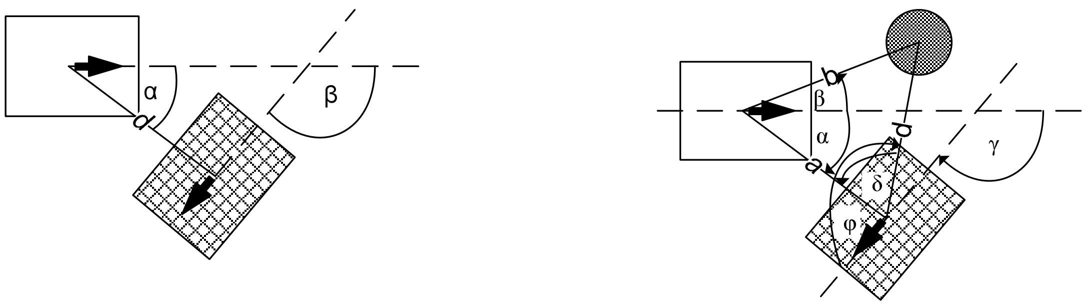


Figure 2. a) (left) State of the robot (white rectangle) in polar coordinate system. b) (right) The obstacle (dotted circle) and its relative position to the robot and the target (checked rectangle)

C The extension of the neural network input

To solve the problems of the online training we have proposed the following solution. The different movement patterns in the different situations could be stored in the controller ANN. To do that, the different situations must be distinguished from each other at the input of the ANN. This way an offline trained ANN could be used as a controller.

To make the ANN aware of the obstacles, we have extended its input with a polar grid "Fig. 3.", where each cell in the grid is an input of the ANN. If there is an obstacle in the grid cell, its value will be 1, if the cell is empty, the value will be 0. The grid is only in front of the robot, since it is moving only forward, and we do not need to know what is behind the robot. The movement only depends on the direction and distance of the target, and the obstacles in front of the robot.

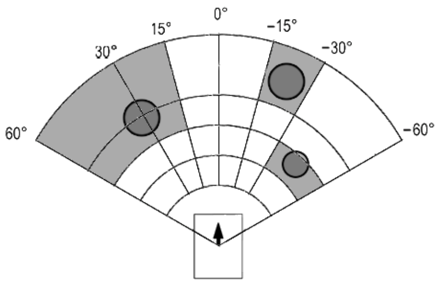


Figure 3. The polar grid in front of the robot. The gray cells are the occupied cells, in which there is at least 20% of an obstacle, the dark circles are the obstacles, and the rectangle is the robot.

With the use of this polar grid, the ANN is able to learn and remember the movement patterns for each situation in which it was trained with the regularized BPTT algorithm. This way it can be used in offline training mode too.

The offline training needs a set of problems that is going to be solved. We used randomly generated situations for the robot. Since the robot state is described with the relative distance and angle of the target, these values can be changed randomly. The angles could take any value from [-π; π), and we restricted the robot–target distance to the interval of [200 mm; 3000 mm]. For comparison, the shaft length of the robot was 120 mm. We used 1–3 obstacles, which were placed randomly between the robot and the target. For each situation, we trained the ANN only for one BPTT step, to prevent overfitting to any situation. With these randomly selected situations and one

BPTT training step per situation, we achieved, that the network was trained to solve these kinds of situations in general.

With the offline training only a little, local part of the environment is taken into account, on which it is not possible to compute a globally optimal path. This offline method is useful when the navigation system have any information only on the local surroundings of the robot. If a bigger part or the whole map of the environment is known, other path planning algorithms can be used to compute a globally optimal path. Using our method with waypoints on the path, the robot would be able to follow it, without colliding with any obstacles.

IV. Simulation Results

The method was tested in simulation only. The former simulator application was modified with the previously mentioned changes. The tests showed that the polar coordinate description and robot centric approach is more suitable for this problem. The convergence of the training was considerably faster than it was with the Descartes-coordinate system.

We have also tested its capabilities with offline training in a random generated situation. The robot was able to get to the target without collision. Since it was not aware of all of the obstacles and their exact location, its path was not optimal by any means, but we haven’t expected that.

V. Conclusion

In this paper we have shown that the classical BPTT training approach can be extended using regularization, to solve the motion planning problem among obstacles, and this could be done effectively, if the state of the robot and its environment is described using a polar coordinate system.

It was also shown that this is also possible using offline training, by extending the inputs of the neural network with a polar occupancy grid of the obstacles. This offline trained ANN is able to navigate the robot on a collision free course towards the target.

We have shown that there are some limitations on the capabilities of the method: it is not guaranteed that the path of the robot will be optimal by any means, or that the robot will reach the target and it won’t get stuck in a local minimum.

Acknowledgment

The authors gratefully acknowledge the support of the Hungarian Fund for Scientific Research (OTKA), Grant #73496

This work is connected to the scientific program of the "Development of quality-oriented and cooperative R+D+I strategy and functional model at BME" project. This project is supported by the New Hungary Development Plan (Project ID: TÁMOP-4.2.1/B-09/1/KMR-2010-0002).

References

[1] Sebastian Thrun, Wolfram Burgard, Dieter Fox: Probabilistic Robotics, MIT Press 2005

[2] Stuart Russel, Peter Norwig: Artificial Intelligence A Modern Approach, Prentice Hall 2003

[3] R. Glasius, A. Komoda and S. A. M. Gielen, Neural Network Dynamics for Path Planning and Obstacle Avoidance, Neural Networks, 8(1), 1995, 125-133.

[4] D.V. Lebedev, J.J. Steil, and H. Ritter, “A neural network model that calculates dynamic distance transform for path planning and exploration in a changing environment,” in Proc. IEEE Int. Conf. on Robotics and Automation, 2003, pp. 4201-4214.

[5] I. Engedy and G. Horváth, “Artificial Neural Network based Mobile Robot Navigation,” in Proc. of the IEEE International Symposium on Intelligent Signal Processing, pp. 241-246, Budapest, Hungary, Aug. 2009

[6] D. Nguyen and B. Widrow, ``The Truck Backer-Upper: An Example of Self-Learning in Neural Networks," Proceedings of the International Joint Conference on Neural Networks, 2:357-362, 1989.

A HYBRID LUNG NODULE DETECTION SCHEME ON CHEST X-RAY IMAGES

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[1], 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. Although this is true from the performance point of view, we observed that radiologists lose their faith in the system and ignore its results if it fails to detect some obvious cases. This remains true even if we explicitly specify what type of nodules the system searches for.

To overcome this issue, the current study focuses on enhancing the capabilities of an existing CADe scheme to find special types of nodules that were missed previously, but were frequently found by the radiologists. Therefore we introduce the Large Nodule Filter (LNF) targeting nodules larger than 30 mm diameter, having high contrast and usually overlapped by large structures for example the shadow of the heart or the spine. Finding these nodules have little utility in everyday screening, but may help the acceptance of CADe amongst radiologists. As a side effect we developed a framework that enables the simultaneous use of many nodule enhancing algorithms and the efficient integration of their results. This may help us in the near future to integrate more algorithms that complement our current algorithms by finding more subtle cases.

II. Materials and Methods

For our solution we used the following three-step scheme. The first step described in [2] segments the viewable area of the lung and frees the image from unnecessary objects and noise, thus making the nodule more visible. The next two steps can be seen in Figure 1. Nodule enhancement highlights round shaped objects like the target lung nodules by using image processing algorithms. After normalization and resizing the processing splits up based on the targeted nodule type. The Constrained Sliding Band Filter (CSBF) is used for the enhancement of smaller and subtle nodules and the recently developed LNF for large nodules with high contrast. The candidates collection algorithm is the same for the two threads. 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

[1]Sensitivity is the fraction of correctly diagnosed positive cases and all positive cases.

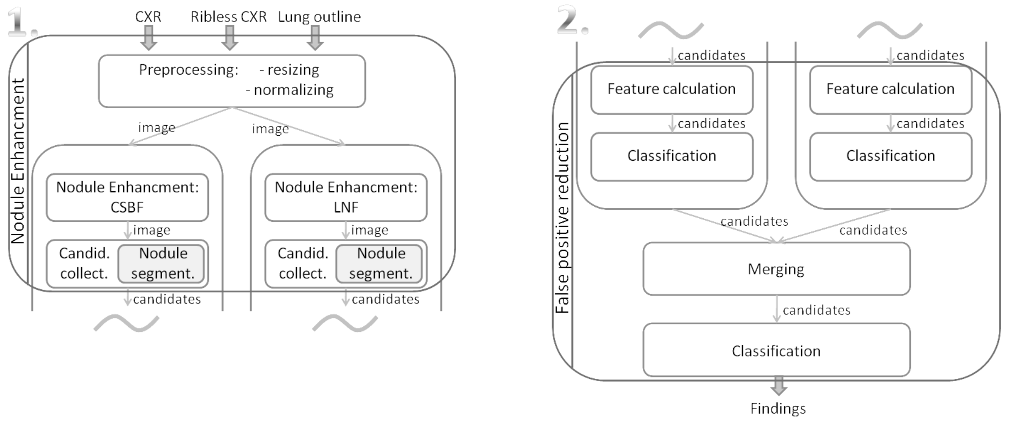


Figure 1: The nodule detection process. Arrow labels reveal the type of data flow.

classifier. Classification is done for each thread followed by a merging step that eliminates duplicate and highly overlapping results. A final classification is carried out on the merged set of candidates.

A. Nodule Enhancement

According to our observations different types of nodules may need completely different algorithms for efficient enhancement as they have different characteristic properties. A smaller nodule in the early stage is usually dimmer but has an approximately circular shape, thus the convergence of the gradient vectors can serve as an important clue. In large scales this can be computationally intensive, furthermore larger structures and intensity variations can alter gradient directions. On the other hand large nodules tend to have high contrast that can help detection. Hereinafter we will introduce the new LNF algorithm specialised for large nodules as the CSBF filter targeting smaller ones has been described in detail previously [3].

The LNF aims to enhance nodules with diameter between 30 mm and 75 mm and high contrast but allows them to lie almost completely outside the viewable lung. The basic idea behind the algorithm is a modified Local Contrast Enhancement (LCE) followed by a Top-hat filter. The LCE output is

$$G(x,y) = \frac{1}{1-\exp-(F(x,y)-\frac{1}{|R(x,y)|}\sum_{(u,v)\in R(x,y)}F(u,v))}, \tag{1}$$

$$R(x,y) = \begin{cases} \{(u,v)|(u-x)^2+(u-y)^2<2r^2\}\cap L & (x,y)\in L \\ \{(u,v)|(u-x)^2+(u-y)^2<2r^2\}\cap /L & \text{otherwise} \end{cases}, \tag{2}$$

where F is the original image, L is the viewable lung and r is the radius of the targeted nodule. The trimming of R with L ensures we have a homogeneous area completely inside or outside the lung. The rationale behind the logistic function is to get a result in between local normalization and local thresholding as it can be seen in Figure 2 (2nd).

Top-hat filtering is a simple convolution by a cylinder shaped kernel with radius r. The side of the cylinder is slightly tilted to allow circular shapes with little distortion or a small deviation of the nodule radius. This method besides nodules would also enhance other dark structures like remainders of rib shadows or areas filled with vessels like the mediastinum. To suppress these areas the filter output is weighted with the smoothness of the area. For smoothness the standard deviation of smoothed nodule pixels inside the viewable lung is calculated. An example is shown in Figure 2 (3rd). The method uses a multi-scale framework to detect different sized nodules.

To find nodules lying outside the viewable lung the Top-hat filter is run for the entire image. This would enhance structures like the vertebra so the filter output is kept only where both negative and positive parts of the cylinder overlap with the viewable lung, and the area of intersection for both is greater than 15% of the filter part area. As a post processing step the areas where the predicted nodule would lie outside the whole lung are suppressed. This requires an overestimation of the whole lung area, for which we use the following algorithm. The binary masks of the left and right viewable lung parts are dilated towards the centre and then eroded with the estimated nodule radius. Then the union is taken for the two parts. The resulting mask is consistent with our assumption that nodules can reside under the shadow of the heart, aortic arth or hemidiaphragm but cannot hang out towards the side of the body. An example is shown in Figure 2 (4th and 5th).

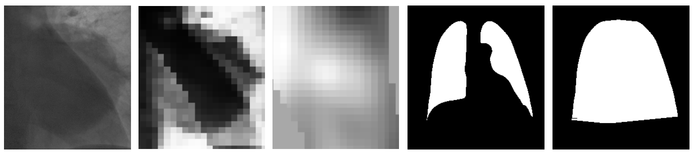


Figure 2: From left to right: a large nodule partly overlapped by the heart (1st), the LCE output (2nd), the final LNF output (3rd), the masks of the viewable lung area (4th) and the estimated area of the whole lung (5th).

B. Reduction of false positives

The candidate collector finds approximately 20-30 candidates on the enhanced image. For the reduction of false positive findings we use a Support Vector Machine (SVM). The training data comes from validated findings of radiologists. The most crucial design parameter of the SVM is the kernel function for which we use the isotropic Gaussian kernel. The input vector of the kernel consists of various features describing texture, geometry and location.

To eliminate irrelevant features, we ran a simple feature selection algorithm which observes the performance of the SVM while adding various features in a forward selection manner. As the relevant features turned out to be different for the output of the two nodule enhancing algorithms we decided to use separate classifier for each result set. Using one classifier requires the union of relevant features which would increase the number of dimensions reducing overall performance. Furthermore, tests showed that multiple classifiers can save run time.

The SVM with isotropic Gaussian kernel requires two hyperparameters to be chosen by the user. For this we use a gradient search with some modifications to make it robust against local maxima. For performance measurement cross validation is utilized.

III. Results

We tested the system on a private chest X-ray database containing images of 243 patients where 93 of the cases contained at least one malignant lung nodule. Nodule diameter ranged from 2 mm to 98 mm, the average was 24 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.

