IEEE HUNGARY SECTION
CIRCUITS, SYSTEMS AND
COMPUTERS JOINT CHAPTER

John von Neumann
Computer Society
Neumann János
Számítógép – tudományi Társaság

# PROCEEDINGS OF THE 16TH PHD MINI-SYMPOSIUM

**FEBRUARY 2, 2009**

MŰEGYETEM 1782

BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
FACULTY OF ELECTRICAL ENGINEERING AND INFORMATICS
DEPARTMENT OF MEASUREMENT AND INFORMATION SYSTEMS

Head of the Department: Prof. Dr. Gábor Horváth

Conference Chairman:
Béla Pataki

Organizers:
Gábor Bergmann
György Dancsi
Gergely Hajós
Tamás Krébesz
Tamás Raikovich

Homepage of the Conference:
http://www.mit.bme.hu/events/minisy2009/

Sponsored by:
IEEE Hungary Section
Circuits, Systems and Computers Joint Chapter

John von Neumann Computer Society

evosoft Hungary Kft.

NI Hungary Software és Hardware Gyártó Kft.
National Instruments Hungary Kereskedelmi Kft

Special supporter:

evosoft

ISBN 978-963-420-972-0

# FOREWORD

This proceedings is a collection of the extended abstracts of the lectures of the 16th 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. 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 three main groups. These are Embedded 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 sixteen-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 IBM Budapest CAS; IBM Zürich Lab; Intel; Fujitsu-Siemens München; Toshiba R&D Center, Kawasaki, Japan; TU Darmstadt, Germany; TU Ilmenau, Germany; University of Firenze, Italy; Computer and Automation Research Institute of the Hungarian Academy of Sciences, Budapest; Robert Bosch Kft., Hungary, Knorr-Bremse Fékrendszerek Kft, Research and Development Institute, Budapest; National Instruments Hungary Software és Hardware Gyártó Kft., Debrecen; National Instruments Hungary Kereskedelmi Kft., Budaörs.

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, 2009

Béla Pataki
Chairman of the PhD Mini-Symposium

**evosoft Hungary Kft.**

**Why do we sponsor this program?**

evosoft devotes exceptional care to higher level education institutes, particularly to BME, as our main recruiting base: besides offering diploma work and apprentice programs for students we have been supporting professional forums such as Simonyi conference or this year presentation series with Technical College. In addition we have been giving organizational support or sponsoring other events as well - e.g. Gyűrűavató Szakest, Schönherz Qpa, Csillagtúra, or the autumn Beköltöző Hónap. evosoft have been co-operating with more departments of BME in different fields of education for many years: on the one hand we do our continuous vocational education of our employees arm in arm with the university, on the other hand many of our colleagues give lectures at the university. Currently we have a lot of colleagues owning Ph.D. or are doctoral students, and we do our best to provide the circumstances to assist their personal research and further publications – beside the daily work.

**evosoft Introduction**

evosoft Hungary Kft., founded in 1995 – as a subsidiary company of Siemens AG, Germany – is currently one of the biggest and most stable software houses in Hungary. Besides, it is the market leader in software export towards Germany. Nearly 500 people work on Budapest and Miskolc, developing high quality industrial and enterprise software.

Highly educated professional groups work on our cross country projects with the most advanced technology. Or company business covers the entire software development process, including system specification, design, implementation and operational support, and associated software consultancy as well.

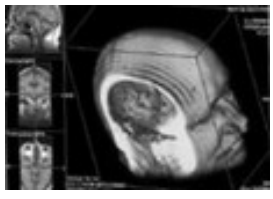
**Field of Business**

Software consultancy, software design, software implementation (product and system implementation), software integration, system test, web-based service (WBS), Application Management Services, SAP consultancy.

**Departments**

Healthcare Platform(image diagnostic), Automation and Drive Technologies, Energy Automation, Drive Technologies, ERP, Business Development, Industry Automation, Mobility/IS.



# LIST OF PARTICIPANTS

| Participant | Advisor(s) | Starting Year of PhD Course |
|---|---|---|
| BERGMANN Gábor | VARRÓ Dániel | 2008 |
| DANCSI György | FEHÉR Béla | 2008 |
| GARAMVÖLGYI Zsolt | BANK Balázs and SUJBERT László | 2008 |
| HAJÓS Gergely | ANTAL Péter and DOBROWIECKI Tadeusz | 2008 |
| HORVÁTH Ákos | VARRÓ Dániel | 2006 |
| JUHÁSZ Sándor | HORVÁTH Gábor and Atsushi Sugahara | 2007 |
| KOCSIS Imre | PATARICZA András | 2006 |
| KOLLÁR Zsolt | PÉCELI Gábor and Reiner Thomä | 2008 |
| KRÉBESZ Tamás | KOLUMBÁN Géza | 2007 |
| OROSZ György | PÉCELI Gábor and SUJBERT László | 2006 |
| PECHAN Imre | FEHÉR Béla | 2008 |
| RAIKOVICH Tamás | FEHÉR Béla | 2007 |
| RÁTH István | VARRÓ Dániel | 2006 |
| SCHERER Balázs | HORVÁTH Gábor | 2007 |
| SZATHMÁRI Marcell | HORVÁTH Gábor | 2008 |
| SZATMÁRI Zoltán | MAJZIK István | 2008 |
| SZOMBATH István | PATARICZA András | 2008 |
| TÓTH Dániel | PATARICZA András | 2006 |

# PROGRAM OF THE MINI-SYMPOSIUM

**Model-based Software Engineering I.**



**Embedded and Intelligent Systems**



**Model-based Software Engineering II.**



**Measurement and Signal Processing**

# CONFERENCE SCHEDULE

| | |
|---|---|
| 08:45 | Opening |
| 09:00 | Model-based Software Engineering I. |
| 10:30 | Embedded and Intelligent Systems |
| 11:30 | Lunch break |
| 13:00 | Model-based Software Engineering II. |
| 15:00 | Measurement and Signal Processing |

# PARALLELIZATION OF INCREMENTAL PATTERN MATCHING IN GRAPH TRANSFORMATION

**Gábor BERGMANN**
**Advisor: Dániel VARRÓ**

## I. Introduction

Nowadays, desktop computers are often equipped with multi-core processors, and single-threaded execution does not take advantage of this increased computational capacity. It is a great challange in the industry to find algorithms that are scalable and capable of exploiting the power of modern computing architectures. Model transformation is an application domain that could benefit greatly from parallelizations (along with other improvements to efficiency), as processing large-scale models often suffer from permormance issues.

Using a graph transformation [1] based approach for model transformations, there are even more possibilities for the exploitation of parallelism. Besides model manipulation sequences, graph transformations involve a graph searching phase, which is targeted at finding the matches of a graph pattern. Nevertheless, graph transformation tools rarely exploit parallel execution.

Previous work revolved aroung improving pattern matching performance by employing an incremental strategy [2] based on the RETE [3] algorithm. Incremental techniques store the occurrence set of graph patterns so that they are always immediately available, and update these caches upon model changes. RETE in particular stores the match set of subpatterns also. A RETE net consists of cache nodes, each responsible for maintaining the match set of a subpattern, and update channels between these nodes. Changes in the model or subpattern caches trigger messages flowing in these channels to prompt the incremental update of the cache stored at the recipient node.

This paper examines ways in which RETE-based pattern matching could benefit from parallelism.

## II. Concurrent pattern matching and model manipulation

Contrary to previous work, the RETE net implementation used throughout this paper relies on *asynchronous* message passing. Using asynchronous messaging, the load on the main thread of the transformation can be reduced by executing the incremental pattern matcher (which consumes change messages from the queue) in a separate thread. When the transformation manipulates the model (see Fig. 1), it only has to send the new update message to the message queue, and continue its operation. The thread of the pattern matcher will execute the update propagation in the background, ideally, without imposing a performance penalty on the transformation thread. When the message queue becomes empty, the RETE network has reached steady state; the pattern matcher thread then goes to sleep and will not resume its operation until a new update message is posted.

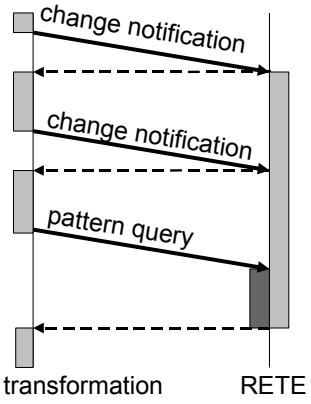



Figure 1: Separate pattern matcher thread with concurrency and waiting

When the transformation initiates pattern matching, it has to assure that background update propagations have terminated and the matches stored at the production nodes are up-to-date. If the network has not yet reached its fixpoint, the model manipulation thread will have to sleep until that happens.

**Performance expectations.** While the local search based pattern matchers operate with cheap model changes and costly pattern queries, a sequential RETE-based matcher [2] relies upon a moderate overhead on model change balanced by instant pattern queries. This novel concurrent incremental pattern matching approach combines the advantages of the former two: it has cheap model manipulation costs, and potentially instant pattern queries. Although the transformation might have to wait for the termination of the background pattern matcher thread, the worst case of this time loss is still comparable with the update overhead of the original RETE approach.

## III. Multi-threaded pattern matching with RETE

The concurrent pattern matching approach can be improved further by parallelizing the update propagation phase. Here I present a simple solution. The basic idea is to employ multiple pattern matcher threads to consume update messages. The proposal splits the network into separate RETE *container*s, each of which is responsible for matching a set of subpatterns. Each container has its own distinct set of nodes, and a dedicated pattern matcher thread consuming update messages of a dedicated queue. Each container is responsible for forwarding messages to its nodes using the dedicated message queue. Forwarding messages between two containers is accomplished by enqueueing the message in the target container. Fig. 2 depicts a multi-threaded pattern matcher illustrating how a RETE net can be split into several containers for parallel execution.

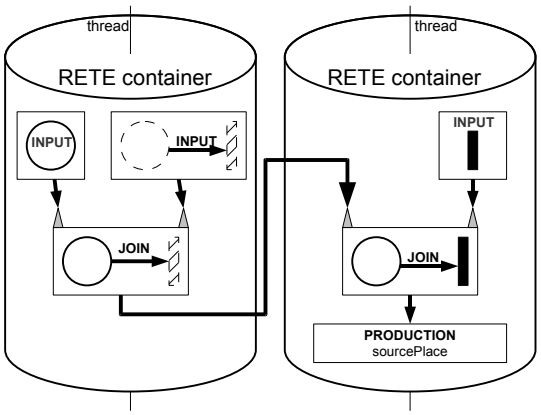



Figure 2: Multiple containers

**Performance expectations.** An ideal application scenario would be several parallel transformations that are known to use different patterns; allowing straightforward splitting and parallelization of the RETE net, with a low amount of inter-connectedness. By partitioning the patterns into relatively independent containers, a multi-threaded RETE pattern matcher may achieve high performance.

## IV. Conclusion

I have designed ways of parallelizing a RETE-based incremental graph pattern matcher and implemented them as part of the VIATRA2 framework [4]. This approach supports large-scale model transformation and complements the parallelization of transformation execution. Future work is required to carefully measure performance in various problem classes, and to fine-tune the implementation.

## References

[1] H. Ehrig, G. Engels, H.-J. Kreowski, and G. Rozenberg, Eds., *Handbook on Graph Grammars and Computing by Graph Transformation*, vol. 2: Applications, Languages and Tools, World Scientific, 1999.

[2] 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.

[3] C. L. Forgy, "Rete: A fast algorithm for the many pattern/many object pattern match problem," *Artificial Intelligence*, 19(1):17–37, September 1982.