As a testing method we used 4-fold cross validation with 100 iterations for each setup to reduce the variance introduced by the random permutation of images. The results can be seen on a free-response receiver operating characteristic (FROC) curve in Figure 3. The plot shows the fraction of malignant cases that can be found as a function of average number of false positives produced for each image. We ran the FROC analysis for a system using the CSBF or the LNF enhancers only and for the complete version using both algorithms. The poor performance of the LNF on its own is clear as the radiographs contained mostly small nodules; however integrating its results with the CSBF improves sensitivity without adding many false positives. The increase compared to the standard CSBF solution is obvious. At constant 70% sensitivity the number of false positives can be reduced from 2.3 to 1.7 per image.

Alternatively with a false positive rate of 3 the sensitivity can be increased from 73% to 77%.

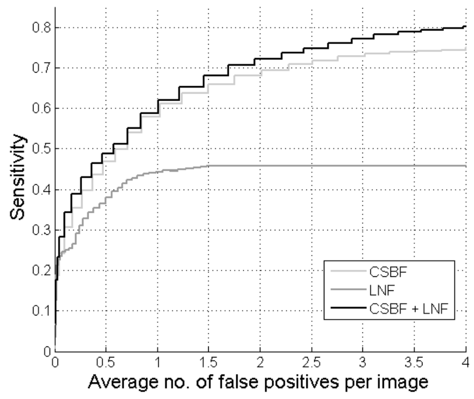


Figure 3: Comparison of systems using only CSBF or LNF and the complete (CSBF + LNF) version.

The final results are still good if we consider everyday applicability. 72% sensitivity with 2 false positives or 77% with 3 requires acceptable extra work from the examiner while it marks malignant areas the examiner may overlook otherwise. Of course the number of false positives should be reduced to further improve the usability of the system. However, the representativity of the used image database is not yet examined thus comparison of the results with other systems should be made carefully. Although we cannot be sure that system performance remains the same in everyday practice, we are optimistic as an in progress clinical study reported similar results based on the first 800 cases.

IV. Conclusion

In the current work we developed a new filter able to find larger nodules and integrated it to our existing CADe scheme. The main contribution to the CADe community is twofold. First, the introduced LNF can be a useful tool if the automated detection of mature tumours is needed. Second, the current case shows that a hybrid system involving specialized filters and proper synthesis can be more efficient than a system using one general-purpose filter. Furthermore the resulting CADe scheme turned out to be efficient for everyday clinical use. Future improvements should focus on the further reduction of false positive rate to ensure that CADe increases only the number of true positive diagnoses and not the number of unnecessary examinations.

References

[1] R. C. Hardie, S. K. Rogers, T. Wilson, and A. Rogers, “Performance analysis of a new computer aided detection system for identifying lung nodules on chest radiographs,” *Medical Image Analysis*, 12(3):240–258, June 2008.

[2] 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.

[3] G. Orbán, Á. Horváth, and G. Horváth, “Lung Nodule Detection on Rib Eliminated 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. 363–366. Springer Berlin Heidelberg, 2010, 10.1007/978-3-642-13039-7_91.

GPU-BASED ACCELERATION OF THE AUTODOCK MOLECULAR DOCKING SOFTWARE

Imre PECHAN
Advisor: Béla FEHÉR

I. Introduction

Molecular docking is a computationally expensive bioinformatics algorithm whose aim is to determine the binding geometry and binding energy of a small molecule called ligand inside the active site of a huge protein macromolecule called receptor. Docking methods are used by the pharmaceutical industry for identifying drug candidate molecules (competitive inhibitors) during the computer-aided drug design process.

AutoDock is a popular, open-source docking software. The FPGA-based acceleration of AutoDock was described in an earlier work [1]. This paper surveys the FPGA-based implementation, and describes the graphics processor-based acceleration of the same algorithm in details. Comparison of the two implementations shows that GPUs can be a real alternative of FPGAs in high-performance computing applications.

A The docking algorithm and its parallelization

The docking algorithm applied in AutoDock consists of a scoring function and an optimization algorithm. The scoring function models different chemical interactions and gives the free energy of a given geometrical arrangement of the molecules. The optimization algorithm changes the position of the molecules, that is, it moves and rotates the ligand relative to the receptor, and tries to find the energetically most favorable geometry, which is the global minimum of the scoring function. This optimal geometry is the predicted binding pose, and the corresponding energy value the predicted binding energy. AutoDock applies a genetic algorithm extended with a local search method for optimization. This means that after evaluating each entity of the new generation a small subset of the entities are subjected to an iterative local search process similar to hill climbing [1].

The algorithm can be parallelized at three levels. Due to the heuristic optimization algorithm usually several independent docking runs are performed for the same molecule pair so that reliable results are obtained. These independent dockings can be run simultaneously, which allows the high-level parallelization of the algorithm.

One docking run consists of evaluating millions of different potential binding geometries, which are grouped into generations according to the genetic algorithm. Each entity of the same generation can be evaluated simultaneously, then each entity of the selected subset can be subjected to the local search process independently. This can be treated as the medium-level parallelization of the algorithm.

In addition, evaluating a geometry consists of three main steps, and each step enables a low-level, fine-grained parallelization. First the degrees of freedom, that is, the genes of the new entity have to be determined, which describe the current position of the ligand. Both in case of the genetic algorithm and in case of the local search the genes are independent from each other and can be generated simultaneously. Then the coordinates of ligand atoms have to be calculated, which consists of performing a lot of rotations, and rotating different ligand atoms are independent operations. Finally, the value of the scoring function has to be determined. The energy value consists of two terms: the intramolecular and the intermolecular energy. In case of the former the scoring function has to be calculated for every atom pair of the ligand, in case of the latter an interpolation formula has to be determined for every ligand atom, but in both cases the required operations for different ligand atom pairs or atoms can be performed simultaneously [1].

B The FPGA-based implementation

The FPGA-based implementation of AutoDock consisted of a three stage pipeline, each stage performed one of the main steps mentioned above. As a consequence, three entities were evaluated simultaneously, so the medium-level parallelization was exploited only partially. On the other hand, each of these main stages consisted of fine-grained pipelines that took advantage of the low-level parallelization. Therefore the first main stage produced a new gene value, the second one performed one rotation, and the third one calculated the energy contribution term of one ligand atom or atom pair in every clock cycle of the FPGA. The implementation did not exploit the high-level parallelization of the algorithm, the independent docking runs were executed one after the other. The FPGA achieved a ×23 average speedup over a 3,2 GHz Intel Xeon CPU [1].

II. GPU for molecular docking

A The NVIDIA CUDA architecture

Graphics processing units are highly parallel, multithreaded, manycore processors optimized for data-parallel applications, which consist of performing the same operations on many independent data elements. GPUs are capable of running hundreds of threads in parallel, each of whom executes the same code but processes different data. GPU architectures are optimized for image processing and rendering algorithms, but they are nowadays widely used for accelerating general purpose, massively parallel algorithms as well.

CUDA (Compute Unified Device Architecture) is the general purpose parallel computing architecture of NVIDIA GPUs, which defines a parallel programming model based on high-level programming languages such as C or FORTRAN. CUDA C includes minimal language extensions to standard C, and enables the programmer to write a C code consisting of serial parts executed on the host CPU, and parallel kernels. Kernels are C functions which are executed M times in parallel by M different CUDA threads on the GPU. These threads are grouped into thread blocks. Threads within the same thread block can cooperate with each other by using synchronization barriers and by communicating via shared memory. Such a cooperation is not possible between different thread blocks since these will be scheduled and distributed among the cores of the GPU in a random, nondeterministic order. This leads to automatic scalability; among ideal circumstances a GPU with twice as many cores will execute the same kernel twice faster [2].

Physically a GPU with CUDA architecture consists of multiprocessors. Each multiprocessor consists of a set of processing cores, a large amount of registers, shared memory and a scheduler. When executing a kernel, each multiprocessor has a certain number of active thread blocks – these are executed by the multiprocessor logically in parallel. The registers and shared memory of the multiprocessor are distributed among these active thread blocks so the number of per-thread registers and the size of per-block shared memory used by the kernel limits the maximal number of active thread blocks that can be assigned to a multiprocessor at one time. The multiprocessor manages, schedules and executes the threads of its active thread blocks in groups of 32 threads called warps. Threads within the same warp are executed physically in parallel, that is, the multiprocessor is able to execute a common instruction of every 32 thread of the warp at the same time. Warp divergence occurs when threads of a warp takes different execution paths after a conditional branch, which will be executed by the multiprocessor serially [2].

The described architecture infers some programming considerations. The number of threads within a block should be a multiple of 32 in order to avoid under-populated warps. A kernel should include as many blocks as possible so that the execution time can scale well with the number of GPU multiprocessors. Care should be taken when choosing the amount of registers and shared memory which is used by the blocks and threads of the kernel, since this directly influences the maximal number of active blocks or active warps that can reside on each multiprocessor at one time. The more this number is, the better the scheduler of the multiprocessor can hide the latency of instructions, register or external memory access by switching between active warps. On the other hand, it is recommended to use registers and shared memory instead of external memory for storing program variables when possible, since the latter has a much lower bandwidth.

B The GPU-based implementation

There are two facts which suggest that GPUs are a promising target platform for molecular docking. Due to the fact that generally many independent docking runs are performed for the same input data, a system with K CPUs could be K-times faster than a single CPU. Since GPUs have manycore architecture, this high-level parallelization could be exploited with GPUs as well. In addition, unlike the majority of bioinformatics-related algorithms, which are based on bitstring processing and comparison operations, AutoDock mainly consists of floating-point arithmetic operations, which are the strength of GPUs.

The basic idea behind the CUDA-based implementation of AutoDock is to exploit the high- and medium-level parallelization of the algorithm with the parallel blocks, and the low-level parallelization with the parallel threads of the same block in the GPU. The original algorithm repeats two steps: first it generates a new generation of entities form the previous one and evaluates the scoring function for each of them, then it performs the local search process for a subset of these entities. In accordance whit this, the implemented software includes two kernels. Kernel1 generates a single entity from the previous generation, so it realizes the genetic algorithm, and evaluates it. Kernel2 performs the local search process for a given entity. The biggest part of both kernels is evaluation of the scoring function (rotation of ligand atoms and calculation of the energy terms), so they differ only in how the values of the degrees of freedom are chosen. The code that runs on the host CPU first launches Kernel1 for each new entity of every independent docking run, then it launches Kernel2 for each entity of every run which was selected for local search. The process is repeated until stop condition is satisfied for every runs. As a consequence, the GPU-based implementation fully exploits the high- and medium-level parallelization of the algorithm. The number of parallel thread blocks in case of Kernel1 and Kernel2 are N*popsize and N*popsize*lsrate, respectively, where N denotes the number of requested docking runs, popsize denotes the number of entities in one generation, and lsrate is the percent of entities that will be subjected to local search. Typically psize is 150, lsrate is 0.06, and N is between 1 and 100, but the latter highly depends on the complexity of the molecules and is set by the user.

The kernels perform the three main steps described earlier: they generate the values of the degrees of freedom, rotate the ligand atoms, and calculate the energy terms. Both of the kernels use 32 threads within a block. In case of the first step every thread is assigned to a different degree of freedom. During the second step independent rotations are executed by different threads in parallel. In case of the energy term calculation, the ligand atoms and ligand atom pairs are distributed among the threads evenly, so 32 energy components are calculated simultaneously. If the number of degrees of freedom, the number of required rotations, or that of the ligand atoms and atom pairs are not an integer multiple of 32, some threads will remain idle in that step, that is, thread divergence occurs. However, using a different number of threads would not be advantageous either due to the warp-based execution model. Actually, this means that the low-level parallelization of the algorithm could not be exploited as optimally in the GPU as in case of the FPGA-based implementation.

III. Evaluation and comparison

The FPGA-based implementation proved to be 23 times faster than a 3,2GHz Intel Xeon CPU during a test which consisted of docking runs for about 60 receptor-ligand molecule pairs. In case of the GPU-based software test runs were performed for the same set of molecules on two different NVIDIA GPUs: the GeForce GT 220 with 6 multiprocessors, and the GeForce GTX 260 with 24 multiprocessors. Run times were compared to the same Intel CPU so that the speedups of the GPU-based and FPGA-based implementations could be directly compared to each other.

Since the FPGA-based implementation does not apply the high-level parallelization of the algorithm, the speedup of the FPGA does not depend on the number of required docking runs (N). On the GPU, however, value of N determines the number of thread blocks that can be launched in parallel. Basically, the higher this value is, the better the capabilities of the GPU can be utilized, thus N directly influences the average runtime of one docking run on the GPU, that is, the speedup of the GPU-based implementation.