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

# GRAPH TRANSFORMATION BASED CONSTRAINT SOLVING

**Ákos HORVÁTH**
**Advisor: Dániel VARRÓ**

## I. Introduction

Constraint satisfaction is a fundamental Artificial Intelligence technique for knowledge representation and reasoning. It has, however, become clear that in many cases the original formulation of static constraint problems (CSP) [1] with imperative constraints is insufficient to model real problems (dynamic allocation, change in the constraint set, etc.) Research has been already carried out to address these shortcomings in the form of two separate extensions known as flexible [2] and dynamic CSP [3] and recent work is focusing on the combination of these separate extension [4].

In this paper we introduce our novel approach, which uses graph patterns and graph transformation (GT) [5] rules to extend the definition of constraint satisfaction problems. The idea is to define the constraints and the labeling rules as graph patterns and transformation rules, respectively treat the model modified by the rules as the only variable of the CSP, where a result is achieved when no matches are found for any constraint pattern. This approach allows to (i) dynamically add/remove constraints from the problem domain, (ii) modify the domain of the variables and (iii) define structural constraints in a more natural way.

The rest of the paper is structured as follows. In Sec. II. we briefly introduce the concept of metamodeling, graph transformation and constraint satisfaction problems. Sec. III. proposes our graph pattern and transformation based constraint solver. Finally, Sec. IV. concludes the paper.

## II. Background

In order to introduce our approach this section briefly outlines the basics of metamodeling, graph transformation and constraint programming.

### *A. Models and Metamodels*

Metamodeling is a fundamental part of model transformation design as it allows the structural definition (i.e., abstract syntax) of modeling languages. More precisely, for the definition of a modeling language the followings shall be given (i) the abstract syntax defining the concepts of the given domain and their relations, (ii) the concrete syntax defining the textual or graphical notations of the concepts, (iii) wellformness rules defining further constraints for the concepts, (iv) the formal semantics defining the dynamic behaviour of the models.

In our approach, we use a unified directed graph representation as the underlying model of the VIATRA2 [6] framework. This way, graph nodes (called entities in VIATRA2) uniformly represent MOF packages, classes, or objects on different metalevels, while graph edges with identities (called relations in VIATRA2) denote MOF association ends, attributes, link ends, and slots in a uniform way. Essentially, nodes represent basic concepts of a (modeling) domain, while edges represent the relationships between model elements. An example metamodel is depicted in Fig. 1 representing a simple allocation problem domain, where `Simple` and `Complex` jobs can be allocated to `Node`s.

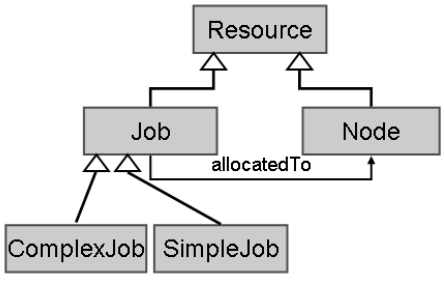
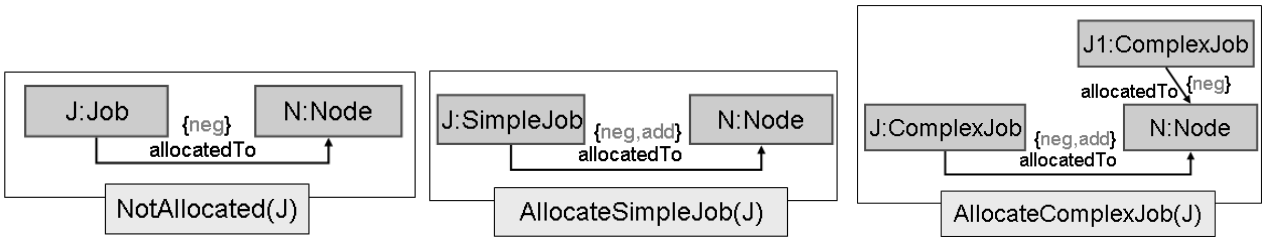



Figure 1: Sample Metamodel




(a) Not allocated `GT pattern` (b) Allocate Simple Job `GT rule` (c) Allocate Complex Job `GT rule`

Figure 2: Sample graph pattern and transformation

### *B. Graph Transformation Rules and Patterns*

Graph transformation (GT) is a rule and pattern-based paradigm frequently used for describing model transformation. A graph transformation rule contains a left-hand side graph LHS, a right-hand side graph RHS, and (one or more) negative application condition graphs NAC connected to the LHS. A negative application condition is a graph morphism, which maps the LHS pattern to a NAC pattern. In other terms, the LHS and NAC graphs together denote the precondition (also referred to as GT pattern) while the RHS denotes the postcondition of a rule.

The application of a rule to a host (instance) model M replaces a matching of the LHS – provided it is not invalidated by a matching of the NAC, which prohibits the presence of certain nodes and edges – in M by an image of the RHS.

Sample GT pattern and rules defined over the metamodel described in Fig. 1 are depicted in Fig. 2, using a combined representation that jointly defines the left hand side (LHS) of the graph transformation rule, and the actions to be carried out. The `NotAllocated` pattern matches an input parameter `Job J` which is not already `allocatedTo` to a `Node N` (the NAC is highlighted by the `neg` adornment). That the `AllocateSimpleJob` and `AllocateComplexJob` represent GT rules that allocate a `Simple` or `Complex` job to a `Node`, respectively. The additional NAC guarantees that only a one `Complex` job can be allocated to a single `Node`. In both cases the `add` adornment defines the element to be created.

### *C. Constraint Satisfaction*

Basically, a CSP is a problem composed of a finite set of variables, each of which is associated with a finite domain, and a set of constraints that restricts the values the variables can simultaneously take. In a more precise way a constraint satisfaction problem is a triple: $(Z, D, C)$ where

- $Z$ is a finite set of variables $x1, x2, ..., xn$;
- $D$ is a function which maps every variable in $Z$ to a set of objects of arbitrary type.
- $C$ is a finite (possibly empty) set of constraints on an arbitrary subset of variables in $Z$. In other words, $C$ is a set of sets of compound labels.

The task is to assign a value to each variable satisfying all the constraints.

## III. Overview of the Approach

This section first, gives an overview of the approach using our running resource allocation example, and then summarizes the advantages and disadvantages

### *A. Resource Allocation*

In general the aim of a resource allocation problem is to determine the allocation of a fixed amount of resources to a given number of activities in order to achieve an effective results. Considering the metamodel described in Sec. A. our task is to allocate `Complex` and `Simple` jobs to `Node`s with a restriction that one `Node` can have arbitrary number of `Simple` but only one `Complex` job. Without

going into details in order to formulate the problem using GT rules and patterns we have to define the following artifacts:

- A typed graph *model* – conforming to a given metamodel – that represents the initial state of the CSP. During the solution process it is modified using *labeling* literals to achieve a concrete state that satisfies all given `goals`, where a *state* of the solution process is a concrete state of the *model*.
- *Goals* (constraints) are represented by GT patterns over the given metamodel. A *goal* is satisfied if no match is found to its representing GT pattern. For a given problem arbitrary number of *goals* can be defined.
- *Labeling* literals are GT rules defined over the given metamodel used to manipulate the *model*. During the solution processa following *state* is achieved by a random selection from the applicable GT rules and its invocation on the *model*. A solution process ends if a solution is found or if all possible *states* are traversed or the depth of the state space tree is higher than an initially given number. This last restriction is necessary to ensure that the process will definitely terminate as in general termination of graph rewriting is undecidable.

We formalized the running example constraint problem with the GT pattern and rules depicted in Fig. 2. The `NotAllocated` pattern defines the *goal* of the problem, namely, that all `Jobs` have to be allocated to a `Node`, which is equivalent to that there is no match found for the pattern. The *labeling* literals are captured by the `AllocateSimpleJob` and the `AllocateCompleyJob`. For an initial model we use the one depicted in Fig. 3. A possible solution is that `CJ1` and `CJ2 ComplexJobs` are allocated to the `N1` and `N2 Nodes`, respectively, while both `SimpleJobs` are allocated to one of the `Nodes`.

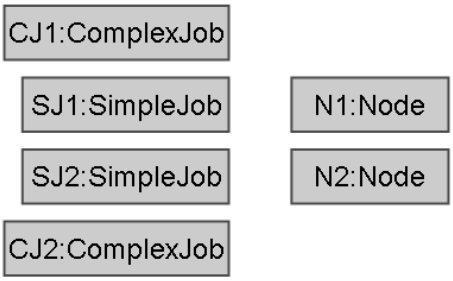



Figure 3: Sample instance model

*B. Advantages and Drawbacks*

The proposed approach promises a number of benefits but also has drawbacks that have to be carefully analyzed in order to use it in its full potential. In the following we try to address some of these issues:

- One of the main advantage of using GT rules for the labeling is that it allows to define **dynamic creation/deletion** rules on any element (defined in the metamodel) of the problem domain. Considering our running example the GT rule depicted in Fig. 4 gives dynamic `Node` creation for a `ComplexJob` that is not allocated. However this kind of dynamic behavior can only be effectively used if proper priorities are defined to the labeling literals and handled during the solution process as these kind of rules can easily run into nonterminational graph rewriting sequences.
- Our approach not only allows to dynamically create and delete elements of the problem domain, but also allows to **dynamically add/remove** *goals* and *labeling* literals to alter the constraint problem. This feature helps to address problems defined in DCSP [7].
- **Flexible constraint** satisfaction can also be easily adapted by defining weights for each goal. In this case a solution is achieved if the accumulated weight of the unsatisfied goals is lower than a predefined limit (weighted CSP [4]). Similar weighting is required for adapting the MAX-CSP [4] approach, where the *quality* of the solution is related to the sum of the unsatisfied goals.
- By using graph based state representation we had to address the problem of **typed graph comparison**. For this we adapted the DSMDIFF [8] algorithm, which relies on (i) signatures (for nodes and edges) composed of type and name information and (ii) containment relation between nodes of the graph. Initial research showed that this approach works only on small scale problems and some cases require problem specific comparators as DSMDIFF can not handle intensive node creation and deletion.

- It is also important to mention that saving **already visited states** effectively is also crucial on overall performance. For which we serialized the models using a containment based traversal combined with a signature driven ordering algorithm.
- In order to support **backtracking** between states of the model, we apply a simple transaction mechanism that saves the atomic model manipulation operations applied on the model in an undo stack. This stack not only allows us to backtrack the manipulations but also ease the computation of difference between the states of the model. This unique ability is useful in problems that require solutions that are ”nearest” to a given initial model (e.g., reconfiguration).

## IV. Conclusion and Future Work

In the current paper, we have presented a novel approach defining constraint problems using a combination of graph transformation rules and patterns and also listed some of the concerns of advantages and disadvantages.

As for the future, we plan to address some of the questions raised in Sec. III. B., especially focusing on dynamic behavioral extensions.

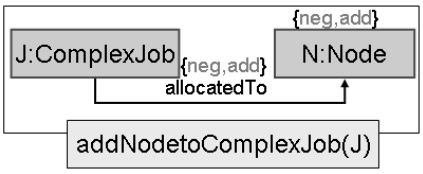



Figure 4: AddNode GT rule

## References

[1] E. Tsang, *Foundations of Constraint Satisfaction,* Academic Press, 1. edition, August 1993.

[2] R. J. Wallace, “Enhancements of branch and bound methods for the maximal constraint satisfaction problem,” in *Proc. of AAAI-96*, pp. 188–195, 1996.

[3] T. Schiex, “Solution reuse in dynamic constraint satisfaction problems,” in *In Proceedings of the 12th National Conference on Artificial Intelligence*, pp. 307–312. AAAI Press, 1994.