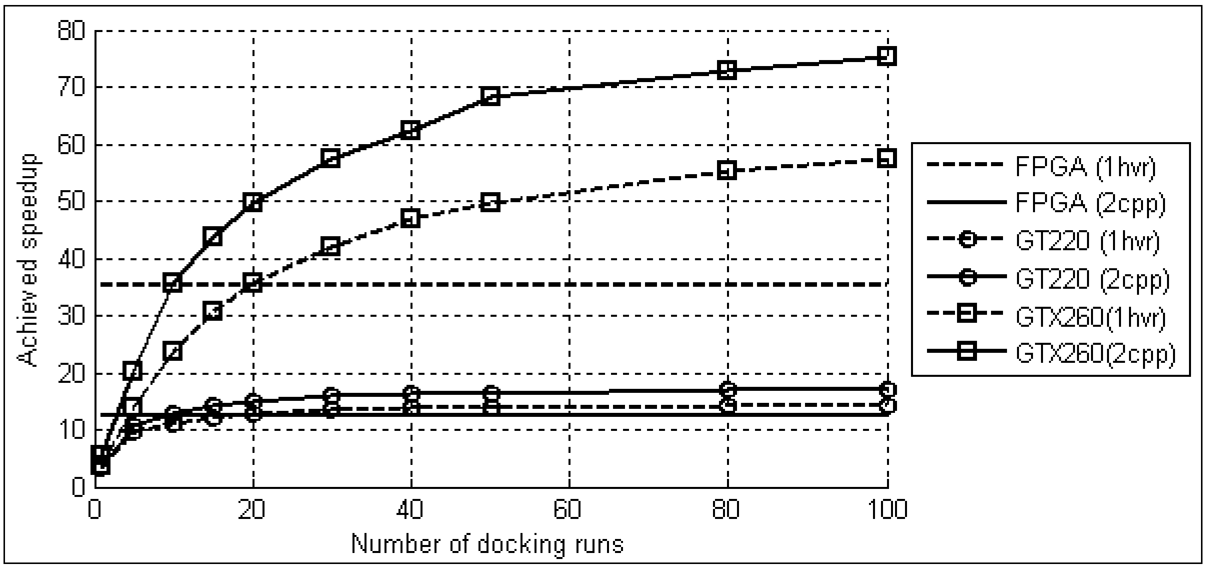


Figure 1: Speedups vs. number of runs (N)

Figure 1 shows the achieved speedups of the FPGA-based implementation and the GPU-based software on the two different GPUs versus N for two typical receptor-ligand complexes (1hvr and 2cpp). In case of 1hvr the speedup of the FPGA is ×35. If N=1, both of the GPUs are much slower, their speedup is about ×3. As N increases, the speedups of the GPUs get higher, and at one point they become saturated. This occurs when the number of parallel thread blocks becomes much higher than the number of blocks which can be active on the multiprocessors at one time – in this case the resources of the GPU are fully utilized, and doubling the number of blocks doubles the runtime as well. In case of the GT220 this saturation point is at about 30 runs, and the maximal speedup is ×14. In case of the larger GPU the achieved speedup is ×57 at 100 runs, but the speedup could increase further for higher N values. Similar tendency can be seen in case of the 2cpp molecule complex; speedup of the FPGA is ×12, maximal speedups of the smaller and bigger GPUs are ×17 and ×75, respectively.

The achieved average speedup for the whole set of molecules is ×12 if N=10 and ×15 if N=100 in case of the GT220, ×30 if N=10 and ×65 if N=100 in case of the GTX260. These numbers reflect the fact that the latter GPU has four times as many multiprocessors as the former one – if the number of blocks is high enough (N=100), the GTX260 is about four times faster than the GT220. The FPGA has an average speedup of ×23, so the FPGA-based implementation is faster than the GPUs if N is low enough. If N=10, the GTX260 shows a similar performance than the FPGA, if N=100, the GTX260 is about three times faster. These speedup values confirm that GPUs can be used effectively for the molecular docking algorithm and are a real alternative to FPGAs in general for high-performance computing applications.

References

[1] I. Pechan and B. Fehér, “FPGA-based acceleration of the AutoDock molecular docking software,” in *Proc. of the 6th Conference on Ph. D. Research in Microelectronics & Electronics*, Berlin, Germany, July 18-21 2010.

[2] NVIDIA CUDA C Programming Guide, URL: http://developer.nvidia.com/object/cuda_3_2_downloads.html

MEASUREMENT-BASED TIMING FAILURE DETECTION

Balázs SCHERER
Advisor: Gábor HORVÁTH

I. Introduction

Automotive embedded software modules are typical examples of safety-critical systems. The testing process of such embedded systems includes white box and black or grey box test types [1]. black or grey box tests like limit value tests, functionality tests and regression tests are performed at the system or subsystem level, in the phase of module integrations. These tests are done by separate test teams that have no overview on the software source code. This paper describes the traditional way of black or grey box testing of an ECU (Electronic Control Unit). After presenting a current way of testing, and the failure modes covered by these traditional tests, suggestion for measurement-based timing failure detection is presented, which can complete these existing methods.

II. Traditional ECU testing

In the ECU testing process regression tests are typical grey box tests, that are made after software modifications. Regressions tests of an automotive ECU typically cover software modules like input/output, communication, on-board diagnostic, diagnostic communication, operation mode management and real-time operation system. These tests are done independently, focusing only on the given module and on its data and control connection to other modules.

The test environment of this regression tests contains the ECU, and a so called Laboratory car, which simulates the behavior of other ECUs, sensors and actuators of the car. The setup also contains a test controller PC that manages the behavior of the Laboratory car, executes the test scripts and runs the test measurement software. The test measurement software is able to monitor or change the internal global variables of the ECU. Most of the regression tests are done by checking, whether these internal variables contain the right values during the test cycle, and many test stimulus are given by modifying the proper internal variables of the system.

The most critical part of this set-up is the test interface that provides access to the internal variables of the ECU. There are two traditional types of test interfaces used: dual port RAM based test interface, and diagnostic communication based interface. The dual port RAM based solution provides a nonintrusive monitoring, but from many points of view it is obsolete. The diagnostic communication based test interface is a more widespread used solution. This technique is an intrusive one, however the load caused by diagnosis is treated as a normal load, and therefore if it is lower than a certain value, which is about up to 10% of the communication bandwidth, it should be handled by the ECU at any time.

III. Timing failure detection

The failure model of such a real-time embedded system [2] can be divided into sequential and multitasking real-time behavior based failures. Feedbacks and experiences have shown that the current test coverage for sequential failures is high enough, but the multitasking real-time failure detection is not perfect, so it should be improved. There are three categories of such multitasking and real-time failures: the timing failures, synchronization failures and interleaving failures. In our work we focus on the timing failures.

The most foundational thing of Timing failure detection is the static prediction or runtime measurement of software task WCETs (Worst Case Execution Time). These execution time predictions or measurements can be used to calculate the worst case response time for each task, and therefore analyze, whether the system will be able to behave in real-time in every situation by keeping the schedule of the tasks. Real-time systems out of their schedule can show symptoms like unwanted resets and strange behavior for a short period of time.

Static calculation of execution times and WCET is not a trivial task and many articles discuss its problems [3], [4]. The two main sources of WCET deviation come from low-level hardware optimization features, and high level path dependencies. There are many tools regarding this topic, some tools use pure static analysis of the program, while other tools combine static analysis with dynamic measurements of the execution times of the program parts. Unlike most applications of static program analysis, WCET tools must analyze the machine code, not (only) the source code. This means that the analysis depends on the target processor, so WCET tools typically come in several versions, one for each supported target processor or even for each target system with a particular set of caches and memory interfaces.

Another way of execution time verification is a purely measurement based solution, when the timing parameters are acquired during run time testing. Although this is theoretically simpler, but requires code instrumentation and its coverage is limited by the run time test (only execution paths affected by the run time test can be inspected).

To make our work easiest to apply in current test systems in the future, we decided to complement the existing grey box tests with a timing failure detector property. Therefore we used a measurement based approach that can be integrated into the test environment, using diagnostic communication.

IV. Measurement

From the measurement and instrumentation point of view the RTOS model of the system is the most important. Specifications for automotive RTOS support are described first in the OSEK/VDX OS standard, which is used as a base of AUTOSAR specifications for operating systems.

The RTOS model of a usual ECU can be described as:

- Fixed priority, preemptive scheduling.
- Fixed task periods with 2.5 ms, 5 ms, 10 ms, 20 ms, 30 ms, 50 ms and 100 ms tasks.
- Tasks with lower period have the higher priority.
- Interrupts have short and predictable execution.
- The worst case response time of the tasks can be calculated accordingly to the Deadline Monotonic Analysis [5] as it is can be done in many automotive OSEK/VDX OS based system.

In some RTOS the built in instrumentation can provide the value of the task execution time (***C***), however it is not a trivial task. Many industrial companies's RTOS hasn't got such instrumentation, and they refuse to make any such deep kernel level modification on their RTOS, because of its high testing cost, so we decided to not use this deep instrumentation. Nearly every RTOS has a very simple instrumentation that can provide the value of task response times (***R***), the number of interrupts in the last 2.5 ms time period, and the processor load in the last 100 ms period, so these basic parameters are used in our work.

Unfortunately the value of task response times ***R*** alone is useless, because a low ***R*** value can cover a high ***C*** value and vice versa. But, complementing the value of ***R*** with the task end timestamp (very simple measurement) can help. From the known priorities and the periodicity of tasks, the measured response time, the end of task time stamps the scheduling can be restored, as we implemented it and evaluated it in our Matlab based simulator, then in our test system. This statement is true with 2 restrictions.

The first is that we do not take into the account the effect caused by priority inversion (this restriction can be done in our cases). The second is to neglect the effect of the interrupts, which is a much harder problem to solve. Figure 1 shows an example for scheduling restoration, without compensating the effects of interrupts.

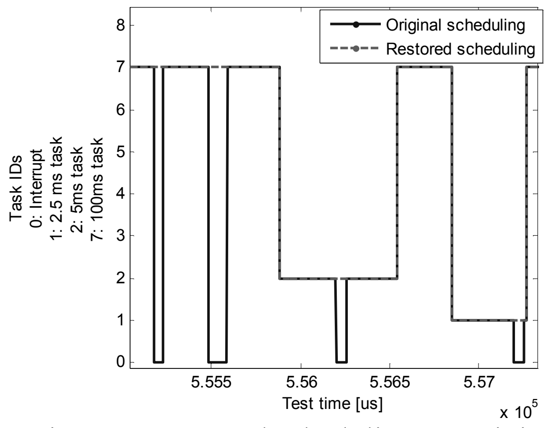


Figure 1. Restored scheduling vs. original

Interrupt modeling has two main parts to discuss. The first is the execution time prediction of an IT, which can be based on statistical WCET calculation results like using extreme value probability distribution. The second is the distribution of interrupt occurrences in time, which is a much harder problem. Some theories try to give probability modeling for the interrupt occurrences in time, which is may be usable in the case of human interactions, but our experiences have shown that in real time systems most of the interrupts has periodic nature, therefore these modeling can not be used.

V. Statistical response time calculation

Usually Deadline Monotonic Analysis (DMA) is used to calculate the worst-case response time of tasks. DMA uses an iterative formula for response time calculation, which is based on the execution times of the system tasks. In our example the execution times of tasks only can be given by using probability distributions instead of exact values. There are suggestions for statistical response time calculations in the literature [3] but generally response time calculation is a very difficult NP-hard problem. The problem is even harder in this situation. We should use uncertain arguments, because as presented before, the effects of interrupts cannot be modeled correctly, and in the procedure above we should use the IT models twice in the calculations. As a conclusion, the statistical response times can be calculated numerically, but the results could lead to a very long tail and imprecise probability distributions at the end. The worst case response time values can be selected from these distributions (usually there is a requirement of 10^{-8} or 10^{-9} probability limit), but as we studied it, these values doubtfully indicate healthy systems except from the trivial cases. Therefore these calculations would indicate a huge value of false positive detections, and therefore it is useless, with this data.

VI. Simplified, low data rate timing failure detection

The question is, whether there is any way to use this low amount of measurement data to make a usable statement about the timing healthiness of the system. Our suggestion is to use the reconstructed scheduling, but without trying to remove the effects of interrupts from it. The result is a set of overestimated executing times for each tasks. There is a guarantee that the maxima of these overestimated execution times will be higher than the worst case execution time in the test cycle. These overestimated worst case execution times could be used for the DMA algorithm, without taking into the account the effect of the interrupts, because the worst case overestimated execution times already contain the effects of the ITs. But there is no guarantee that the result of this calculation will be the realistic worst case situation, because the maxima of the overestimated execution times not definitely contain the maxima of the interrupt interferences. Therefore the possible differences between the interrupt effects embedded into the overestimated execution time maxima, and the worst case interrupt load should be compensated.

Our suggestion is not to make this compensation for each task, because that procedure would not fit to the mainly periodic nature of interrupt occurrences. So instead of compensating the possible interrupt differences for each task we suggest to take this compensation at the system schedule level. To do this we need a prediction for the schedule ability of the tasks, without calculating the worst case response times with DMA. The so called Liu and Layland bound can be used for this purpose [6]. The Liu and Layland bound gives a sufficient and hence conservative condition. A system may be schedulable though its maximum utilization exceeds this bound. Our suggestion is to use this conservative condition and complete it with the worst case IT difference:

$$U \leq n(2^{\frac{1}{n}} - 1) - Id$$

Where n is the number of tasks, U is system utilizations (U = 1 means 100% CPU load) and Id is the difference between the maximum IT load measured during the test, and the minimum of IT loads of the periods, which are used for the calculation of U. A period is used for the calculation of U if an overestimated task execution time is selected from it. The IT load is calculated by subtracting the sum of the overestimated task execution times from the CPU load for the period of time the CPU load was measured (usually it is synchronized to a low period task like the 100 ms task). This IT load is therefore the time when the CPU was handling interrupts during the execution of the idle task. Id = 0 means that the periods where the overestimated task execution times are selected from, had the highest IT loads during the run of the *Idle* task.

Our experiences have shown that this is a really conservative higher bound. If the system passes this, then there is a very low probability any timing failure remains in the measured test case. The diagnostic communication load of this solution is not more then 8% with using CCP messages in a 500 kbit/sec CAN network. Therefore it can be applied in traditional grey box tests.

Conclusions

Our paper has shown a simplified low data rate solution for timing failure detection. This method can be used in the traditional test environments with diagnostic communication. It can give a high probability solution for signaling situations from the measured data that can lead to timing failure. This presented method is usable, but far from the complete solution of the problems described in this paper. For a much complex solution we should take into the account that the tested systems are not full functional ones, and we are performing regression tests on them. We also should investigate the usability of modern hardware tracing technologies, but these topics are for further research.

References

[1] Thomas Müller and others, “Certified Tester Foundation Level Syllabus”, International Software Testing Qualifications Board, Version 2010. www.istqb.org. 2010.01.10

[2] H. Thane. “Monitoring, testing and debugging of distributed real-time systems” In Doctoral Thesis, Royal Institute of Technology, KTH, S100 44 Stockholm, Sweden, May 2000. Mechatronic Laboratory, Department of Machine Design.

[3] Insup Lee, Joseph Y-T. Leung, Sang H. Son, editors, “Handbook of Real-Time and Embedded Systems”, Chapman & Hall/CRC, 200y, ISBN-10: 1-58488-678-1.