[4] I. Miguel and Q. Shen, “Dynamic flexible constraint satisfaction,” *Applied Intelligence, 13(3)*, pp. 231–245, 2000.

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

[6] A. Balogh and D. Varró, “Advanced model transformation language constructs in the VIATRA2 framework,” in *Proc. of the 21st ACM Symposium on Applied Computing*, pp. 1280–1287, Dijon, France, April 2006. ACM Press.

[7] R. Dechter and A. Dechter, “Believe maintenance in dynamic constraint network,” in *Proceedings of the Ninth National Conference on Artificial Intelligence*, pp. 37–42, 1988.

[8] Y. Lin, J. Gray, , and F. Jouault, “Dsmdiff: A differentiation tool for domain-specific models,” *European Journal of Information Systems, Special Issue on Model-Driven Systems Development 16(4)*, pp. 349–361, 2007.

# ENHANCING DESIGN-TIME MODEL EXECUTION IN DOMAIN-SPECIFIC LANGUAGES BY INCREMENTAL PATTERN MATCHING

**István RÁTH**
**Advisor: Dániel VARRÓ**

## I. Introduction

Visual domain-specific languages (DSLs) are used nowadays in a wide range of applications ranging from embedded to enterprise-scale systems. Many of such languages aim to capture the dynamic, behavioral aspects of the system under design. In order to support early validation of design decisions, these dynamic models are investigated by sophisticated simulation tools.

Simulation based early system validation can be carried out by mapping domain-specific models into existing mathematical simulation tools using model transformations. For instance, (i) "token game" simulators capture the data and control flow in a static network, (ii) "birth and death" simulators enable the creation and deletion of objects, but they are unable to model complex contextual logical conditions for such changes. Moreover, when domain-specific models (DSMs) are projected into existing simulation tools, the result of a simulation is provided on the level of mathematical models, and not directly on the DSM level, which can be a limitation for domain experts. In contrast, we focus on efficient simulation directly within a domain-specific modeling environment based upon the dynamic semantics of the DSL.

**Contributions of the paper.** In this paper, we present the enhanced variant of our general purpose discrete event simulation framework for domain-specific visual languages describing system behavior. Building on our efficient approach [1], we use *incremental graph pattern matching applied to model execution* as the main conceptual novelty compared to our previous paper [2] and other graph transformation-based simulation approaches. We provide an integrated execution system based on the VIATRA2 graph-transformation formalism, which leverages the incremental pattern matcher for the (i) *efficient tracking of enabledness conditions of simulation rules*, and (ii) *the efficient execution of simulation rules*. As a result, our simulation framework supports in-place model changes (e.g. user-driven editing) as well as transaction-like model changes (e.g. model imports) during simulation; the system dynamically adapts to how the models evolve and updates the enabledness of simulation rules accordingly. Our approach, integrated into the ViatraDSM framework, primarily targets *interactive* applications where the user can experiment with model execution. However, as the performance of the underlying incremental pattern matching engine allows [3], the system can go beyond model animation to perform real *model simulation*. which enables analysis involving a high number of execution runs.

## II. Overview of the Approach

### A. Discrete model simulation with complex model changes

In this paper, we focus on discrete model simulation which is applicable to domains where the state-space can be evaluated in discrete time intervals. Our approach is targeted at domains where all concepts can be captured using a finite number of dynamic entities, each having a well-defined life cycle (e.g. tokens, stateful model elements). Examples of such domains include various state machine formalisms, token games, or data flow networks. Additionally, our approach supports systems involving arbitrarily complex model changes including the birth and death of objects, reconfiguration of net-

works, etc. Note that dynamic changes are not limited to dedicated model elements of the model (e.g. tokens) but arbitrary model objects may be created and deleted during simulation.

For discrete model execution, our approach makes use of *model transformations* operating on a *graph representation* of model elements. As a demonstrating example, we use Petri nets.

The dynamic semantics is described in our approach by *simulation rules*. A rule $R = (EC, AS)$ is defined by an *enabledness condition* $EC$ and an *action sequence* $AS$, which describes model manipulation. Formally, an enabledness condition corresponds to a graph pattern, and model manipulation can be described by graph transformation (GT) or abstract state machine (ASM) rules as available in the transformation language of the VIATRA2 framework [4]. A simulation rule is executed in a *model context*, which consists of model elements satisfying constraints prescribed by the enabledness condition (a *match* of the graph pattern).

**Enabledness conditions.** *Graph patterns* represent conditions (or constraints) that have to be fulfilled by a part of the model. *Patterns may call other patterns* using the *find* keyword, which enables the reuse of existing patterns as a part of a new (more complex) one. A *negative application condition* (NAC, defined by a negative subpattern with the *neg* keyword) prescribes contextual conditions for the original pattern which are forbidden in order to find a successful match. Negative conditions can be embedded into each other (e.g. negations of negations).

**Action sequences.** *Graph transformation* [5] provides a high-level rule and pattern-based manipulation language for graph models. In VIATRA2, graph transformation rules may be specified by using a *precondition* (or left-hand side – LHS) pattern determining the applicability of the rule, and a *post-condition* pattern (or right-hand side – RHS) which declaratively specifies the result model after rule application.

### *B. Efficient execution*

The efficient execution of the simulation depends on the evaluation of enabledness conditions corresponding to simulation rules. In existing GT-based tools, the re-evaluation of such a condition would require the re-computation of pattern matches, which is expensive. To eliminate the problem, we make use of our novel incremental pattern matcher [1] based on the RETE algorithm. Our approach relies on a network of nodes storing *partial matches* of a graph pattern. A partial match enumerates those model elements which satisfy a subset of the constraints described by the graph pattern. In a relational database analogy, each node stores a *view*. Matches of a pattern are readily available at any time, and they will be incrementally updated whenever model changes occur. As an example, the pattern matcher RETE network constructed for the *transitionFireable* pattern is shown in Fig. 1.

```
pattern transitionFireable(T)=
{
  transition(T);
  neg pattern notFireable(T) =
  {
    find notEnabled(T);
  } or {
    find isInhibited(T);
  }
}
```

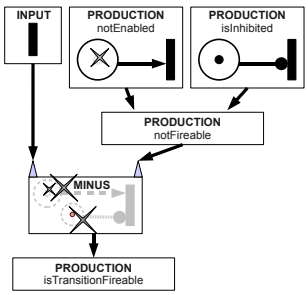



**Figure 1:** RETE matcher: 'transitionFireable'

Our RETE implementation supports *set operations* at *intermediate nodes*, which can be executed on the match sets stored at input nodes to compute the match set which is stored at the intermediate node (the minus-join node in Fig. 1 represents the negative application condition, while the inter-

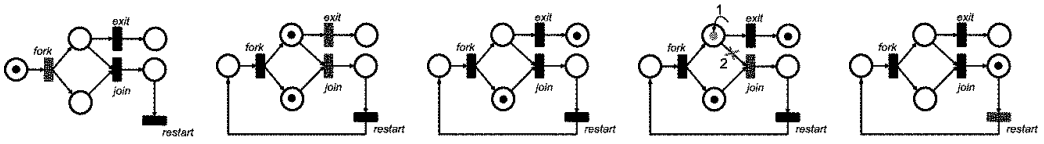
mediate production node represents the disjunction by the OR-pattern). After the network has been constructed, the match set for the entire pattern can be retrieved from the output *production node*. Our implementation supports the entire VIATRA2 pattern language, so any VIATRA2 pattern can be matched incrementally.

**Incremental updates.** The RETE network receives notifications about changes on the model. These changes are propagated through the network, modifying the match sets stored at the nodes incrementally, since each node only recomputes a partial matching. After the changes have been processed, the match set can be retrieved from the network instantly. As a trade-off, there is increased memory consumption, and a moderate overhead on update operations. From the point-of-view of the simulation engine, this means that the *enabledness conditions* can be evaluated at any given time without any expensive pattern matching, for both user-made changes (editing), transformation-induced changes, or even model imports.

### C. Integrated design-time simulation

Based upon the set of rules capturing dynamic semantics, our simulation framework supports the execution of a domain-specific model inside the modeling environment. Integrated into the ViatraDSM domain-specific modeling tool, the simulation engine may be invoked at any time; moreover, models are executed at the high abstraction level of the DSL (like in specialised simulation tools), ensuring domain consistency.

Model execution can be either fully automatic or user-guided. User assistance is a requirement in many cases, e.g. when design-time testing is performed by designers. Additionally, interactivity is also used for resolving non-determinism, which is frequently present in practical model simulation scenarios. In that case, the user is given a set of choices at a *choice point*, and the execution continues depending on the actual choice.




(a) Initial state (b) Choice point (c) Trap state (d) On-the-fly model editing (e) Final state

**Figure 2:** Petri net simulation phases

The most important phases of an example simulation process in the Petri net example are illustrated in Fig. 2. The user can make various changes as the simulation is being executed by *editing the model on-the-fly*. For instance, the user may re-enable the transition *join* by either placing a new *token* into the empty input place (denoted with 1 in Fig. 2(d)), or delete the input arc (denoted with 2). In both cases, the RETE network is automatically updated following the editing action, moving the simulation system back into a state where *join* is enabled for firing. The system also provides dynamic support for the addition, change, or removal of transformation rules (with enabledness conditions and action sequences), since the construction of RETE networks is dynamically performed as pattern definitions are loaded. This feature is analogous to "hot code replace" found in modern program debuggers, and it is very important for agile development.

## III. Related work

Industrial DSM frameworks such as Microsoft DSL Tools and GMF both concentrate on a generative approach to ease the development of modeling environments; model execution and transformation in general are yet to be integrated.

Several academic DSM frameworks are complemented with support for model transformations, typically, using a *graph transformation* [5] approach, which approaches show the closest correspondence with our ViatraDSM framework. DiaMeta (a follow-up of DiaGen [6]) replaces hypergraph grammars by MOF as provided by the MOFLON tool suite [7] to allow users not only to specify domain-specific modeling languages but also to generate corresponding diagram editors. Advanced multi-domain modeling features are supported by ATOM3 [8], which defines the concept of view metamodels sharing a common metamodel of a visual language. The consistency of different views and abstract-to-concrete syntax mappings [9] is maintained by triple graph grammars (TGG) while simulators are also defined by graph grammar rules.

## IV. Conclusion

In the paper, we presented a discrete event interactive simulation framework for dynamic domain-specific modeling languages. The dynamic semantics of a language was captured by a combination of graph transformation and abstract state machine rules as provided by the transformation language of the VIATRA2 framework.

As the main novelty, our approach is built upon an incremental graph pattern matching engine, which instantly identifies all contexts where the enabledness condition of a simulation rule is enabled. This is carried out by incrementally keeping track of matches of graph patterns related to enabledness conditions. As a result, simulations requiring complex model changes (even with intensive creation and deletion of objects) can be executed in an efficient way.

## References


[1] G. Bergmann, A. Ökrös, I. Ráth, D. Varró, and G. Varró, "Incremental pattern matching in the VIATRA transformation system," in *GRaMoT'08, 3rd International Workshop on Graph and Model Transformation*. 30th International Conference on Software Engineering, 2008, Accepted.

[2] I. Ráth and D. Varró, "Design-time Simulation of Domain-specific Models by Interactive Model Transformations," in *PhD MiniSymposium*. BME MIT, 2008.

[3] G. Bergmann, A. Horváth, I. Ráth, and D. Varró, "A Benchmark Evaluation of Incremental Pattern Matching in Graph Transformation," in *ICGT'08, 4th International Conference on Graph Transformation*. European Association for Theoretical Computer Science and European Association of Software Science and Technology, 2008, In Press.

[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] H. Ehrig, G. Engels, H.-J. Kreowski, and G. Rozenberg, Eds., *Handbook on Graph Grammars and Computing by Graph Transformation*, vol. 2: Applications, Languages and Tools, World Scientific, 1999.

[6] O. Köth and M. Minas, "Generating diagram editors providing free-hand editing as well as syntax-directed editing," in *GRATRA 2000 Joint APPLIGRAPH and GETGRATS Workshop on Graph Transformation Systems*, H. Ehrig and G. Taentzer, Eds., pp. 32–39, Berlin, Germany, March 25–27 2000.

[7] M. Minas, "Generating visual editors based on FUJABA/MOFLON and DIAMETA," Tech. Rep., University Paderborn, 2006, Proc. 4th Fujaba Days, pp. 35-42, Technical Report tr-ri-06-275.

[8] J. de Lara and H. Vangheluwe, "AToM3: A tool for multi-formalism and meta-modelling," in *5th International Conference, FASE 2002: Fundamental Approaches to Software Engineering, Grenoble, France, April 8-12, 2002, Proceedings*, R.-D. Kutsche and H. Weber, Eds., vol. 2306 of *LNCS*, pp. 174–188. Springer, 2002.

[9] E. Guerra and J. de Lara, "Event-driven grammars: Towards the integration of meta-modelling and graph transformation," in *Proc. 2nd International Conference On Graph Transformation (ICGT 2004)*, H. Ehrig, G. Engels, F. Parisi-Presicce, and G. Rozenberg, Eds., vol. 3256 of *LNCS*, pp. 54–69. Springer, 2004.

# STANDARDS-BASED ASSESSMENT OF DEVELOPMENT TOOLCHAINS

**Zoltán SZATMÁRI**
**Advisor: István MAJZIK**

## I. Introduction

Software pervasiveness has the consequence that our everyday life depends on software to a considerable extent. To reduce the risks of software design failures, the *software development processes* are more and more subject to regulations fixed in (domain-specific) standards. Accordingly, if software is deployed in a critical environment then an independent assessment is needed to certify that its development process, i.e., the set of applied *methods* and the supporting *tools* or *toolchains* are compliant to the requirements stated in the standard. The need for certification has arisen in several European projects (e.g., DECOS, MOGENTES) that aim at the elaboration of toolchains for the development of embedded and/or safety-critical systems.

The goal of my work is to support the assessment of development processes and toolchains by elaborating *formal verification techniques* that allow the automated checking of the compliance to standards. This vision necessitates the solution of the following tasks:

- Formalization of the requirements in standards that concern the use of methods and tools.
- Definition (or adaptation) of modeling techniques to describe the relation of methods, the capabilities of tools, and the construction of (domain-specific) development processes.
- Elaboration of techniques that check the compliance of concrete development processes (constructed by process designers) to the requirements.

It is worth mentioning that the formalization of the requirements and the model-based description of tools and methods open a way to support also the *synthesis of processes and toolchains* that are compliant to the standard. The process designer can be assisted by (i) identifying missing methods and tools, (ii) proposing alternative solutions, (iii) offering a library of toolchain patterns, (iv) optimizing processes from the point of view of costs, time, safety etc.

In the following we describe ideas and initial results related to the implementation of the above mentioned tasks.

## II. Formalization of the requirements

Formalization is a prerequisite of both formal verification and synthesis support. I focused on the development processes for safety critical applications, and analyzed the EN50128 standard for railway applications [1]. This standard defines five *safety integrity levels* (SIL) for development processes and describes methods that can be applied during the process. For each development step the mandatory, highly recommended, recommended and not recommended methods are described in a tabular form. The development methods are refined hierarchically, i.e., several high level methods are decomposed into alternative combinations of lower level ones (See Fig. 1).

A formal representation of the hierarchical structure of methods can be provided by defining an ontology [3] and describing it using description logic, ontology. Here concepts refer to the development methods and their relations include the refinement.

Using the concepts defined in the ontology, the *necessary* and *sufficient conditions* for the selection of methods and the dependency on the safety integrity level can be described for each step in the process using logical statements (interpreted as static constraints). The required temporal ordering of methods can be formalized using *temporal logic* formulae.

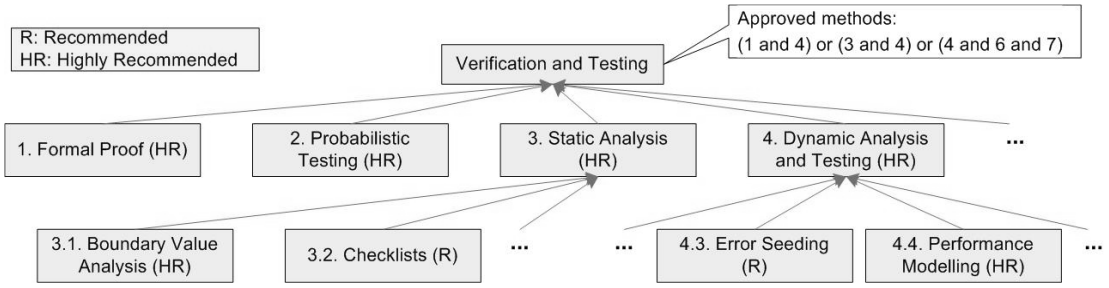



Figure 1: Verification and Testing methods for SIL-4 (EN50128)

## III. Assembling an extensible tool repository

The next step of the formalization process is the construction of the tool repository. This repository is a collection of tools that can be used during the construction of the development processes. Each available tool is classified on the basis of the concepts defined in the ontology constructed in the previous step, i.e., for each tool the supported methods are given. Complex methods are supported by *toolchain patterns* that are formed by tools that have to be executed in a predefined sequence.

## IV. Modeling the development process

The (domain-specific) development process is formalized using a *process model* which describes the tasks, input and output artifacts, the roles and tools involved in the development process. The MOGENTES project uses the Eclipse Process Framework [2] to model the processes. Using this framework the process designer (i) can construct the specific development process, and can *assign the available tools* to the tasks of the process or (ii) can choose from the available *toolchain patterns*. The tasks of the process implement particular methods that are classified using the ontology.

## V. Assessment of domain-specific development toolchains

Using the formalism described above, all of the tasks, the tools and the requirements of development processes are characterized using the concepts represented in the ontology. Accordingly, the *standard conformance* of the selection of methods and their supporting tools in the development process can be checked using an *ontology reasoner*. The requirements about the ordering of the methods can be checked using a temporal logic *model checker*. This way the assessment can be supported by reusing existing formal methods and checker tools.

Note that based on the process model and the available tools in the repository the reasoner can identify missing methods and can give hints about the supporting tools as well.

## VI. Conclusion

Formalization of requirements is a wide research area. In this work I proposed an approach to *formalize the requirements of development processes*. This approach forms the basis for the assessment of the standard compliance of specific toolchains and provides support for the process designers to construct certifiable development processes.

## References

[1] EN 50128: *Railway applications - Communication, signalling and processing systems - Software for railway control and protection systems*. URL: http://www.cenelec.eu

[2] *Eclipse Process Framework project*, URL: http://www.eclipse.org/epf/

[3] Szeredi P., Lukácsy G., Benkő T.: *A szemantikus világháló elmélete és gyakorlata.,* Typotex, Budapest, 2005.

# DECISION SUPPORT SYSTEM FOR DESIGNING SNP ASSOCIATION STUDIES

**Gergely HAJÓS**
**Advisors: Péter ANTAL, Tadeusz DOBROWIECKI**

## I. Introduction

The aim of the single nucleotide polymorphisms (SNPs) based association studies is to find SNPs relevant to the disease under investigation. Despite of the growing importance of genome-wide association studies (GWAS), the selection of tens out of million SNPs present in the partial genome screening studies (PGSs) is still an important problem and cannot be expected to disappear in the near future.

The presented solution provides help for the researchers in the design of association studies by searching, browsing and selecting biologically and medically relevant SNPs. Another important issue is to consider technological aspects of the laboratory's measurement tools [1] – whether the SNP set is appropriate for measurement. Finally we support the design of the overall measurement, i.e. the scoring of the joint set of potential SNPs, by balancing their individual relevance and internal redundancy. With the integration and automated analysis of biomedical databases and derived datasets, the solution shortens from one or two weeks long preselection period to some days.

Because of the importance of the accumulating potentially related data sets and multistep dependent measurements, we decided to formalize this problem as a sequential decision problem.

## II. The method

Earlier SNP properties were collected to estimate the biological and medical relevance of SNP sets with the help of physicians and biologists [2] from the Semmelweis University. The implemented decision support system offers the researchers a graphical interface wherewith it is possible to express their preferences to score these mentioned properties. The program finally selects a SNP set by the biological properties, the given scores, and the correlation between them.

The program performs three consecutive functions controlled by the user (Fig. 1):

1. **Searching and browsing** – The first step is a complex search which is designed to integrate certain SNP searching methods used by the researchers earlier at the DGCI laboratory (i.e.: UCSC [3], HapMap).

2. **Defining priorities** – In the second step it is possible for the users to set coefficients to express preferences about SNP properties. In this step with the help of the decision network, every SNP gets a score expressing its utility and relevance.

3. **Set selection** – Finally the third step performs the set selection, the program attempts to cover the whole given sequence while it chooses relevant and (if necessary) measurable SNPs [4].

The program uses several data sources, databases of the NCBI project such as dbSNP [5], GO [6], HapMap [7]. It performs meta-analysis on the results of text and data mining processes, previous measurements and Bayesian statistical analysis.

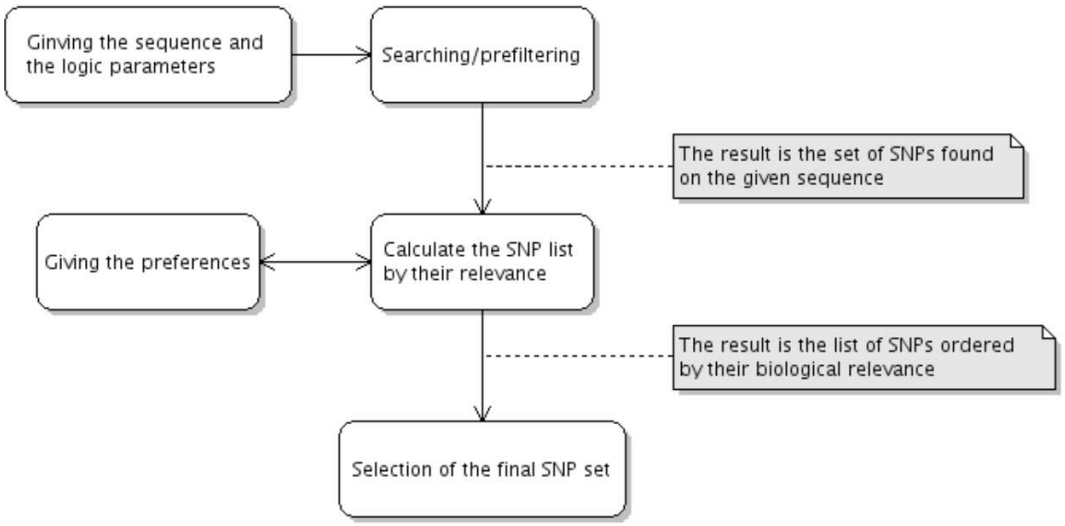



Figure 1: The process of SNP selection

## III. Conclusion

The automated and intelligent selection of SNPs is a central issue in modern statistical genetics. We implemented a flexible and principled system, focused on the needs of the SOTE DGCI laboratory with a rich user interface designed to help physicians and biologists.