[4] R. Wilhelm, and others, “The worst-case execution time problem - overview of methods and survey of tools” ACM Transactions on Embedded Computing Systems, Volume 7, Issue 3, April 2008.

[5] Ken Tindell “Deadline Monotonic Analysis”, Embedded System Progrming magazine, June 2000.

[6] C. L. Liu and J. W. Layland.”Scheduling algorithms for multiprogramming in a hard-real-time environment”, Journal of the Association for Computing Machinery, 20(1):46-61, January 1973.

Wavelet Transform Based Methods in Digital Audio Processing

Robert GALAMBOS
Advisor: Laszlo SUJBERT

I. Introduction

Nowadays the computational capacity of computers and DSP-s increased radically letting the audio processing move to the digital domain, where new possibilities are available. Beside the well known tools, and algorithms for analyzing and synthesizing digital signals there are special solutions and algorithms adapted to the features of the human perception.

II. Time and frequency resolution

The digital signal can be processed in time domain or transformed – e.g. with the Fourier transform – into the time-frequency plane. The time-frequency resolution is limited by the uncertainty principle [1], that means that the resolution of the time and the frequency can not be arbitrary large at the same time.

A. *Fourier transform*

The basic element of the continuous-time Fourier transform is an infinitely long complex periodic wave. The transformation of this wave is a so called Dirac impulse. We can not talk about time resolution, because the time domain signal is infinitely long.

By sampling this continuous signal we get to the discrete-time Fourier transform. Still there is no time resolution because in time domain the sampled signal has infinite number of samples. To be able to localize the signal we need to multiply it with a window function, and "cut out a piece" in the time domain.

This is the so called short-time Fourier transform. The multiplication of the signal and the window function in time domain, is convolution in the frequency domain, so the signal itself is modified by the convolution with the window functions Fourier transform [2]. The short-time Fourier transform has the same window function for each frequency location, resulting an equidistant grid on the time-frequency plane. Because the human ear recognizes the frequencies in a logarithmic manner, the Fourier transform might not produce as good result as a transformation would do that follows this logarithmic manner.

B. *Wavelet transform*

The wavelet transform can be seen as a modification of the short-time Fourier transform. The window function multiplied signal is replaced by the so called mother wavelet, a special function with scale and shift parameters. This is the source of the daughter wavelets. They are derived from the mother wavelet by shifting and scaling. The mother wavelet must vanish at zero frequencies, so it has a band-pass like characteristic. So the average of the mother wavelet must be zero, it has to be a "wave". The daughter wavelets will have another central frequency and bandwidth in the frequency domain, so the time-frequency plane grid structure will not be equidistant.

From the multi-rate signal processing point of view the wavelet transform can be implemented as filter bank based calculation, where lower frequencies are represented more accurate in the frequency parameter, and higher frequencies are represented more accurate in the time parameter.

Following the filter bank approach, and because the wavelets must vanish at zero frequency it is not possible to cover the whole frequency range with the wavelets. There will be always a blind area under the lowest band. There is a so called scaling function that covers the near zero frequency range.

Because the wavelet filter bank is built up in this way, it is possible to use sub-band coding [3] to calculate the wavelet transform of a signal. This can be done in two ways. We can use parallel bandpass filters, or an iterative filter bank structure. The latter has the advantage that the calculation complexity is less.

III. Practical occurrences of Wavelet transform

There are several situations where wavelet transform can be used instead of the usual methods (e.g. Fourier transform). With the advantage of fitting better to the human ears perception the wavelet transform is fairly a better choice for audio applications.

Decomposing the signal into wavelets makes it possible to filter the noise on each decomposition to reduce the overall noise of the signal. Each decomposition has a threshold level, and if the amplitude falls under this level, the gain of the given part will be muted or dumped [4].

Watermarks are special signs in the audio (or image) to store information, which can be used to identify the author of the signal, or to store copyrights. Wavelet transform can be used for giving another representation of the signal producing more robust watermarks [5].

The wavelet decomposition can be used for data compression. By decomposing the signal to wavelets, it is possible to simplify the dimensions needed to describe the signal [6].

Wavelet transform can be used in special audio effect algorithms, to produce a higher quality output. For example there is a big advantage of using wavelet transform in a phase vocoder algorithm instead of using the usual short-time Fourier transform. In this case the fine details of the signal remain after the processing, and less distortion is applied.

As the short-time Fourier transform the wavelet transform can be used in audio identification algorithms, by creating a fingerprint from the wavelet coefficients. This fingerprint must be robust against noises and other distortions on the audio signal. It is used for example for automatic music detection.

For modelling musical instruments it could be a good choice to use wavelet based analysis and/or synthesis and so allow the model to use less computational capacity.

IV. Conclusions

The wavelet transform is a transform that comes closer to the human perception because of its logarithmic manner. According to this it is possible to achieve better results with this transformation than with the usual Fourier transform. The difference is in the resolution of the time-frequency plane. Where the short-time Fourier transform has equidistant grids the wavelet transform has more option. It is possible to achieve a frequency resolution on the needed level in high and low frequency ranges. Beneath these features in some cases the wavelet transform can be calculated with a small amount of computation power by using the so called fast wavelet transform.

References

[1] C. K. Chui, Ed., *An Introduction to Wavelets*, Academic Press, 1992.

[2] G. Kaiser, Ed., *A Friendly Guide to Wavelets*, Birkhauser, 1994.

[3] S. Mallat, Ed., *A Wavelet Tour of Signal Processing, Second Edition*, Academic Press, 1999.

[4] E. B. Guoshen Yu, Stephane Mallat, "Audio denoising by time-frequency block thresholding," *IEEE Transaction on Signal Processing*, 56(5):1830–1839, 2008.

[5] A. M. Ali Al-Haj, "Digital audio watermarking based on the discrete wavelets transform and singular value decomposition," *European Journal of Scientific Research*, 39(1):6–21, 2010.

[6] M. Zanartu, "Audio compression using wavelet techniques," Tech. Rep., Purdue University, 2005.

COMPARISON OF CORTICAL CONNECTIVITY MODELS IN SCHIZOPHRENIA BASED ON FMRI DATA

Mihály BÁNYAI
Advisors: Péter ÉRDI, Fülöp BAZSÓ, György STRAUSZ

I. Introduction

Schizophrenia is a complex disorder with diverse related impairments in the central nervous system. Specifically, altered function of the prefrontal cortex (involved in cognitive control of cortical processes) and hippocampus (involved in associative memory formation) and altered interaction of these areas [1] is hypothesized to be a central aspect of its pathophysiology and may be related to anatomical and/or functional disconnection [2]. Here we investigated impaired functional macro-network interactions in schizophrenia by applying Dynamic Causal Modelling (DCM) [3] to the analyses of fMRI data collected during a paired-associate learning paradigm. Functional disconnection means that instead of physiological alterations, we are looking for changes in the task-related coherence in activity of the areas. The aim of modeling interactions of brain areas using DCM is to characterize: a) the intrinsic connectivity of the network and b) the contextual modulation of the intrinsic connections by psychological aspects of the task.

The relevance of the concept of functional disconnection for interpreting schizophrenia is established by previous studies reporting impairments in functional macro-networks [4], and based on brain imaging experiments it was suggested that reduced performance of schizophrenic patients in cognitive tasks requiring working memory is related to abnormal prefronto-hippocampal connectivity.

Task-related functional connectivity can be investigated with respect to various functions of the brain, e.g. learning, memory, control, etc. We studied associative learning, since this is a cortical function that requires the integration of multiple sensory, representation and cognitive control pathways, making it a useful approach to grasp disordered functional interaction [5].

II. Methods

A. *Associative learning: behavioral task and data*

We adopted a paired-associate learning paradigm in which subjects are required to learning arbitrary associations between locations (in space) and objects (with unique identities). 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 m^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.

B. *Dynamic causal modelling*

DCM provides a complete phenomenological model framework for the analysis of fMRI data. For a detailed description see [6]. 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. This is a bilinear formula of the state variables themselves and the input variables, u, which are the conditions defined by the experiment

(Eq. 1). 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.

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

$$y = \lambda(x, \theta_h) \tag{2}$$

The hemodynamic component describes the nonlinear mapping from the neural activity to the fMRI signal, y, actually measured in the brain areas (Eq. 2). For the details of the hemodynamic model see [7]. We need to estimate the values of the parameter set, $\theta = \{\theta_h, \theta_n\}$ best fitting to measurement data. One possible procedure to do so is the Bayesian maximum a posteriori (MAP) estimation technique defined by Eq. 3, where M denotes the specific connectivity pattern of the model.

$$p(\theta \mid y, M) = \frac{p(y \mid \theta, M)p(\theta \mid M)}{p(y \mid M)} \tag{3}$$

For all probability distributions in 3, 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.

C. *Comparison of models*

We can compare models with different connectivity patterns in a Bayesian way by estimating their model evidence:

$$p(y \mid M) = \int p(y \mid \theta, M)p(\theta \mid M)\, d\theta \tag{4}$$

The model evidence is the probability of obtaining the actual measurement conditioned on the model form, integrated on the whole parameter space of the model. The computation of the evidence is usually not feasible, so we approximate it by a variational Bayesian method. To obtain the expected posterior probabilities of all models in a model set, we assume a hierarchical model of data generation. We can invert this model using a variational Bayesian method that requires only the estimates of the log-evidences for each subject-model pair. This way we obtain the expected posterior probability of each model regarding the subject group. For a complete description of the comparison method see [8].

D. *Model definitions*

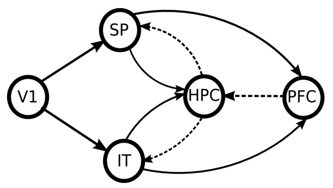


Figure 1: *The model space for varying intrinsic connections. Connections marked by dashed line are varied.*

A set of causal models of fMRI signal generation (with the mathematical structure described above) were defined to evaluate connections between five regions material to the task. To model the information processing in the associative learning task, we assumed the presence of two streams connecting the five brain regions. The ”forward” or ”data” stream propagates sensory information at different levels of processing from the low-level sensory areas towards high-level cognitive areas. The ”backward” or ”control” stream propagates control signals from the high-level areas towards the lower-level ones. In this paper we examine impairments in cognitive control, so the focus of the investigations is the control stream.

First, multiple models were evaluated by varying hypothesis-related intrinsic connections between regions, while fixing other connections (A matrix). We included the intrinsic connections of the data stream to all models. Based on the hypothetical control stream we defined three additional connections that may extend the basic model in different combinations. The eight possible combinations of these connections constitute the first model class. All possible intrinsic connections are visualized in Fig. 1. A second set of models were evaluated by fixing all intrinsic connections (both data and control streams) and some of the contextual modulations, and varying the hypothesis-related modulatory connections (B matrix).

III. Results

All models were fitted to the measurement data from all subjects, sorted into two groups, schizophrenia patients (SCZ) and healthy controls (HC). The computational procedure applied serves two quantities, the maximum a posteriori (MAP) parameter values and estimates of model log-evidences. The results can be evaluated on multiple levels of abstraction. First, we made an intra-group model comparison to find out which model structures are more probable than others in the subject groups, then we investigated the differences on the level of the individual parameters.

A. Model comparison

The goodness of a model can be quantitatively described by its posterior probability obtained from Bayesian model selection. The expected posterior probabilities are plotted in Fig. 2. The comparison was done within the first and second model classes separately.

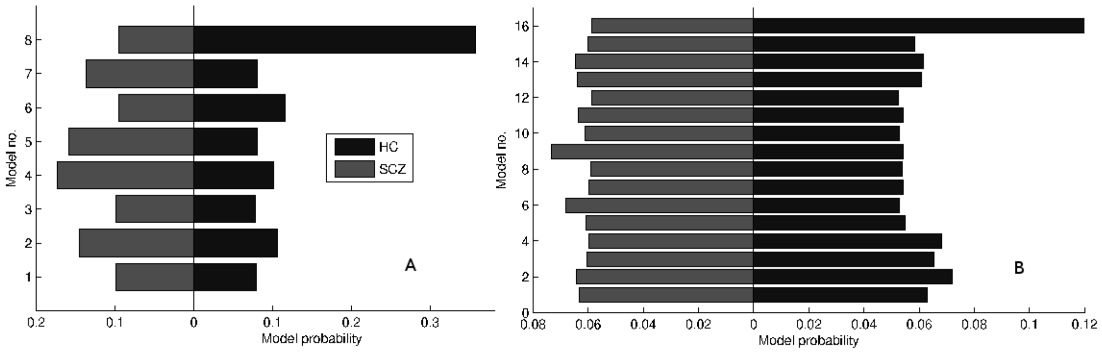


Figure 2: *Posterior probabilities of models with varied intrinsic (A) and modulatory (B) connections in the two subject groups. The two colors indicate the probabilities of models calculated independently based on the data of the healthy and patient groups (assuming that only the models in our model space are possible).*

The results show that in the control group there is a clear winner for both the intrinsic and modulatory connection patterns, the model that contains the full control stream. In the SCZ group, there is no clear winner, there are several more probable models, and the differences are smaller between model probabilities. It can also be seen that while the winning model in the HC group contains all the connections defined, while the most probable models in the SCZ group lack more or less connections. This result implies that the information processing network of schizophrenia patients is fundamentally different than the one of controls. However, the model selection does not provide the specific pathways being impaired, so the parameter level analysis is also necessary.

B. Effective connectivities

In the next step of the analysis, we give a more detailed quantitative characterization of the results. At the parameter level, we look for significant differences in the effective connectivity in the models fit to the data of the two subject groups, assuming fixed model structure. To do so, we selected a reference model for comparison by running the model selection for all subjects, no distinction by group.

The winning model is the one containing all hypothesized connections. The means and standard deviations of the intrinsic coupling and modulatory parameters are depicted in Fig. 3. To obtain the significance of the differences we applied a two-sample t-test on the parameter values in the two groups.