## IV. Future goals

A typical association research project is an iterative sequence of selections, measurements and analysis, while currently available systems — including ours — are only semi sequential, as they can incorporate previous results at best, but they cannot predict the value of further analysis. Such a partially sequential approach is based on three data sources: expert knowledge, publicly available databases, and the results of previous analysis. In the future we want to emphasize the role of potential subsequent measurements. There are two key elements to achieve this in a principled way in the new version of the system:

- modeling the future probabilities of the relevance of SNPs in a hypothetical study with additional measurements. It is solved by constructing typical learning curves for SNPs w.r.t. a given disease (using bootstrap and Bayesian methods [8, 9]),
- approximating the value (expected utility) of their further measurements.

## References

[1] Beckman Couler, *Beckman Couler SNPStream genotyping tools*, http://www.mathworks.com/matlabcentral/fileexchange/.

[2] P. Wang, "Snp function portal: a web database for exploring the function implication of snp alleles," *Bioinformatics*, 22(14):523–529, July 2006.

[3] University of California, Santa Cruz, *UCSC Genome Bioinformatics Site*, http://genome.ucsc.edu/.

[4] Z. S. Qin, "An efficient comprehensive search algorithm for tagsnp selection using linkage disequilibrium criteria," *Bioinformatics*, 22(2):220–225, Jan. 2006.

[5] National Center for Biotechnology Information, *dbSNP*, http://www.ncbi.nlm.nih.gov/projects/SNP/.

[6] GO Consortium, *Gene Ontology*, http://www.geneontology.org/.

[7] *International HapMap Project*, http://www.hapmap.org.

[8] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap*, Chapman & Hall, London, 1993.

[9] P. Antal, "A bayesian view of challenges in feature selection: Feature aggregation, multiple targets, redundancy and interaction," *JMLR Workshop and Conference Proceedings 4*, pp. 74–89, 2008.

# DEPENDABILITY MODELING FOR MODEL BASED SYSTEM MANAGEMENT

**Imre KOCSIS**
**Advisor: András PATARICZA**

## I. Introduction

Qualitative, static abstraction of error propagation [1] is a paradigm that promises to provide a hierarchical modeling approach with manageable complexity for the dependability analysis and on-line diagnosis of general purpose IT systems.

Our research group has developed an integrated pilot environment to demonstrate modeling and monitoring supported diagnosis in this framework and to validate the feasibility of the overall approach. This paper describes the lessons learned regarding modeling and the engineering interpretation of some modeling and diagnostic concepts. Qualitative, static error propagation modeling is briefly introduced in Section II; a more extensive and mathematically rigorous treatment of the subject can be found in [1] and [3]. Due to the limited extent of the paper, the pilot environment is not described here.

## II. Static, qualitative error propagation modeling

Let us model system components are state transition systems with a vector $\boldsymbol{s}$ of state variables; $\forall s_i \in \boldsymbol{s}$ is associated with a domain $\boldsymbol{D_i}$.Variables can be input, output and internal ones; the sub-vector of $\boldsymbol{s}$ containing only the input and output variables is the *interface vector* of the component. A system model $\boldsymbol{M}$ consists of interconnected components $\{C_1, \dots, C_n\}$.

For all components, a finite set of anticipated faults $\boldsymbol{F_i}$ is associated with component $C_i$. Each component $C_i$ has a fault free reference instance $C_i^{good}$ (that is, a reference state transition system); a separate mutation $C_i^j$ is created for $\forall j: j \in 1 \dots |\boldsymbol{F_i}|$ to describe behavior in the presence of the expected faults. The instance effective at a given time is selected by the *fault state variable* of the component: $f_i \in \boldsymbol{F_i} \cup \{good\}$, where the valuation "good" selects the fault-free reference (see Figure 1).

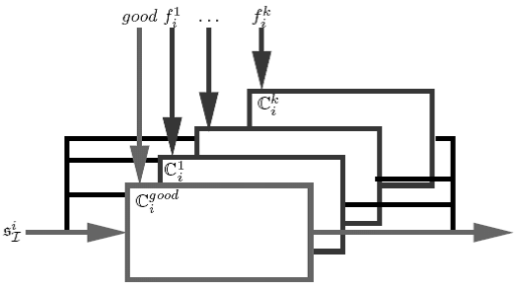


Figure 1. Component model

By *error* we refer to impacts of faults observable as a discrepancy in the behavior of the actual system from that of the reference system. Let us denote the model of the reference system with $\boldsymbol{M^0}$ and that of the actual one with $\boldsymbol{M^a}$. $\boldsymbol{M^0}$ is fault free; $\boldsymbol{M^a}$ may contain faulty components. In this composite model, *composite state variables* are pairs $c = [a, r]$ of the corresponding state variables in the reference and the actual model. *Error modes* are a complete finite partitioning of the state spaces of composite state variables into disjoint subsets by error predicates $\varepsilon_0, \dots, \varepsilon_n$ on the composite state variables, arbitrarily chosen during modeling so that for any composite state variable value, exactly one error predicate shall be true. $\varepsilon_0$ shall always denote the error-free case with the associated error mode "good". Error predicates are a form of spatial abstraction over the space of possible erroneous discrepancies from the error-free states. The

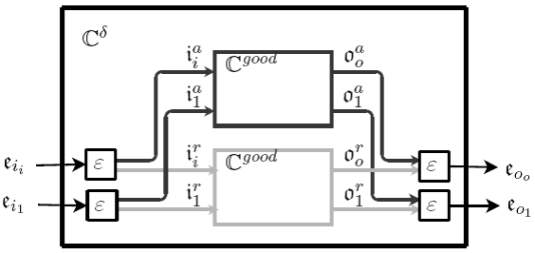


Figure 2. Composite model

simplest case is when there are only two error predicates, generating the discrepancy-partitions "matching" and "erroneous", leading to a binary, error detection oriented abstraction. It is to note that this formalization of errors also allows for modeling errors in nondeterministic systems. The error predicate sets defined on a pair of actual and reference state transition systems give rise to component error state-transition systems. Such a state transition system can be seen as an automaton that "reads" incoming errors on its inputs and "writes" errors on its outputs (Figure 2).

*Syndromes* are temporal abstractions of the variable valuation runs of the composite state variables. For any composite state variable, a complete and mutually exclusive set of failure predicates categorizes the set of error sequences $\boldsymbol{\Sigma^e}$ by the failure mode function $\gamma: \boldsymbol{\Sigma^e} \rightarrow \boldsymbol{\nu}$ into the set $\nu = \{y_0, \ldots, y_t\}$ of syndromes. Again, the simplest example is a binary one; error sequences that contain only the "good" value are categorized as "good", while all others (containing at least one error mode that is not "good") as "failure". Syndromes observable by the system user are referred to as *failure modes*, too.

In terms of syndromes the dependability-related behavior of a component is described by the *syndrome relation*. Formally, the syndrome relation is a relation in the classical sense, with exactly one underlying variable associated to each interface variable of the component at hand; each underlying variable has as a domain the set of the syndromes used for the temporal compaction of the error runs of the associated interface variable. Elements of the relation are the input syndrome vectors and output syndrome vectors that the underlying error state transition system supports. As an example, let us model a classical Triple Modular Redundancy unit (TMR), with binary error modes (0: "good"; 1: "erroneous") on all inputs and the output and binary failure modes (0: "no error", 1: "at least one error") on sequences over the binary error mode set. Table 1 shows that eliminating the need for a state based representation comes at a cost. The syndrome relation cannot make a distinction between inputs based on the timing of incoming errors and so, it can define only a nondeterministic outcome for a number of cases.

| INPUT #1 | INPUT #2 | INPUT #3 | OUTPUT |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 or 1 |
| 1 | 0 | 1 | 0 or 1 |
| 0 | 1 | 1 | 0 or 1 |
| 1 | 1 | 1 | 0 or 1 |

Table 1. TMR syndrome relation

By associating *syndrome variables* to composite state variable inputs, outputs and fault mode variables of components, dependability analysis problems as diagnosis or impact estimation can be formulated as search for (partial) syndrome variable valuations with a pre-defined (partial) variable valuation as an assumption and syndrome relations as constraints to be met.

Such problems can be easily translated to finite domain Constraint Satisfaction Problems (CSP) [2]. CSPs are networks of variables interconnected by a set of relations called constraints. A CSP is solved, if a value assignment has been found for all variables, such that all the relations are satisfied. A CSP can have multiple solutions that are variable valuation vectors each; the single-variable projections of these vectors form the possible value sets of the variables. Our current implementation uses the built-in finite domain CSP module of SICStus Prolog [6].

## III. Monitoring support for syndrome detection

For on-line diagnosis, events and measurements of the system have to be transformed into *error valuation streams* of "monitored" composite variables of the system model and the error streams classified into syndromes. However, while static, qualitative modeling by definition operates with potentially infinite error runs, on-line monitoring tracks prefixes of such error runs. This way, at any given moment we can only compute the set of *possible syndrome values* for any monitored composite variable by checking that which syndrome predicates *can* be true for the set of error runs that is the subset of all error runs with the observed prefix.

**Proposition 1:** *the set $\boldsymbol{\sigma}_{c_i}^{mon}$ of possible syndrome values on the i-th monitored composite variable of a component is a set that monotonically decreases with time.*

It follows that a CSP model is suitable for incremental on-line diagnosis, as with the passing of time domains of syndrome variables are only further constrained by observations. Also, this way error propagation (or measurement processing) delays do not lead to incorrect, only ambiguous diagnosis.

In practice, most syndrome dictionaries are composed of syndromes that are *not pairwise finitely distinguishable*; i.e. for a syndrome set $\nu$ there is at least one pair $y_i, y_j \in \nu\ (i \neq j)$ so that $\nexists i \in \mathbb{N}$, a finite and bounded limit for which *any* error sequence of length $i$ can be the prefix of only $y_i$ or only $y_j$ error runs – or neither. Most importantly, the earlier introduced "good" and existentially formulated "failure" failure modes are such. This means that for a monitored composite input or output variable where we consistently do not observe errors $\boldsymbol{\sigma}_{c_i}^{mon} = \{good, failure\}$ indefinitely, not constrained further by monitoring itself – as "failure" also covers potential errors that are to *happen in the future*. Note that these include not only effects of future faults, but also effects of faults with delayed manifestation (dormant faults, error propagation delay).

**Proposition 2:** $\forall j \in \mathbb{N}$, $\boldsymbol{\sigma}_{c_i}^{mon}$ *defines a language of* $j$ *long* $(j: 1 \ldots \infty)$ *error run prefixes, that (trivially) always contains the observed error run prefix and possibly also others. It is by construction an abstraction of the observed finite behavior of the composite system in the formal sense.*

More precisely, $\boldsymbol{\sigma}_{c_i}^{mon}$ has "finite AND" semantics, so that the language defined contains only the finite error run prefixes that piecewise can be extended to match any syndrome in the set. E.g. $\boldsymbol{\sigma}_{c_i}^{mon} = \{good, failure\}$ on a monitored variable means that up until the last measurement, no error was observed. Similarly, a syndrome set can also have "OR" semantics, defining a language of finite prefixes where each prefix can be extended to comply to at least one of the syndromes. An "OR" set produces a superset of the "finite AND" set defined language over the same syndromes (again, a true abstraction). The purely CSP-inferred possible syndrome sets of unmonitored composite variables have "OR" semantics in this sense due to the possible nondeterminism of syndrome relations. This asymmetric handling of possible syndrome set semantics certainly loses much information during constraint propagation through purely inferred variables; however, generalized "AND/OR" possible syndrome set representation based diagnosis is infeasible with classical CSP engines. As a part of our current research efforts we are examining whether Multiple Decision Diagram based CSP solving could support this approach (see [4] and [5]).

## IV. Fault activation history modeling in the static framework

The fault state variable $f_i$ was deliberately not included in the interface vector of components; it is a case by case, conscious modeling decision whether it should be an artificial input variable (a fault mode activator input) or an internal variable. In most cases, $f_i$ appears besides the input and output variables as an underlying variable for the syndrome relation(s) of the component. But as $f_i$ is a proper state variable, at least a partitioning of the possible fault mode sequences must be undertaken in order to elevate the fault mode variable into a syndrome relation. In other words, fault activation runs have to be classified into *fault syndromes*.

The occurrence (and disappearance) patterns of (internal) faults of software and hardware components are by definition largely arbitrary; as a consequence, the set of all fault activation runs generally cannot be partitioned as readily as error runs. Consequently, for off-line analytical as well as on-line diagnostic purposes permanent faults, pure appearance (not allowing fault mode change after the first cycle of active state) and time bounded transients of a single type should be considered as syndromes – the simplifying assumptions that are usual in dependability modeling in general. Certainly the associated loss in behavioral coverage must be taken into account.

It is to note that for on-line diagnostic usage, purely static modeling is in a slight conceptual mismatch with the usual goal of diagnosis: inferring the *current fault state* of components (or more

precisely, field replaceable/repairable units), not their fault activation history or future fault activations (what is anyhow almost arbitrary). Also, our approach is of asymmetric nature:

**Proposition 3:** *on-line diagnosis based on inference by syndrome relations is asymmetric in the sense of classical system level diagnosis [7]: in the general case, tests (monitoring and active probing) narrow the possible fault syndrome set of the fault state variable of a component in the presence of faults, but not in their absence.*

From the point of view of diagnostic interpretation, this means that a set of possible fault syndromes $\boldsymbol{\sigma}_{f_i}^{inferred}$ on a fault state variable is not to be expected to be constrained to a single "no fault" value. Let us define a binary fault syndrome dictionary ("ok": no fault activation, "fault": the single permanent fault is eventually activated and remains unrepaired). $\boldsymbol{\sigma}_{f_i}^{inferred}$ is of "OR" semantics (unless monitored), so $\boldsymbol{\sigma}_{f_i}^{inferred} = \{ok, fault\}$ does not mean that no fault was activated up until some last point of measurement supporting inference; however, as another asymmetrical property $\boldsymbol{\sigma}_{f_i}^{inferred} = \{fault\}$ *does* mean that the fault activation has already happened, marked by the exclusion of the "ok" value.

An obvious drawback of modeling only the usually significant fault activation patterns (and so managing complexity) is that the model will not describe the behavior of the components after the repair of permanent faults and under subsequent faults. However, under those circumstances the otherwise incremental diagnosis can be "reset": the CSP solving globally reinitialized and the next measurement cycle of error detectors interpreted as the first errors of newly started syndromes. Two conditions must be met for resets to be applicable. First, the syndrome relations must be "attachable" to the components at any time during their functioning (at any time the errors on the component can be the first errors of the appropriate syndromes), or at least attachable at suitable well defined states of the component (e.g. after taking it to a provably fault free state). Second, the transient errors caused by the previously faulty component must be allowed to leave the system. After reset, *stored errors* (for example, erroneous data injected into a database) can appear as fault modes in the model.

## V. Future work

Based on the outlined experiences, the most important research direction is switching to decision diagram based CSP solving, or at least solution representation (although not discussed here, among other reasons this is also needed for diagnosis-integrated repair action planning). We also have some initial results regarding a hybrid modeling approach where fault states replace fault syndromes in syndrome relations, thus eliminating the drawbacks of fully static modeling from the diagnostic point of view.

## References


[1] A. Patarica, *Model-based dependability analysis*, DSc Thesis, Hungarian Academy of Sciences, Budapest, 2006.

[2] E. Tsang, *Foundations of Constraint Satisfaction*, Academic Press, London, 1993.

[3] A. Pataricza, "Systematic generation of dependability cases from functional models", *Formal Methods for Automation and Safety in Railway and Automotive Systems (Proc. of Symposium FORMS/FORMAT 2008)*, L'Harmattan Hongrie, Budapest, 2008.

[4] D.M. Miller, R. Drechsler, "Implementing a multiple-valued decision diagram package", *Proc. of the 28th IEEE International Symposium on Multiple-Valued Logic*, 1998, pp 52-57.

[5] H.R. Andersen, T. Hadzic, J.N. Hooker, P. Tiedemann: "A constraint store based on multivalued decision diagrams", In: *Bessière, C. (ed.) CP 2007. LNCS, vol. 4741*, Springer, Heidelberg, 2007, pp. 118–132.

[6] M. Carlsson, G. Ottosson, B. Carlson: "An Open-Ended Finite Domain Constraint Solver", *Proc. of the 9th International Symposium Programming Languages: Implementations, Logics, and Programs*, Springer, 1997.

[7] W.R. Simpson, J.W. Sheppard: *System Test and Diagnosis*, Springer, 1994.

# SYNCHRONIZATION OF IT INFRASTRUCTURE MODELS

**István SZOMBATH**
**Advisor: András PATARICZA**

## I. Introduction

Creating and maintaining the appropriate quality of services delivered by business-driven IT infrastructure is a huge challenge. System management provides a systematic approach to operate and control the IT infrastructure in order to keep costs and resource usage at a reasonable level.

System management has to assure an appropriate reaction to changes in the environment and in the system itself originating both in evaluation and faults. This way, the closed-loop model elaborated in control theory is widely adopted in system management as well. Efficient feedback requires exact information both on the structure and on the state of the management system. The first category of information on the structure on the system under control becomes more and more important in the point of view of practical system management due to the widespread use of adaptive architectures like those in autonomic grid and self healing systems in which structural changes form an essential part of their operation. This operation is over fundamental importance for synchronizing the models of the control system and the system under supervision.

Furthermore it is problematic how to obtain this information with an affordable effort in the terms of intrusiveness and additional workload on the resources.

The selection of adequate information sources and the processing of data gain by them is another core issue. For instance it is known that fast impact analysis predicting the potential consequences and their severity of the individual events would be an outmost importance in preventing or at least confining outages in critical business services by focusing the reaction priorities onto critical events.

Therefore the collections of all necessary but moderate amount of information required for an efficient situation analysis and reaction design is the primary objective. Reusability and generality of the approach are important in order to implement and validate the monitoring, analysis, decision making, and feedback mechanisms embedded into a product by using a reasonable amount of effort.

## II. Objectives and scientific background candidates of the research and development

Dependency between infrastructure components, application, and business layer dependencies are all important for impact and root cause analysis for assessing the impact of changes and faults on the system. However to obtain this information is difficult. Passive observation approaches that extract this information by monitor and analyzing network traffic are superior to active probing based network dependency discovery as passive methods are non-intrusive and cause minimal additional workload on the network. Such a passive approach is dependency mining from network information flow logs (such as NetFlow records).

The main difficulty with passive methods is the selection of relevant data out of the huge amount of logged but typically incomplete (truncated), once collected originally for statistical purposes only.

End to end dependency discovery aiming at determining the minimal amount of flow information required to discover hidden inter-application dependencies. An important use case of end-to-end discovery in enterprise infrastructures and datacenters is that applications communicating indirectly via multiple tiers without any observability of the infrastructure itself or of the deployment of business tiers of the applications onto the physical components.

According to practical experiences simple event correlation is unable to cope with the problem of the reconstruction of the interoperation structure from the observations available in the form of dirty and incomplete traces primarily due to the huge amount of data hiding the relevant information.

More affective approaches like representative patterns mining, Hidden Markov Model, graph mining or simple regular expression search will be investigated.

Typical applications can be collected into a library of architectural patterns composed of typed components and their inter-component communications relations. The discovery process for inter-node communications can rely on matching those sub patterns instead of searching for individual p2p logic relations. The existence of typical architectural patterns for the majority of business applications promises the substitution of elementary interconnection search with a high level pattern search. A promising algorithmic approach uses the theory of graph coloring. The information in the traces can be represented as a graph spanned by nodes and communication relations colored by their respective types. The problem of high-level pattern identification can be reformulated as finding matches of the subgraphs describing the business patterns. Each match delivers a candidate deployment conforming to the observations. In this way static structure reconstruction of actual deployment is possible by gradually testing the candidate solutions**.**

Majority of existing dependency detection methods can deal with static models only accordingly they are unable to catch such important affects as dynamic resource allocation and task migration and demand driven interactions (like those in SOA). The goal of the research is to elaborate on a flow record information guided event driven change management exploiting the locality of the changes in typical dynamic structures for modifying the dependency model.

Incremental model building is a prerequisite to synchronize the core management model (CMDB) with the changes in the infrastructure in order to assure an on line evaluation to the criteria to fulfill. While the algorithmic background incremental model building and maintenance is well elaborated this approach was never deeply investigated in the context of IT infrastructure management yet.

## III. Model synchronization of IT infrastructure models

As a proof of concept first a so called source model is created that represents an IT infrastructure. The used source metamodel is compatible to the CMDB metamodel described by ITIL. It consists of components and channels (relationships, that connect two components) both with attributes such as name, type, and state. From the source model a target model is created using graph transformation. The target metamodel consists of facts and rules (similar to prolog). Rules describe behavior of components, while the facts describe states of components and channels. As a result a query can be evaluated upon the statements where the answer represents possible values of the unknown component properties (such as state). The transformation itself is event driven (live), thus components in the target model may added, deleted, or changed upon change in the source model.

In this way a pilot system is created that transforms a CMDB like model into an analytical model where fault propagation can be evaluated. It also proves that event driven response to changes in the IT infrastructure is possible.

As a result it is proved that incremental and live paradigm can be used in IT system management. In the near future we would like to implement an industrial enterprise solution that uses these paradigms to reflect changes in the IT infrastructure and synchronizes separate infrastructure models.

## References


[1] Paramvir Bahl *et al*., *Discovering Dependencies for Network Management*, Proceedings of the FifthWorkshop on Hot Topics in Networks (HotNets-V), November, 2006.

[2] Thomas Karagiannis,Konstantina Papagiannaki, Michalis Faloutsos, *BLINC: Multilevel Traffic Classification in the Dark*, ACM SIGCOMM Computer Communication Review, vol. 35, no. 4, pp. 229-240, October, 2005.

[3] Paramvir Bahl *et al*., *Towards Highly Reliable Enterprise Network Services Via Inference of Multilevel Dependencies*, SIGCOMM, pp. 13-24, 2007.

[4] Alexandru Caracas *et al*., *Mining Semantic Relations using NetFlow*, Third IEEE/IFIP International Workshop on Business-driven IT Management, 2008.

# AUTOMATED RECONFIGURATION IN HETEROGENEOUS IT INFRASTRUCTURES

**Dániel TÓTH**
**Advisor: András PATARICZA**

## I. Introduction

Reliable and scalable IT services are a critical part in the daily operation of many organizations. These services are provided by a wide range of special purpose *applications*, composed of many individual service *components*, deployed over a complex runtime environment called *IT infrastructure*. The complexity of the deployment and the generally unreliable nature of the commonly available components (built-in redundancy limited due to cost constraints) makes the IT services prone to failures which can threaten the operation of the organization. Contemporary IT service management procedures (such as ITIL[1]) call for both qualitative and quantitative specification of the expected services and their performance. The minimum acceptable levels and desired target levels of chosen performance metrics are specified in *Service Level Agreements (SLA)*.

Compliance to the SLAs requires a constant management of IT infrastructure, including *monitoring* of the components' health state and external threats such as increased or significantly changed load, then *reacting* to errors and changed conditions. It is also desirable to minimize the cost of equipment and maintenance in the process, so underutilization of resources can be considered an abnormal condition just like overutilization.