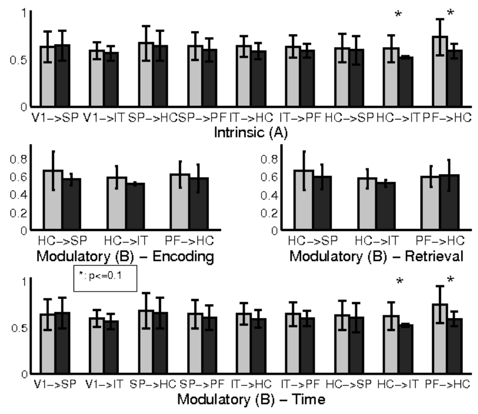


Figure 3: *Average connectivity parameters for HC and SCZ groups.*

The significant differences between groups are in the strength of intrinsic connections between prefrontal cortex and hippocampus and between hippocampus and inferior temporal cortex. All these connections are weakened in the SCZ group, supporting the hypothesis about impaired effective connectivity in the control stream in schizophrenia. Both these connections are playing important roles in the cognitive control of the associative memory formation. Furthermore, we see the reduced effects of Time on these causal links meaning reduced excitatory contextual modulation by learning. This can be seen as reduced task-related plasticity of a pathway in the illness.

IV. Conclusion

Bayesian model selection was used to investigate hypotheses on differences in the information processing network implemeted by scizophrenia patients and healthy controls during an associative learning task. The method indicated fundamental differences, models lacking connections related to cognitive control were more probable in the patient group. Differences in effective connectivity between groups were tested by comparing coupling parameters in winning model structures. This analysis indicated reduced fronto-hippocampal and hippocampo-inferior temporal coupling in patients. These findings may account for the reductions in learning performance of schizophrenia patients.

Acknowledgement

I am grateful to Vaibhav Diwadkar for conducting the experiment and helping to define connectivity models, and also to Balázs Ujfalussy for insightful discussions.

References

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

[2] K. J. Friston, "The disconnection hypothesis," *Schizophrenia Research*, 30:115–125, 2007.

[3] K. E. Stephan, "Dynamic causal models of neural system dynamics:current state and future extensions," *Journal of Biosciences*, 32:129–144, 1998.

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

[5] S. A. Bunge, B. Burrows, and A. D. Wagner, "Prefrontal and hippocampal contributions to visual associative recognition: interactions between cognitive control and episodic retrieval," *Brain Cognition*, 56:141–152, 2004.

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

[7] 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.

[8] 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.

NEURAL MODELS FOR AN INTELLIGENT GREENHOUSE - THE HEATING

Péter EREDICS
Advisor: Tadeusz P. DOBROWIECKI

I. Introduction

Greenhouses are building structures widely used in vegetable production and for growing ornamental plants or flowers. Solar radiation passing through the transparent walls and roofs is essential for the photosynthesis, and supplements heating in the cold season. In hot weather other actuators, like roof vents, shading systems, exhaust fans or evaporative cooling may be used to avoid overheating. In most modern greenhouses these automated actuators are operated by some kind of control system.

Control systems for greenhouses available on the market have not changed much in the last years: actuators are individually controlled based on set-points and actual measurements [1]. This traditional control design has three major drawbacks: (1) The adjustment of set-points depends strongly on the expertise of the greenhouse operator. (2) The control system is reactive: without predicting the future state of the greenhouse, it is impossible to control effectively for a long time horizon. (3) The actuator operations are unsynchronized (all are set independently from each other), resulting in possible oscillations in the control and poor maintenance of the internal climate.

The solution overcoming these limitations is to increase the level of intelligence of the system by applying an intelligent control solution [2].

II. Intelligent Control

The primary objective of greenhouse control solutions is to provide suitable environmental conditions for the plants. The traditional form of control depends on human intelligence. The operator will in theory be able to choose optimal set-point combinations. Unfortunately with several actuators finding the optimal control configuration intuitively, without serious theoretical modeling and computing, is impossible. Consequently optimal control in this case is neither possible. Yet the greenhouse operator is fully aware of the physiology of the plants and their physical and chemical needs. The control system therefore should operate on this available information. Instead of accepting set-points, an intelligent control system should expect global control goals.

The concept of control goal as the direct control information makes the human interaction easier, but the knowledge intensive transformation from the goals to the control actions is left to the control system. This transformation can be implemented with predictive modeling, which also solves the second problem of traditional control, namely its reactiveness. Predictive modeling means in this case making assumptions about the actuator settings, and predicting the future thermal states of the greenhouse. These thermal states reported over a given time span can be then evaluated with respect to the control goals. The costs of the actuator setting and the deviation from the goals can be fused together into a numerical cost function. By computing the minimal value of this cost, e.g. by trying all different actuator configurations, makes it possible to find the most appropriate actuator settings.

Predictive modeling solves in part the missing synchronization of the actuators, but the common problem of traditional control, i.e. swinging controlled variable, still remains. The possible control loops, repeatedly setting and resetting the actuators cannot be avoided this way. All such problems can be however handled by (AI) planning. We expect that the concept of intelligent control will provide solutions to all principal limitations of the traditional greenhouse control, with better environmental conditions for the plants and lower costs for the owners.

III. Modeling the Heating

A The Modeling Problem

The necessary basis of the intelligent control is the prediction of the future thermal state of the greenhouse, thus the first step is modeling. The greenhouse model must be able to predict all important internal and external parameters of the house for a reasonable span of the time. This paper focuses on the modeling of the heating system. This subsystem within the greenhouse is very important because it is the main financial cost factor of the greenhouse during the cold period of the year. Fig. 1. (A) shows an example of the heating pipe temperature recorded on 11-02-2009 with the heating turned on 11 times. The modeling of the heating system can be easily separated from the whole greenhouse because the involved quantities are relatively independent from other quantities affecting the greenhouse. On the other hand the influence of the internal and external temperature on the heating pipe temperature cannot be completely neglected. Fig. 1. (B) shows 10 different graphs recorded in the greenhouse all starting from 25 °C heating pipe temperature when the heating was turned on. Depending on the actual internal temperature and the changes in both internal and external temperatures (even weather changes) different graphs of the heating pipe temperature were obtained.

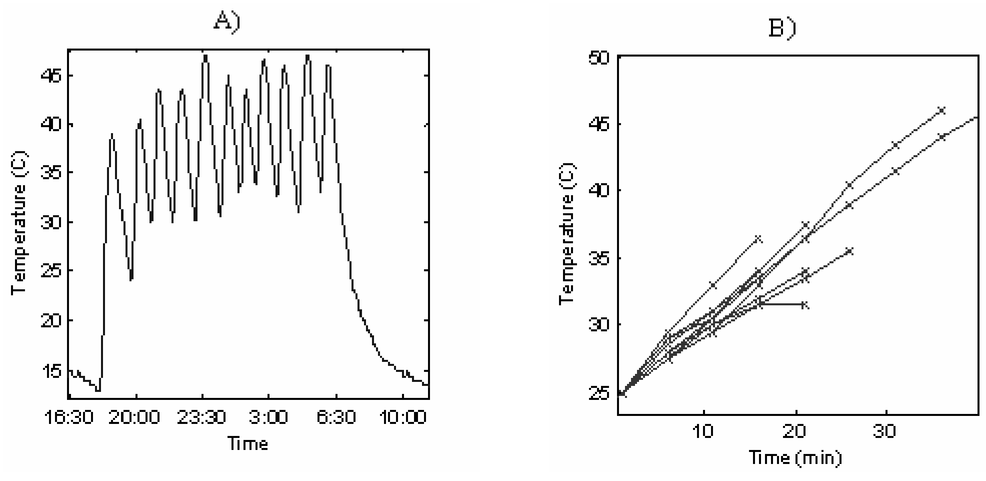


Figure 1: A) The recorded heating pipe temperature on 11-02-2009; B) Different graphs of the heating pipe temperature both starting from 25 °C

B Using Prediction Tables

In the given greenhouse the temperature of the heating pipe is always in the range of 0-50 °C. The resolution of the measurement system is 0.5 °C, which means that the heating pipe has 100 states determined by its temperature. This small state space makes it possible to inspect all states separately and store the prediction from each state in a table for easy retrieval. In practice two tables are needed: one for the warming state of the pipe and another for the cooling state. Examining the data from the greenhouse for the former case it is evident, that after activating the heating, it was almost always operating for at least 20 minutes. The time resolution of the measurement system is 5 minutes thus we have many training data with at least 4 steps. In the later case we have more training data, but most of the time the pipe temperature passively follows the internal temperature. Accordingly we require the pipe to be warmer by 10 °C than the internal air to use these data as training examples.

Both tables can be filled in with data generated by the following algorithm described here for the warming case: the algorithm looks for any t time points in the measurements when the heating was on. If the heating was not turned off before t + 20, then t can serve as the beginning of an example. The prediction table has 5 columns and 100 rows. The first column is the starting temperature, so the algorithm looks for the T(t) (heating pipe temperature at time t) in the first column. After finding it, the T(t+5), T(t+10), T(t+15) and T(t+20) temperature values are inserted into column 2-5. If there are values in these cells, then weighted averages are stored. Other temperatures close to T(t) are also updated with small weight factors to smooth out the prediction. Fig. 2. (A) visualizes the values in the warming state table. Axis X is the starting temperature (T(t) in the algorithm) while axis Y shows the predicted value. The dashed line represents no change, other curves above each other are the predicted changes for 5/10/15/20 minutes ahead accordingly.

The main advantage of using the tables is simplicity: tables can be quickly generated and used. Because of the simplicity the precision of this solution is limited. This method can not handle other inputs (such as internal greenhouse temperature) as increasing the number of inputs would exponentially scale up the size of table. It is unacceptable with limited number of training examples.

C Using Monolith Neural Network

In order to gain more accurate predictions more inputs have to be considered than the current pipe temperature alone. The actual internal air temperature seems especially important, therefore it is also used as input in this method. The need for two separate networks can be eliminated if the control signal for the heating system is also considered as input. To handle these different inputs a neural network (MLP) was built from 20 neurons. The inputs are as specified above, while the outputs are the predicted pipe temperature values for 1-4 steps ahead. The only requirement for the example data records was that the heating pipe temperature is at least 10 °C higher than the internal air temperature. Fig. 2. (B) shows graphs for this method, where the curves are smoother because of the interpolation property of the network. Comparing the figures large differences can only be observed close to the Y axis caused by smaller number of examples covering the X axis here.

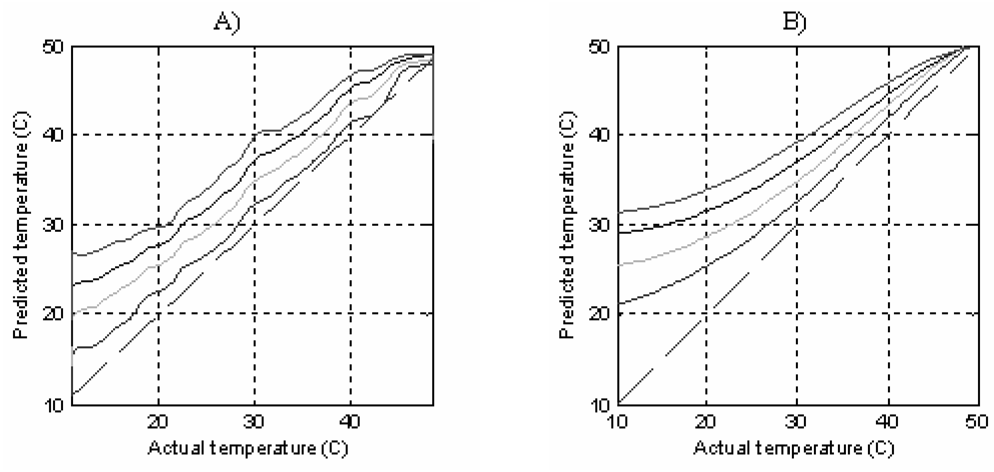


Figure 2: Visualization of the transfer function of the prediction table (A) and the monolith neural network method (B). The dashed line is the constant reference, graphs above are the predictions for 5/10/15/20 minutes ahead

D Neural Network Decomposition

The monolithic neural network solution has the advantage of modeling the whole process (both the warming and the cooling regime) with a single neural network, but the complexity of the network had to be increased to obtain good results. It seems reasonable to decompose it into separate neural models. The warming pipe model is realized by a neural network with 7 neurons in the hidden layer. The inputs and outputs are identical to the monolithic network discussed earlier, but the training samples were selected only from examples where the heating was on. This model has a very similar transfer function to Fig. 2. (B). The cooling pipe model has 8 neurons in the hidden layer and it has an additional input: the number of minutes since the heating was turned off. This network was trained with examples where the heating was off for the whole training sample therefore the heating control inputs could be omitted from both models. To create predictions the models are coupled with a simple control logic. Based on the planned heating control signal this logic switches from one model to the other.

IV. Results

The accuracy of the methods introduced in the previous section was compared by testing them on randomly selected validation heating sequences. Such a sequence is shown in Fig. 3. (A) for the decomposed neural method.

The table and the decomposed neural network method were applied repeatedly as many times as it was necessary. In both cases the switching between tables or models had to be explicitly implemented. The monolith neural network solution was able to handle changes of the heating signal on its own, thus it was easier to experiment with. In all three cases the models served predictions for 1-4 steps ahead. The table method on the validation example set had the lowest accuracy. The table method has a very noisy transfer function which means that it was unable to extract the smooth changes of the heating pipe temperature. We have to note on the other hand, that this method has the lowest computing complexity and in some environments (e.g. in embedded applications) it might be an important factor. The monolith neural network had better performance than the table method by the price of its higher complexity. This model is compact, but modeling the notably different warming and cooling processes together is not the optimal solution. The best accuracy was obtained by the decomposed neural network model. This solution had a lower complexity than the previous and it provided the best predictions. Using this method a test prediction for 12 hours was created and the predicted value was never out of the 1 degree proximity of the measured value while the heating pipe was notably warmer than its environment. This 12 hours long prediction is shown in Fig. 3. (B).