Reaction to errors and non-ideal load conditions may range from taking action to repair or *parametric reconfiguration* of an individual component, to a *structural reconfiguration* involving the reorganization of many components. These actions may either be automated or manual, the former having the advantage of significantly faster reaction, and less human resource needs. However automated reconfiguration is only available in very specialized cases.

Some concepts of adaptivity by automatic reconfiguration are already implemented by technologies such as fail-over and load-balancing clusters. Clusters typically require specially developed components which limits their general applicability in applications composed of already existing, heterogeneous components. Furthermore most cluster technologies make use of statically configured components meaning that external intervention is required in case their redundancy or performance capacity becomes exhausted.

In this paper I present an approach for a more general way of automated reconfiguration that can extend and complement the existing technologies.

## II. Modeling for Reconfiguration

Automated reconfiguration requires intelligent decisions to be made to select the action to be taken to reach a given goal. The decision process is supported by the model representations of the system along with executable model manipulation rules to model the semantics of the reconfiguration operations and the behavior of the system. Graph Transformation (GT) formalism was chosen for rule definition.

ITIL introduces the concept of *Configuration Management Database (CMDB)* as a central repository describing the system components, their parameters and dependency relationships. A number of industry standard tools exist that implement a CMDB with automatic discovery tools. These tools typically use custom database schemata or metamodels for representing CMDB, although some standards also exist for configuration modeling.

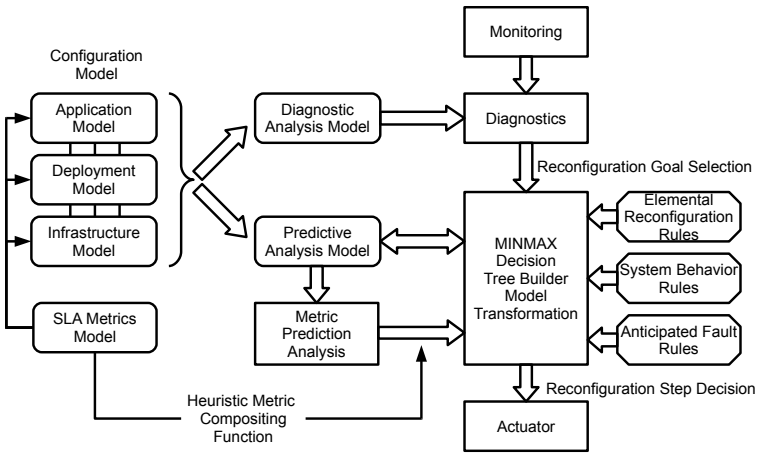



Figure 1: Overview of Reconfiguration Process and Models

### A. *Configuration Model*

In our approach (*Fig. 1*) it is necessary to employ a specific structure for configuration modeling, that is somewhat different from the available CMDB implementations. The system is viewed as a two layer model, with a separate mapping model between the layers:

- The Infrastructure layer consists of the runtime environment, and available basic resources, typically the *concrete* pieces of physical hardware and optionally the installed operating system, virtualization framework or firmware. This layer is immutable by the reconfiguration.
- The Application layer is an *abstract* model of the necessary service component types of the application and its possible relationships. Relationship multiplicity annotations describe constraints on possible number of instances for each service component type. A typical example would be a web application that consists of a database, an arbitrary number of application servers and a load balancing web front-end. The relations are the high level data and control flow connections, such as JDBC, Java Remoting and HTTP.
- The Deployment mapping shows how the *concrete instances of the application components* are assigned to the infrastructure components. This constitutes the *"configuration"* in its narrower meaning, this part of the model is subject to the reconfiguration process. The assignment may be non-trivial, the same infrastructure resource may host multiple components or some application components may have more instances and span multiple resources. In this sense the abstract application layer model is partially the metamodel for the deployment mapping.

The infrastructure layer and the deployment mapping can be derived from a traditional CMDB by model transformation, the abstract application model needs to be provided separately, for example from UML component model artifacts available from the application development.

### B. *Metric Definition*

To assess the health state and performance of the individual components as well as the complete provided service, metrics must be defined at both the application level (including the external service access point) and the infrastructure level. The metrics can be annotated with the limits imposed by the SLA and also engineering rules of thumb (e.g. warning levels before capacity saturation). It is important that the metrics must be directly measurable (or trivially calculated from directly measured values) by the monitoring application supervising the actual system.

### C. *Prediction of Metric Values*

The above introduced model-based representation of the system enables "what-if" type analyses to be carried out on the model rather than experimentally measuring them on actual system. Accurately

predicting the performance of a complex application is a complex and generally unsolved problem, usually involving computationally costly simulation. Our application model is a very coarse grained abstraction of the software implementation, so it is insufficiently detailed for this kind of analysis.

In some cases however, acceptable estimates can be made. Such an example is network throughput utilization, and in some cases CPU and memory utilization as well. The current measured values can be used as a basis of prediction by assuming that the values are insensitive to which concrete infrastructure component is in use, in case there are no bottlenecks in the system. This way if the deployment mapping is changed the current values are simply transferred to the new component.

The bottleneck analysis can be carried out for example in the network domain by mapping the application's data and control flow relationships to the actual network connections and comparing the utilization values to the limits of the hardware represented by attributes of the infrastructure model. The throughput and the service payload performance is proportionally linearly scaled in case of a bottleneck, to give a very rough estimate on expected application performance. CPU sharing can also be handled this way, however memory saturation (swapping), and shared disk access exhibit significant nonlinear and non-monotonic characteristics (trashing) so the predicted results are not useful.

### *D. Reconfiguration Actions and System Response*

The elemental actions that change the deployment mapping [2] are defined as graph transformation rules, having a precondition pattern to be matched on the mapping model, and a postcondition pattern representing the result. These rules are applied with bound parameters, on exactly defined components.

Each action GT rule is accompanied with an appropriate command that can be carried out by an *actuator component* managing the real system. The command is assumed to be tested for correctness in advance.

Some components may autonomously react to reconfiguration actions and faults. A standby component in a fail-over cluster switches to active mode if the previously active component goes down. Such automatic behavior is modeled by GT rules applied in an "As Long As Possible" execution: these rules are kept being applied anywhere where their precondition pattern matches.

### *E. Anticipated Faults*

Faults can occur in the system at any time, causing errors and ultimately service level failures if no redundancy prevents this. Similarly overload conditions can occur, that can lower the performance to unacceptably low levels. Additionally the reconfiguration operations are not atomic and may fail during execution, leaving parts of the system unusable.

Some possible fault modes are known in advance, which are modeled by GT rules that can be applied one by one.

## III. Reconfiguration Decision Process

Reconfiguration is initiated explicitly by specifying a goal to be reached. A goal can be an externally initiated change in some metric to a new target level, or a response to an error condition, such as SLA violation or component fault. In either case a metric can be selected, whose value needs to be changed. The goal specification can be the output of a diagnostic engine logically inferring the fault causing propagated errors. The detailed discussion of such diagnostic engine is beyond the scope of this paper.

A selection has to be made from the set of possible applications of the reconfiguration rules, taking into account the possible consequences of the chosen reconfiguration step. The possible configurations of the managed deployment can be formalized as a state transition system, with a potentially unbounded state space. The transitions are applications of either an elemental reconfiguration rule or a fault event, both with their consequential automatic reactions.

The *MINMAX* family of decision algorithms more specifically $\alpha - \beta$ Pruning [3], traditionally used in artificial intelligence and game theory is adapted to our application. MINMAX algorithms select

the next step that minimizes the maximum loss of a player against an intelligent opponent. In our case the "opponent" tries to maximize the loss by choosing the anticipated fault with the worst possible consequence. This way the selection of the foreseeable best reconfiguration trajectory can be made that avoids threats to the highest possible extent during the execution.

MINMAX algorithms optimize the outcome of a single scalar variable, so a single *composite metric* must be calculated from the set of available metrics. The following metric composition function is proposed:

- The range of each scalar metric ($M_i$) is divided into discrete intervals ($I_j^{M_i}$) based on limits imposed by SLA.
- Boolean terms ($B_{M_i,I_j}$) are formulated from elemental clauses of a metric value being in an interval or not ($Value(M_i) \in I_j^{M_i}$).
- The terms are assigned into *priority classes* ($C_i$) based on their relative importance.
- The metric selected for improvement by the goal ($M_g$) is not quantized, instead its difference from the targeted value is taken ($T = |Target(M_g) - Value(M_g)|$). It is also assigned to an importance class ($C_k$).
- The composite metric is calculated. The boolean AND is taken of all terms assigned into each priority class, the score ($2^i$) is only added to the final sum, if all requirements in the class are satisfied. This is done for priorities both higher and lower than the metric selected for improvement. $\sum_{i=0}^{k-1} 2^i [\bigcap_{B_j \in C_i} trueB_j] + 2^k T + \sum_{i=k}^{n} 2^{i+\lceil log_2 Max(T) \rceil} [\bigcap_{B_j \in C_i} trueB_j]$.

The composition function preserves the importance so an SLA metric constraint with a high priority cannot be disregarded in favor of any good combination of lower priority constraints. Also the quantization to intervals ensures that only limit crossing has significance in all metrics except for the one that is being improved. Metrics without SLA limits are completely disregarded.

The coarse estimates produced by the prediction model, then subjected to the composing function, can be acceptable as a heuristic input for the MINMAX decision. The monotonic behavior of the predicted values still has to be ensured to make a good heuristic. Adding high priority constraints to problematic metrics (memory consumption, disk usage) prevents them from getting near overutilization levels so that the model evaluation does not return significantly unrealistic estimates.

In each iteration the first step is executed that has the highest possible score from the MINMAX tree lookahead. The result of the execution is monitored and evaluated in a new round of diagnosis and decision, no memory is preserved from the previous executions.

## IV. Conclusion and further work

This paper presented an approach that uses MINMAX search and model-based predictive heuristics for controlling structural reconfiguration decisions. This research is still in early stage, an in-depth evaluation of the current proof-of-concept needs to be performed to find out the practical feasibility of this approach. Particularly the scalability of the $\alpha - \beta$ pruning and the prediction execution need to be tested on large models typically found in IT infrastructures. The prediction also needs further research, as of yet it is only demonstrated for a few specific metrics. Furthermore the current decision process takes neither the cost of the reconfiguration actions nor any possible standardized workflow sequences into account.

## References


[1] U.K. Office of Government Commerce, *Information Technology Infrastructure Library (ITIL) version 3*, http://www.itil.org.

[2] L. S. J. X. E Farr, R Harper, "A case for high availability in a virtualized environment," *Availability, Reliability and Security*, pp. 675–682, 2008.

[3] D. E. Knuth and R. W. Moore, "An analysis of alpha-beta pruning," *Artificial Intelligence*, 6(4):293–326, 1975.

# DIRECT CONNECTION OF HIGH CAPACITY MASS STORAGE TO HARDWARE ACCELERATORS

**György DANCSI**
**Advisor: Béla FEHÉR**

## I. Introduction

An important application of the FPGA devices is the reconfigurable computing, where they are used to realize algorithms in a custom hardware. The accelerator module is usually not a standalone-device, but a specialized part of a system, which is built form general purpose pieces (memories, buses, CPUs). The resultant computing performance of the systems depends on the components and the interconnections. In data intensive computing the speed of the interconnections became decisive. For example in molecular biology there are quite huge data sets (hundreds of gigabytes) as primary input source and many temporary results arise. To eliminate the overhead associated with the I/O bus, the memory bus and the operation system dedicated communication channels should be used. For this purpose an architecture that supports direct connection to disk at high data-rate is evaluated using reconfigurable hardware. This storage subsystem is attached to the processing units through FIFOs. The following restrictions are applied to the disk usage:

- The processing units use block data processing. The data blocks stem from one ore more storage units (file) of the disk
- Similarly the results need to be written back to the disk into storage units

A high speed, low latency interface should be chosen to match the functionality.