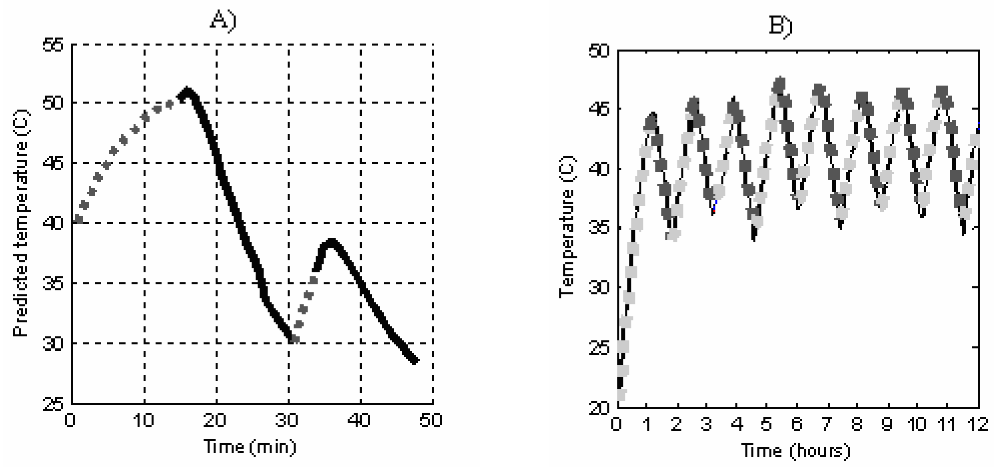


Figure 3: Predicting the heating pipe temperature for A) 50 minutes ahead with two active heating sessions (0-15 and 30-35 minutes on the X axis) with model switching and for B) 12 hours ahead

Acknowledgement

The authors gratefully acknowledge the support of the Hungarian Fund for Scientific Research (OTKA).

References

[1] F.S. Zazueta, R. Bucklin, P.H. Jones, A.G. Smajstrla: Basic Concepts in Environmental Computer Control of Agricultural Systems. Agricultural and Biological Engineering Dept, Institute of Food and Agricultural Sciences, University of Florida, 2008.

[2] X. Blasco, M. Martineza, J.M. Herreroa, C. Ramosa, J. Sanchisa: Model-based predictive control of greenhouse climate for reducing energy and water consumption. Computers and Electronics in Agriculture, pp. 49–70, Elsevier Science Publishers B. V. Amsterdam, The Netherlands, 2007.

A FRAMEWORK FOR GENOMIC REARRANGEMENT ANALYSIS

Péter LACZKÓ
Advisors: Béla FEHÉR, István MIKLÓS

I. Introduction

Cancer is a class of diseases in which cells display uncontrolled growth through division beyond the normal limits. The exact molecular biologic mechanisms of such severe alterations in the regulatory processes are widespread, however, in every case they can be traced back to somatic mutations of the tumor cell genomes [1]. These mutations range from single nucleotide polymorphisms to larger scale structural variations, in extreme cases even to chromosome aneuploidity. An important subclass of somatic mutations is genomic rearrangement, which involve the relocation of various blocks of the genome, sometimes in a rather complicated way. Understanding the dynamics behind these rearrangement events allows for the assessment of their oncogenic effects and the subsequent classification of tumor cell samples based solely on their genomic landscape, leading eventually to better diagnostic and prognostic methods.

In the era in which next-generation sequencing (NGS) technologies are becoming ubiquitous and the cost of whole human genome sequencing is decreasing rapidly, there is a clear rationale for the inception of a tumor genome classification framework which would make the aforementioned diagnostics possible. Our research aims at developing such a framework, incorporating every data processing step necessary to obtain the classification of a tumor cell from its raw next-generation sequencing data.

In this paper I briefly describe the nature of genomic rearrangements that we aim to detect and discuss other methods available for similar purposes. I describe our proposed framework and the algorithm behind the components where applicable. Finally, I introduce a simulator tool that we developed and which allows for rapid evaluation of prospective rearrangement detection methods.

II. The cancer genome

DNA in healthy human cells is continuously damaged by mutagens of both internal and external origins, leading to the accumulation of somatic mutations over the lifetime of the individual. These mutations range from single nucleotide polymorphisms to large scale structural variations. While their contribution to the oncogenesis can be deduced in some cases, and perhaps even more remarkably, their signatures sometimes give away the mutagenic factors which caused them [2], making the distinction between "driver" and "passenger" mutations is by no means trivial, especially in the case of large scale structural variations which do not always have a direct genetic linkage and therefore must be analyzed on the genomic scale.

Even simple structural variations such as insertions, deletions or inversions may have radical phenotypic effects, for instance, the formation of fusion (onco-) genes or loss of function in cancer suppressor genes [3]. The rearrangement of sometimes distant parts of the genome may join the sequences of two different genes to create a fusion gene or it may position the cancer gene adjacent to regulatory elements from elsewhere in the genome, resulting in abnormal expression patterns [1]. For example, clones obtained from the MCF-7 breast cancer cell line genome were found to have a complex internal structure [4], with some genomic regions extensively scrambled around a remarkable number of breakpoints. The rearrangement process is frequently associated with gene amplification, a somatically acquired increase in copy number of restricted genomic regions also known as amplicons.

III. Rearrangement analysis

A. *End-sequence profiling*

While several techniques outside the scope of this paper exist for genomic rearrangement analysis (e.g. comparative genomic hybridization, CGH), high resolution mapping of rearrangements is made possible only by sequencing techniques. The paper [4] describes a widely used method known as end-sequence profiling (ESP). Such an analysis begins with the cloning of the genome of interest into large vectors called bacterial artificial chromosomes (BAC). Subsequently, both ends of these clones are sequenced using traditional Sanger sequencing, and the sequence obtained is mapped back to the reference genome. The size distribution of the clones is known, therefore outliers (clones whose ends map unexpectedly close or far from each other) can be identified. These clones, containing a putative genomic rearrangement, can then be sequenced in their full length and rearrangements contained therein can be analyzed.

As Sanger-sequencing a full human genome is prohibitively expensive, the ability to sequence only the clones that are likely to contain a rearrangement breakpoint is a significant advantage of the ESP technique. A further advantage is due to the relatively long read lengths inherent to Sanger sequencing: the sequence of the clones can be determined with high confidence, resulting in base-pair level identification of breakpoints. On the other hand, rearrangements significantly smaller in scale than the size of the BAC clones may remain undetected as their effect in end mapping distance remains within the distribution of normal BAC end distance. Furthermore, full sequencing of the BACs is expensive, so only genomic regions of interest are mapped to a high resolution. The mapping distances of *all* the BAC ends *together* could be used to reconstruct at least large-scale rearrangements genome-wide; however, this was found to be a challenging computational problem [5].

B. *Next-generation sequencing methods*

Complex rearrangements have also been observed in NGS studies ([2]). While the ESP method allows for selection of relevant clones for capillary sequencing and thus for exact breakpoint localization, the short read technology involved in NGS methods makes the identification of complex rearrangements rather difficult (see [6] for a review). Most studies for automated structural variation (SV) discovery used a paired-end mapping strategy: both ends of short (200 - 1400 bp) fragments of DNA are sequenced and the reads are mapped to the reference genome. Abnormally mapping reads are compared to known signatures of simple insertions, deletions, inversions, insertions of longer segments of distant or novel sequence and translocations.

The detection of fingerprints of such simple SVs may be based on heuristics, possibly complemented with exact treatment of measurement error ([7]), or probabilistic (e.g. [8]) algorithms. While it is possible to manually detect complex rearrangements from discordantly mapping reads, neither the aforementioned methods nor any others I am aware of are capable of the automated description of a rearranged genomic landscape.

With short reads long enough to be uniquely aligned in part to the genome split-read alignment becomes possible. This technique can be used for direct detection of SV breakpoints captured by a single read. It is commonly used for identification of insertions and deletions, both as separate tools (e.g. [9]) or as a step of more comprehensive SV analyses. With high enough sequence coverage split-read mapping may also be used to map the breakpoints of other, perhaps more complex structural variations, including genomic rearrangements (e.g. [10]). These studies, however, only used these breakpoints to select genomic regions for further analysis by more precise experimental methods, and to the best of my knowledge, no split-read mapping approach for automated rearrangement discovery exists.

IV. The proposed framework

Our framework aims at the classification of individual tumor genomes based on their next-generation sequencing data. This process consists of three distinct steps, whose relation as well as their inputs and outputs are illustrated in Figure 1.

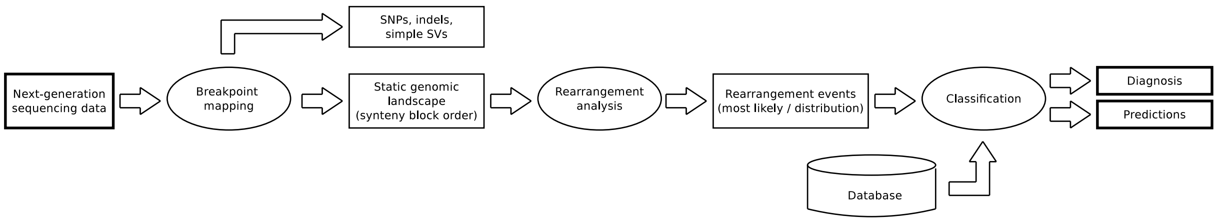

SNPs, indels, simple SVs
Next-generation sequencing data
Breakpoint mapping
Static genomic landscape (synteny block order)
Rearrangement analysis
Rearrangement events (most likely / distribution)
Classification
Diagnosis
Predictions
Database

Figure 1: Overview of the framework

The first step is the mapping of the rearranged genomic landscape from the short read data. The output of this preprocessing step is the relative order of conserved (so called synteny) blocks of the genome. Our method is based on split-read mapping, but unlike existing methods, it is designed for automatic reconstruction of the synteny block order. Reads which can be divided in two parts mapping to remote locations of the reference are referred to as *bridge reads*. Every such bridge read is considered as an evidence supporting an adjacency relation between the two genomic intervals ending at the breakpoint of the read.

The bridge reads are represented in a graph whose vertices belong to the interval ends, and edges represent the aforementioned adjacency relations. If we add an extra vertex for every interval (connected to its endpoints), symbolizing the passage through the interval, the problem of finding the order of synteny blocks reduces to finding a Hamiltonian path in this graph.

Obviously, one faces challenges applying this formal model to real-world data. First, next-generation sequencing data is inherently noisy: reads generated from repetitive or homologous regions of the genome map to several different locations, and introduce false adjacency edges to the graph. It is therefore very likely that multiple Hamiltonian paths exist in the graph, and our method must choose the one that is supported by the most reads. Second, the number of intervals and adjacencies in a real genome is rather high, which questions the feasibility of this NP-complete model. However, we found that the structure of the graphs obtained is surprisingly sparse, and can be further simplified by removing vertices with an unrealistically high number of putative neighbors (likely having resulted from repetitive regions). An integer linear programming formulation of the Hamiltonian path problem on such graphs was found to be soluble in a matter of seconds (simulated rearrangements on *E. coli* data).

Having determined the synteny block order, the rearrangement steps that could have lead to the given static landscape must be identified. We use the most general model which considers inversions, translocations, fusions, fissions, duplications and deletions. Due to the exponential complexity arising when one attempts to reconstruct the rearrangement process with such a general model, we employ a Markov-chain Monte Carlo method to sample from the distribution of the possible rearrangement pathways. In addition to its ability to provide a high-probability solution within reasonable time limits, this stochastic approach also allows for reporting not only the most parsimonious rearrangement scenario, but rather the probability distribution of several most likely ones. This makes our approach more robust in cases with not only a single reasonable rearrange pathway.

With the dynamics of rearrangements known the genome can be compared to known cancer genomes for classification. This last step is yet to be developed.

V. A simulator

Breakpoint detection from real next-generation data is a rather difficult task. Before having arrived to our current method we evaluated several candidate techniques and beyond any doubt we will keep doing so in our further attempts to refine our framework. The analysis of a given idea is much easier if one can easily implement it with the need to care for as few irrelevant details as possible. To facilitate this rapid development we designed and implemented a simulator program.

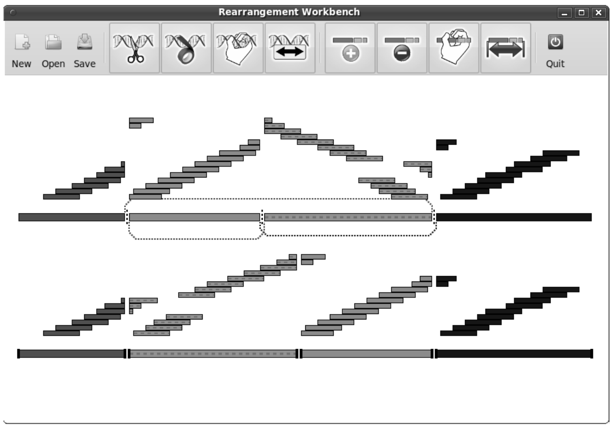


Figure 2: Screenshot of the simulator

The simulator has a graphical user interface (Figure 2) which allows the user to introduce inverted and non-inverted rearrangements to a hypothetical genome. The dual view allows for simultaneous inspection of the reference and the rearranged genome. One can map reads to the sample and see their mapping to the reference; various properties of the reads are customizable, and even mate-pair reads are supported.

The scenario set by the user can be exported to a file for the breakpoint detection software. However, the simulator allows for the implementation of such algorithms within the framework itself. This way, the input data along with the expected output is readily available for the algorithm which allows for short modification cycles, resulting in better interactive evaluation.

References

[1] M. R. Stratton et al., "The cancer genome.," *Nature*, 458(7239):719–24, 2009.

[2] E. D. Pleasance et al., "A small-cell lung cancer genome with complex signatures of tobacco exposure.," *Nature*, 463(7278):184–90, 2010.

[3] M. J. Clark et al., "U87MG decoded: the genomic sequence of a cytogenetically aberrant human cancer cell line.," *PLoS Genet.*, 6(1):e1000832, 2010.

[4] S. Volik et al., "End-sequence profiling: sequence-based analysis of aberrant genomes," *Proceedings of the National Academy of Sciences of the United States of America*, 100(13):7696–7701, 2003.

[5] B. J. Raphael and P. A. Pevzner, "Reconstructing tumor amplisomes.," *Bioinformatics*, 20 Suppl 1:i265–73, 2004.

[6] P. Medvedev et al., "Computational methods for discovering structural variation with next-generation sequencing.," *Nat. Methods*, 6(11 Suppl):S13–20, 2009.

[7] S. Sindi et al., "A geometric approach for classification and comparison of structural variants.," *Bioinformatics*, 25(12):i222–30, 2009.

[8] F. Hormozdiari et al., "Combinatorial algorithms for structural variation detection in high-throughput sequenced genomes.," *Genome Res.*, 19(7):1270–8, 2009.

[9] K. Ye et al., "Pindel: a pattern growth approach to detect break points of large deletions and medium sized insertions from paired-end short reads.," *Bioinformatics*, 25(21):2865–71, 2009.

[10] R. Bueno et al., "Second generation sequencing of the mesothelioma tumor genome.," *PLoS ONE*, 5(5):e10612, 2010.

PERFORMABILITY CONTROL OF IT INFRASTRUCTURES

Gergely János PALJAK
Advisors: András PATARICZA, Tamás KOVÁCSHÁZY

I. Introduction

Datacenter monitoring and performability control today still lack solutions that integrate well into system management solutions and provide autonomous supervisory control for such complex systems to meet the performance and dependability goals. In this paper, the problem of qualitative performance control of It infrastructures is presented. This field is today a topic of active research, especially in the context of autonomic computing [1-2, 4-5]. An approach and its evaluation on proof-of-concept system are described. In order to qualitatively predict the overall quality of service (QoS), internal performance metrics of service components are measured, and used to select the most important factors, which in turn lead to a prediction model. Depending on the prediction, reconfiguration can be applied to preserve QoS and avoid its degradation.

II. Performability control of IT systems

IT infrastructures are systems of interconnected hardware and software components that cooperate to provide business services. The processes and tools of IT system management have to assure appropriate level of dependability and performability of such systems to meet service level objectives (SLOs) of QoS metrics, while ensuring efficiency in resource utilization.

Applying resource prediction [1] and control theory (Figure 1.) with fine-grained modeling, and segment-wise linear approximations is well-known [2]. However, the relations among internal attributes are generally non-linear – especially during saturation -, from the practical point of view these approaches are not compatible with the usual way of governing IT systems by best-practice qualitative policies on higher levels.

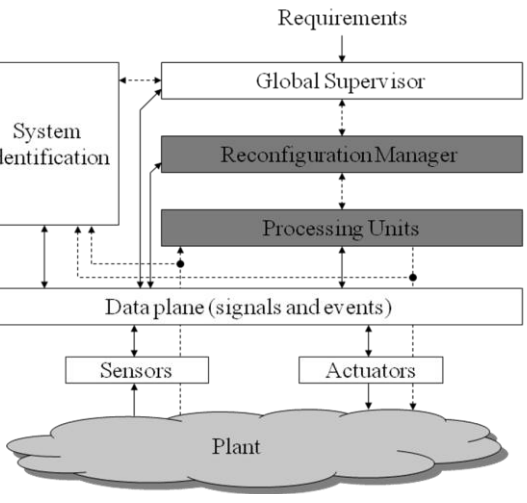


Figure 1. Reconfigurable control architecture for classical control systems

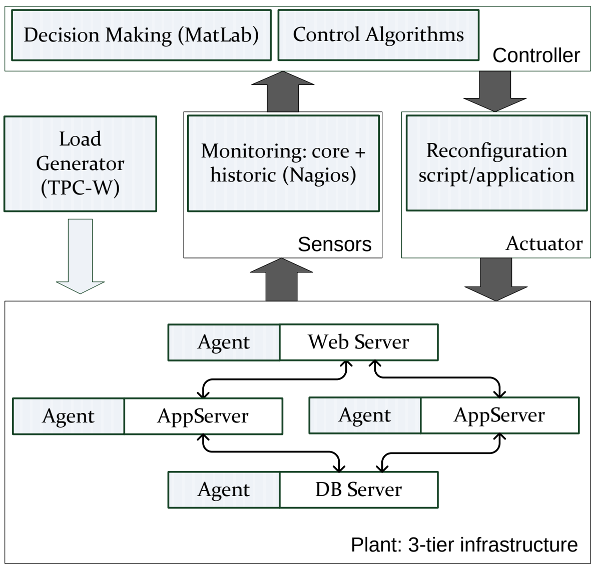
III. Pilot infrastructure and workload

In order to evaluate contemporary system-level monitoring and control techniques, a pilot application infrastructure (Figure 2.) has been built emulating a scaled-down datacenter that is integrated with monitoring and control framework where software and platform performance metrics are measured and processed in realistic scenarios.

The TPC-W benchmark [3] deployed on the infrastructure is a widely accepted benchmark for on-line transaction processing (OLTP) systems. It is described in detail in our previous work [4].

A workload profile is used that stresses the infrastructure in three operational domains. A single experiment takes two hours and is composed of 5 phases of equal duration: it starts in the 'linear' operational domain (20 parallel users), then steps to the 'degrading' operational domain (50 parallel users), then steps to the 'saturated' operational domain (70 parallel users).

The available system-internal metrics are measured and throughput as a QoS metric. This experiment is repeated to provide our training and validation datasets. 75% of our experiments is used for feature selection and to train the neural networks and 25% to validate the approach; results shown here are from 8 single experiments (16 hours). During load generation we follow the TPC-W standard; the prescribed random factors lead to a very high variance (Figure 2.).



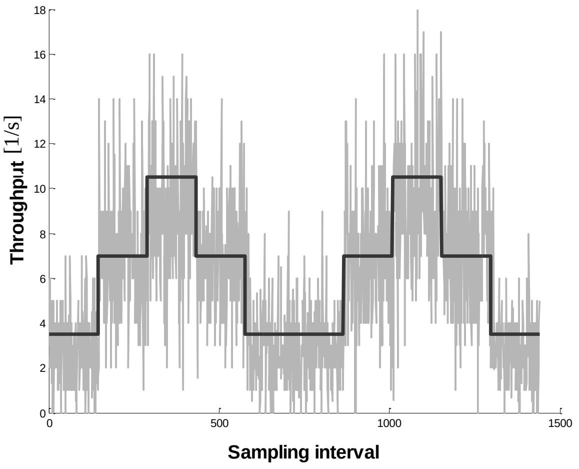


Figure 2. Experimental setup (left) and a typical generated workload (right), operational domain in black, measured throughput (transaction/s) in gray

IV. Data processing and experimental evaluation

In the data processing phase we apply a knowledge base of previous and current measurements of QoS and per-machine internal metrics to predict the system state in 1,3 and 5 minutes. The data processing approach is described in [5-6], here an outline is given.

Feature selection is performed on the measured variables, their three cycle deep temporal unfolding and the future value of the QoS metric at the temporal horizon we aim to predict – 1, 3 and 5 minutes, respectively. We select metric sets with 8 members – currently, the number of metrics to be selected is chosen manually.

As a next step, for the three prediction horizons we three feed-forward neural networks with a single hidden layer are trained with a QoS value based state classification resulting from the teaching runs. The trained networks are simulated with measurement data separated for validation. The output is post-processed, mapped to the discrete values defining the operational domain and fed to a simple decision algorithm. The decision algorithm only allows a change of operational domain prediction if outputs are equal in a certain window length aimed at reducing the stochastic noise of TPC-W, avoiding an oscillation of state changes.

Results show a prediction accuracy of 84-92% in operational domain recognition of all three time horizons, with a large proportion of inaccurate predictions originating from the degrading state, which again could be avoided by implementing a more sophisticated decision algorithm.

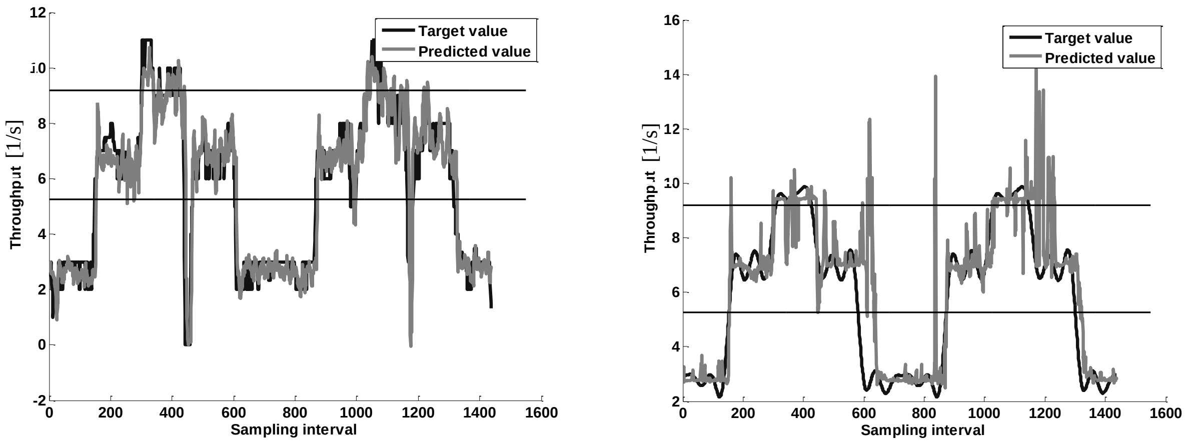


Figure 3. Prediction accuracy excerpts for two different experiment runs

V. Shortcomings of current system monitoring best practices

We examined how to enhance state prediction accuracy and have identified several flaws of widely used; state-of-the-art system monitoring frameworks (both open source and commercial):

A Centrally initiated sampling

In typical monitoring systems, measurements are initiated by a central monitoring server by sequentially sending out requests to a set of nodes (software agents). The agents carry out the measurement upon receiving the request and send back the result. This technique results in the skew of measurement data, and problem this grows with size of the supervised system (note: in our small, 4 node setup, we measured ~160 attributes of the possible thousands).

We propose that measurement times and intervals should be pre-negotiated between the server and each agent, then at every sampling interval, the agent carries out the measurement (timing provided by a globally synchronized clock) and sends the measurement value and the timestamp to the server. The monitoring server should only initiate the pre-negotiation, and receive the pushed metric values.

B Inappropriate handling of timestamps

Every measurement value must be accompanied by a timestamp of the exact time of the measurement. Though this seems to be a basic criterion, many monitoring systems are flawed even in this aspect as they do not differentiate between the time of the measurement (on a remote machine, done by a specific agent) and the time when the value is stored in a central data store. Due to network lags, or simply because of the fact that the monitoring server has to queue the incoming measurement values this can be severely different, resulting in skew in the timeseries.

We advocate timestamping by the agent to ensure timeliness, with appropriate precision, and storing the recorded timestamps with proper precision (preferably 64 bits).

C Lack of synchronization

Sampling in a distributed system usually does not happen in a synchronized way. This results in skewed timeseries, incoherent system state estimations. To employ control theory techniques synchronous sampling is essential; processing skewed data streams necessitates interpolation, naturally resulting in loss of precision.

Today computer clocks are typically synchronized by the Network Time Protocol (NTP), this protocol keeps the offset of the clocks in a LAN-connected distributed system in the range of 10-20 milliseconds, this precision is being further enhanced by the IEEE 1588 Precision Time Protocol (PTP) currently gaining ground and offering synchronization in the microseconds range.

We emphasize that sampling should occur according to a globally synchronized clock, not in a request-response way.

D Very long sampling intervals

Most IT monitoring systems are comfortable with a 5-15 minutes interval for sampling, by straining the software to its very limits this can be reduced to 0.5-2 minutes. It is still clearly inadequate for representative sampling and autonomous performability control. However, central data collectors rarely are ready for this kind of workload, and can easily be overloaded.

We recommend implementing intelligent agents capable of storing for short term and aggregating monitoring information to reduce the stress on the network and central data stores.

E Computation intensive sampling

In operating system-level metrics it is a fairly cheap operation to sample a performance metric. However, this is usually not the case for middleware or application level metrics: the monitoring instrumentation of many software components is so slow and resource intensive, that polling is not only infeasible in the some hundreds of milliseconds, but even in the tens of seconds range.

Applications should be designed with ease of management and maintainability in mind, exporting well-designed monitoring and actuation interfaces with efficient instrumentation.

F Ambiguous metric definitions, documentation

System monitoring frameworks have a notorious problem with defining metrics: measurement units, time of validity, update frequency are documented ambiguously if at all. Many performance indicators are implemented as counters, these can turn around and need to be reset from time to time, even widely deployed monitoring solutions fail to handle these situations. A usual source of performance metrics is the Linux proc filesystem: the documentation is at numerous points outdated or missing with respect to the latest kernels. [7]

VI. Conclusion and future work

During our work in designing and implementing supervisory systems for IT infrastructures, we have learnt that prediction and thereby prediction-based control, resource allocation is possible; with the current, highly complex cloud computing systems only model-based control seems to be an efficient solution. In order to achieve these goals, first the issues with state-of-the-art monitoring must be addressed. As future work, we plan to elaborate a blueprint for an efficient monitoring system with correct sampling for large-scale systems and develop further our supervisory algorithms.

References

[1] J. Hellerstein, Y. Diao, S. Parekh, and D. Tilbury, *Feedback control of computing systems*, IEEE, 2004

[2] G. a Hoffmann, K.S. Trivedi, and M. Malek, “A Best Practice Guide to Resource Forecasting for Computing Systems,*” IEEE Transactions on Reliability*, vol. 56, 2007

[3] TPC-W official website, http://www.tpc.org/tpcw/default.asp

[4] G. J. Paljak, "Infrastructure for Model-based Control of Distributed IT systems”, *17th BME MIT PhD Minisymposium*, 2010

[5] G. J. Paljak, Z. Égel, D. Tóth, I. Kocsis, T. Kovácsházy, A. Pataricza, "Qualitative Performance Control in Supervised IT Infrastructures", *DSN PFARM*, 2010

[6] G. J. Paljak, I. Kocsis, Z. Égel, D. Tóth, A. Pataricza, „Sensor Selection for IT Infrastructure Monitoring”, *AUTONOMICS*, 2009

[7] Linux PROC File system documentation, http://kernel.org/doc/Documentation/filesystems/proc.txt

OPTIMIZING SATURATION BASED MODEL CHECKING

András VÖRÖS
Advisor: Tamás BARTHA

I. Introduction

Formal methods are widely used for the verification of safety critical and embedded systems. However, nowadays the use of formal methods is not limited to these areas as the demand for proven functional correctness increased in many other fields, e.g. in telecommunication and automotive industry. The main advantage of formal methods compared to extensive testing is that they can provide a mathematical proof for the correct behavior of the system, or they can prove that the system does not meet its specification. On the other hand, testing can only examine a portion of the possible behaviors. Usually the system undergoes testing after the implementation phase; formal methods can help finding bugs and design errors earlier, in the design phase. This makes the correction cheaper, and the development process more effective.

Nevertheless, formal verification is a complex task. As systems become more complex, the excessively growing number of possible behaviors of them leads to enormous memory or time consumption, so that the completion of the analysis process becomes impossible due to the limitations of the computational architecture. This is called the state space explosion problem.

One of the most prevalent techniques in the field of formal verification is model checking, which is an automatic technique to check whether a system fulfills the requirements. There are several model checking algorithms and approaches, starting from explicit traversal through the use of different reduction techniques and ending at the abstract methods. However, all methods need to generate a representation of the state space in order to run some analysis on it. Storing the state space representation can turn out to be quite difficult in cases where the state space explodes.

There are two main reasons behind this problem. The first reason is that the independently updated state variables lead to exponential growth in the number of the system states, even in the case of Boolean variables. The second reason is the asynchronous characteristic of distributed systems; the composite state space is often the Cartesian product of the local components' state spaces.

There are different ways to efficiently handle state space explosion. We can reduce the memory and time consumption with intelligent state traversal – we only traverse and store a representative set of the state space, which preserves the examined properties.

To overcome the difficulties caused by the large number of state variables, the use of symbolic methods is widespread. These algorithms can avoid storing state space explicitly; instead they encode the states and store them implicitly for example in decision diagrams. Symbolic methods enable us to efficiently store huge sets in memory. In general: symbolic techniques were a good choice at the register transfer level (RTL) design, where the large number of variables lead to large memory consumption. At system level, asynchronous components lead to large state spaces, dealt with reduction. This also shows the different strengths of the algorithms. In this paper I introduce a special symbolic model checking algorithm, called saturation, and I examine the effect of some heuristics to its performance.

II. Symbolic state representation and saturation

Traditional symbolic model checking use encoding to store the traversed state space, and stores this compact representation only. Decision diagrams proved to be a proper solution for this purpose, as applied reduction rules provide a compact representation form. In the early times Binary Decision Diagrams were the most common, but in the last decade many other variants appeared to comply with the broadening application requirements. From this wide area I introduce Multiple Valued Decision Diagrams (MDD), which I present in the remaining of the paper.

Multiple Valued Decision Diagram is a directed acyclic graph with a node set containing 2 types of nodes ordered into levels: non-terminal and 2 terminal vertices. A non-terminal vertex is labeled by a variable index k, which indicates to which level belongs the node (which variable is represented by it), and has n_k (domain size of the variable, in binary case n_k=2) arcs pointing to nodes in level k-*1*. Duplicate nodes are not allowed, so if in level k two nodes have identical successors, they are also identical. Redundant nodes are allowed, so it is possible that a node's all arcs point to the same successor. These rules ensure that MDD-s are canonical representation of a given function or set.

The first step in symbolic model checking is to encode the possible states. Traditional approach encodes each state with a given variable assignment (v_1, v_2... v_n) and stores it in a decision diagram. To encode the possible state changes, we have to encode the transition relation, which can be done in a $2n$ level decision diagram with variables: (v_1, v_2 ...v_n, v'_1, v'_2...v'_n), where the first n variables represent the "*from*", and variables (v'_1, v'_2...v'_n) the "*to*" states. The state space traversal builds this during a breadth first search, or in a combined breadth first and depth first traversal called chaining.

The main motivation of saturation state space exploration [1] is to combine symbolic methods with a special iteration strategy, which proved to be very efficient for asynchronous models. These models consist of some components interacting with each other less often than executing their local transitions. In this case, we can decompose the model to local sub-models, and the global state is the composition of the components' local states: $s_g = (s_1, s_2,..., s_n)$, where n is the number of components. This decomposition is the first step of the saturation algorithm. Saturation needs the so called *Kronecker* consistent decomposition, which means that the global transition (*Next-state*) relation is the Cartesian product of the local-state transition relations. Formally: if $\mathcal{N}_{i,e}$ is the *Next-state* function of the transition (event) e in the i-th sub-model, the global *Next-state* of event e is: $\mathcal{N}_e = \mathcal{N}_{1,e} \times \mathcal{N}_{2,e} \times ... \times \mathcal{N}_{n,e}$. Note that when modeling asynchronous systems, a transition usually affects only some or some parts of the sub-models. This kind of event locality can be easily exploited with this decomposition. Many modeling languages, such as Petri nets have this property [1].

During saturation, decomposition serves the basic of the symbolic encoding – we code in a variable a sub-model's local states. We need as many variables as many sub-models we divided the model into. Opposite to the traditional approach, the *Next-state* function is not coded explicitly in a *from-to* relation in the MDD, instead we store to each sub-model the local *Next-state* function in the so called *Kronecker* matrix [2]. As the transitions usually have only local effects, 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, the top (top_e) and bottom (bot_e) levels (sub-models). Outside this interval we don't deal with it. *Kronecker* matrices and local state spaces are built dynamically during the traversal, while affected level intervals of events (top_e, bot_e) are defined before, from structure.

Saturation is a special iteration strategy, which consists of a bottom-up building of the decision diagram with a special depth first traversal of the state space. The main aim is to explore the sub-models local state space in a greedy manner, and encode them before other sub-models are explored. This is similar to the chaining method; the main difference is that saturation localizes the effects of the changes in the MDD, updating nodes instead of updating the whole MDD.

The reachable set of states $\mathcal{S}$ from a given initial state s is the closure of the *Next-state* relation: $\mathcal{S} = \mathcal{N}^*(s)$. Saturation iterates through the MDD nodes and reaches the whole state space representation from node to node transitive closure generation. In this way saturation avoids that 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 ε be the set of events affecting k. level and below, so where $top_e \leq k$. We call a node p at level k saturated, if node $\mathcal{B}(k,p) = \cup_{\varepsilon} \mathcal{N}_e^*(\mathcal{B}(k,p))$. The state space generation ends when the node at the top level becomes saturated, so it represents $\mathcal{S} = \mathcal{N}^*(s)$ for all events.

As saturated nodes represent the local fixed-points for events having effects localized to the sub-graph, only saturated nodes can appear in the final state space representation [2]. In addition, updating nodes during the traversal can be made locally, called in-place update, just setting the edges to the newly explored sub-MDD. So we need fewer operations affecting the whole MDD than traditional approaches, instead we make operations with smaller impact. This property helps keeping the peak size of the MDD low during the state space exploration.

State space generation serves as a prerequisite for the structural model checking: verifying temporal properties needs the state space and transition relation representation. CTL is widely used to express temporal specifications of systems, as it has expressive syntax and there are efficient algorithms for its analysis. In CTL, operators occurs in pairs: 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, from the 8 possible pairings due to the duality [1], we only need to implement 3: *EX*, *EU*, *EG*, and we can express the remaining with the help of them. These operators also benefit from the locality exploited by saturation. The semantics of them are:

- ***EX***: $i^0 \models EX\ p$ iff $\exists\ i^1 \in \mathcal{N}(i^0)$ s.t. $i^1 \models p$ ("|=" means "satisfies"). This means that *EX* corresponds to the inverse $\mathcal{N}$ function, applying one step backward through the *Next-state* relation, using transposed *Kronecker* matrix. This way we can exploit locality efficiently.
- ***EG***: $i^0 \models EG\ p$ iff $\forall n \geq 0, \exists i^n \in \mathcal{N}(i^{n-1})$ s.t. $i^n \models p$ so that there is a strongly connected component containing states satisfying p. This computation needs a greatest fixed-point computation, so that saturation cannot be applied for it. Computing the closure of this relation however profits from the locality accompanying the decomposition.
- ***EU***: $i^0 \models E[pU\ q]$ iff $\exists n \geq 0, \exists i^1 \in \mathcal{N}(i^0), ..., \exists i^n \in \mathcal{N}(i^{n-1})$ s.t. $i^n \models q$ and $i^m \models p$ 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, which can exploit the efficiency of saturation.

However, before performing saturation in *EU*, we have to classify events into categories in order to define the breadth first and the saturation based steps in the fixed-point calculation:

- An event *e* is **dead** with respect to a set of states *S* if $\mathcal{N}_e^{-1}(S) \cap S = \emptyset$, these events are omitted from the fixed-point calculation.
- An event *e* is **safe**, if it cannot lead from outside *S* to sates in *S*, formally: $\emptyset \subset \mathcal{N}_e^{-1}(S) \subseteq S$.
- All other events are **unsafe**.

With the help of this categorization, we decompose the fixed-point calculation into 2 steps:

- Computing the closure of relations of the safe events can be efficiently done by saturation
- By breadth-first traversal the algorithm explores unsafe events. As from states reached by unsafe steps we have to filter out those, which do not satisfy *p* or *q*, we have to compute the intersection of them with $p \cup q$. This intersection is evaluated in all iteration.

The efficiency of *EU* computation highly depends on the efficiency of the saturation steps, because the number of breadth first steps (and intersection operations) depends on the model and the temporal logic formula itself, so we can only reduce the runtime of the saturation.

Time and memory consumption of saturation depends on the iteration order, and of course on the size of the MDD. Measurements showed that the size of the *Next-state Kronecker* matrices is just a small fraction of the whole memory required by the algorithms.

Iteration order and MDD size are not independent. Actually, the algorithm iterates through all nodes, so the complexity is at least the number of nodes in the MDD. We have to point out that there are more iterations, usually by a polynomial factor. As the size of the MDD can be exponential in the size of the variables, this is a critical point in symbolic model checking. Good variable ordering is the key in MDD size reduction. Finding good variable order is an NP hard problem, however there are usable heuristics that perform well.

III. Optimization

The main motivation of the optimization is to enable the algorithm to exploit locality efficiently, which is important not only because of the number of the iteration steps, but the size – and of course, the efficiency – of the MDD operations. In addition, the cached values of the event-node pairs – which enable the algorithm to avoid redundant operations on MDD nodes – couldn't be used efficiently, when the iteration order, so that the MDD variable order doesn't reflect the structure of the system. Optimizing saturation consists of the decomposition of the model into sub-models, and then the algorithm assigns an MDD variable to each of them. Good decomposition, i.e. when functionally dependent sub-models are composed into the same variable can reduce the size of the MDD [2]. However, coding big state spaces into a single variable leads soon to an unusable algorithm as it converges to the explicit state space representation method.

There are two main trends in order to improve the efficiency of the saturation iteration strategy [2]: choosing the ordering of the variables to make the effects of the transitions locally or to make the top levels of the transitions effects lower. First approach improves locality by trying to reduce the distance between the top and bottom levels of the transitions, so that we minimize the sum of transition spans. Second approach uses the fact that saturation builds the MDD representing $\mathcal{S}$ in a bottom-up fashion, so placing transitions as low as possible is the preferable. This approach minimizes the sum of the transition tops. Combining the approaches is a good compromise.

The literature about MDD size reduction is wide. These approaches use some kind of variable reordering method. The basic operation is the so called "adjacent level interchange", which changes the order of the adjacent variables, trying to reduce their sizes. This way we can reduce the size of the MDD step-by-step, and avoid overrunning the memory limit. The optimization algorithms perform these steps until a good variable order is found. However, in this context, this method can only be used after the state space generation, so it cannot make saturation faster.

Reducing the size of the MDD needs some heuristics to find a good variable order before the generation. If we use some knowledge about the model it can help us to avoid mistakes which enlarge the MDD. It is well known, that if there are concurrent sub-models, the variables representing their states shouldn't be overlapped. Consider 2 components with variables x_1, x_2 and y_1, y_2, where functional dependencies ($\rightarrow$) occur inside the components. In this case there is no functional dependence: $(x_1, x_2) \rightarrow (y_1, y_2)$, only $(x_1) \rightarrow (x_2, y_1, y_2)$ and $(x_1, x_2, y_1) \rightarrow (y_2)$. Using the variable order x_1, y_1, x_2, y_2, 2 dependencies remain: $(x_1) \rightarrow (y_1, x_2, y_2)$ and $(x_1, y_1, x_2) \rightarrow (y_2)$, however we create a new dependence: $(x_1, y_1) \rightarrow (x_2, y_2)$, which will increase the number of nodes. Let me point out, that these functional dependencies mean some transitions in the model checking context, which control the possible reachable states, so the heuristics used for the iteration order indeed localizes the dependencies in the MDD making it smaller.

Using heuristics from [2] in the structural CTL model checking may improve the performance. As *EU* and *EF* computation builds mostly on saturation, they can exploit heuristics after the state space generation. However the algorithm was optimized also before state space generation, applying the heuristics during the model checking, there were a 3-5% additional performance gain. Applying MDD size reduction before *EG* computation decreased the computation time with 5-50%.

IV. Conclusion, further work

The usage of static variable ordering in CTL model checking proved its usefulness. In the future I would like to extend the algorithms to apply dynamic reordering during saturation.

References

[1] Ciardo, G., R. Siminiceanu.: "Structural symbolic CTL model checking of asynchronous systems." *CAV,* 2003

[2] Ciardo, G., G. Lüttgen, A.J. Yu.: "Improving static variable orders via invariants." *ICATPN 2007,* Springer