## II. Serial ATA Interface

The Serial ATA computer bus has the primary function of transferring data between the main memory of the *host* computer system and mass storage *devices* (such as hard disk drives, optical drives and solid-state drives). Designed as a successor to the Advanced Technology Attachment standard (ATA also known as IDE or EIDE), it is expected to eventually replace the older technology. Serial ATA adapters and devices communicate over a high-speed serial cable.

### *A. Features*

The current SATA specification [1, 2] can support data transfer rates as high as 3.0 Gbit/s per device. Taking into account 8b10b coding overhead, they have an actual uncoded transfer-rate of 300 MB/s. SATA supports hot-swapping, Native Command Queuing (NCQ), Enclosure Services, using Port Selector and Port Multiplier. The following layers are specified:

- Physical layer: serialization/deserialization, synchronizing, data and clock recovery, link initialization with out-of-band (OOB) signaling
- Link layer: 8b10b encode/decode, scrambler/descrambler, CRC calculate/check, primitives and frames, flow control
- Transport layer: TaskFile shadow registers, Command and Control protocol; PIO, DMA and First-party DMA data transmission protocol

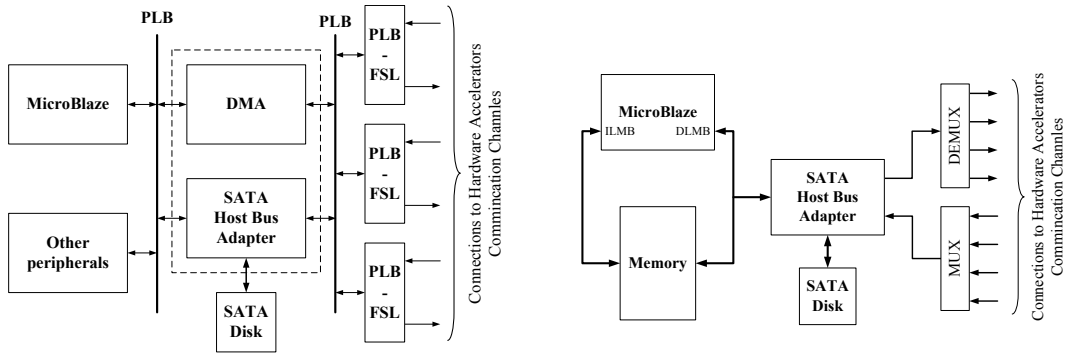
### *B. Developing Host Bus Adapter IP*

The Host Bus Adapter (HBA) includes the above three layers and a bus interface. The HBA is developed in Xilinx Virtex-5 platform. These devices contain RocketIO GTPs that eliminate the need for external physical chip to implement the Physical layer. In the first design the GTP is configured for

Serial ATA operation and the Link layer is implemented in hardware. It connects to the MicroBlaze processor via Fast Simplex Link (FSL).

## III. Proposed Architectures

Two possible architectures are elaborated to satisfy the requirements. Fig. 1 shows these block diagrams. The main aspect is the direct data movement between the disk and the processing units bypassing the processor. At the (a) solution it is provided with a DMA controller and the (b) solution applies point-to-point connections in both direction. The HBA has another independent interface for the MicroBlaze processor that allows overlapped operation. The previous transmission can be acknowledged or the next transmission can be prepared in the same time with the transfer of payload data. The tasks of the processor are handling file system and checking the FIFOs of processing units.




(a) PLB interface (b) LMB interface

Figure 1: Possible architectures to connect Serial ATA Host Bus Adapter to processing units

The first concept uses the built-in buses and peripheral devices of the Platform Studio development system. The storage subsystem contains two independent Processor Local Buses (PLB) so the processor can access the Task File or programming the DMA controller without breaking the data movement.

The second concept differs from the first one in the processor interface and it uses point-to-point architecture for data movement in place of a bus system. The TaskFile is mapped directly in the data bus of MicroBlaze via Data Local Memory Bus (DLMB). The TaskFile is defined as a structure in the program code and the linker script ensure the structure is placed to the proper address during compiling.

The (a) solution offers more flexibility, allows keeping the HBA relatively simple and creating the system level design in graphical development environment, but it uses more hardware resources because of bus interfaces. With the (b) solution higher data-rate and bidirectional operation can be achieved.

## IV. Results and Future Work

A simple host interface without real Transport layer is implemented to test the connection with the disk. The proper operation is verified with executing basic ATA commands for example IDENTIFY DEVICE and some power management instructions. Currently the development and verification of Transport layer is in progress. Testbenches are used to simulate transaction is in accordance with non-data, PIO and DMA protocol of Serial ATA. To check more complex cases an improved verification methodology need to be used, typically the assertion based verification.

## References

[1] SerialATA Workgroup, *Serial ATA: High Speed Serialized AT Attachment Revision 1.0a*, 3 Jan. 2003.

[2] SerialATA Workgroup, *Serial ATA II: Extensions to Serial ATA 1.0a*, 27 Aug. 2004.

# QUASI COHERENT DETECTION ALGORITHM FOR UWB IMPULSE RADIO

**Tamás KRÉBESZ**
**Advisor: Géza KOLUMBÁN**

## I. Introduction

Regulations for Ultra-Wideband (UWB) radio specify only the maximum value of psd and the minimum bandwidth over the assigned UWB frequency band that goes from 3.1 GHz to 10.6 GHz. Neither the type of the carrier nor the modulation technique are defined. The psd of Equivalent Isotropically Radiated Power (EIRP), measured with a resolution of 1 MHz, must be below -41.3 dBm [1]. One-to-one recovery of UWB carriers is not feasible, consequently, a pure coherent UWB receiver cannot be built. On the other hand, noncoherent detectors offer a bad noise performance. As a trade-off, a new near-coherent detector configuration is proposed here where there is no need for an exact recovery of UWB carrier to perform detection.

## II. Modulation schemes used in UWB radio

The waveforms used to carry digital information are ultra wideband signals in UWB radio with 500-MHz minimum bandwidth. Fixed or chaotic carriers can be used to transmit information. The former is a deterministic signal while the latter is a continuously varying wavelet [2]. This paper considers the fixed waveform UWB radio.

### *A. Modulation using fixed waveform*

In the simplest case, one bit information is carried by one fixed UWB wavelet. The structure of this UWB signal is shown in Fig. 1 where $g(t)$ denotes a fixed but arbitrary wavelet, used as carrier, in the pulse bin $T_{bin}$. Pulse bin determines the time elapsed between two consecutive wavelets, i.e. the symbol rate. Its role is to prevent inter-symbol interference in a multipath channel. Observation time $T_{obs}$ determines the interval during which the detector observes the received signal. The position $t_{pos}$, the amplitude and the sign of the waveform may be varied in accordance with the digital information to be transmitted but the best noise performance is offered by an antipodal modulation scheme. Therefore, Pulse Polarity Modulation (PPoM) is considered here where the information is mapped into the sign of the radiated deterministic signal.

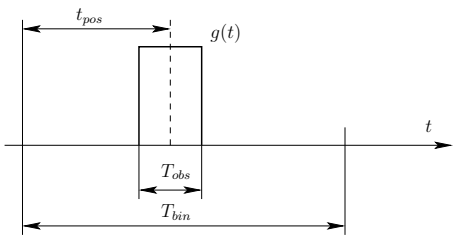



Figure 1: Waveform structure of UWB impulse radio.

### *B. Deterministic wavelets used in UWB impulse radio*

Since regulations specify only the psd mask and minimum bandwidth of UWB wavelet, there is a high degree of freedom in generating UWB waveforms.

In this paper the bell-shaped Gaussian pulse [1] is investigated where the bandwidths are 2 GHz and 500 MHz. The only basis function takes the form

$$g(t) = \sqrt{\frac{2E_b}{\sqrt{\pi}u_b}} \exp\left(-\frac{t^2}{2u_B^2}\right) \cos(\omega_C t) = v(t)\cos(\omega_C t) \tag{1}$$

consequently, the elements of PPoM signal set are $s_{1,2} = \pm g(t)$. The carrier $f_C = \omega_C/2\pi$ is the UWB center frequency, $E_b$ denotes the energy per bit, and $u_B$ is a constant determined by the 10-dB bandwidth of UWB wavelet [1]. Depending on their bandwidths (i) wideband, 2 GHz and (ii) narrowband, 500 MHz UWB systems are distinguished. The bell-shaped Gaussian wavelets are shown

in the time domain for the wide- and narrowband UWB systems in Fig. 2. Although the duration of Gaussian waveforms is infinite the power decreases rapidly as a function of time, consequently, the optimum duration of observation time $T_{obs}$ cannot be determined from the UWB wavelet, it is obtained from the noise performance of the PPoM modulation scheme. By definition, observation time assuring minimum BER is considered as $T_{obs}$.

To implement a near-coherent correlation receiver, a reference signal has to be generated. Because of their spectral shape, the UWB carriers cannot be recovered. Instead, a noise-free reference signal approximating $g(t)$ over $T_{obs}$ as close as possible should be found that may be recovered from the received UWB waveform by implementable circuitry.

As shown in Fig. 2, less than two cycles of carrier are transmitted in wideband UWB radio. The energy of such a signal is concentrated in the main lobe, therefore, the detection may be performed by windowing the received UWB waveform by a square-wave template signal [2]. This technique, referred to as template detection, is used if the energy of UWB signal is spread over a few carrier cycles.

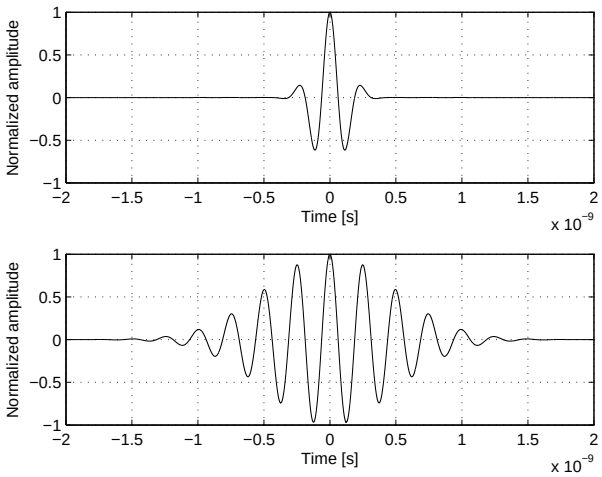



Figure 2: Bell-shaped Gaussian pulses with 2-GHz (upper trace) and 500-MHz (lower trace) bandwidths for $f_C$=4 GHz.

In narrowband UWB radio, the UWB wavelet contains about 10 carrier cycles as shown in Fig. 2 for $f_C = 4$ GHz. In this case a sinusoidal signal recovered by a phase-locked loop (PLL) can be considered as a reference signal. The detection of narrowband UWB signal using a sinusoidal reference is referred to as near-coherent technique.

## III. Near-coherent detection technique

In narrowband UWB impulse radio the transmitted signal contains enough cycles of the carrier to be recovered by a PLL, i.e. a sinusoidal reference $c(t)$ is used in the coherent correlation receiver. At the receiver a gated phase-locked loop (G-PLL) structure is used to recover $c(t)$.

The block diagram of near-coherent UWB detector proposed here for the narrowband UWB systems is shown in Fig. 3, where $r_m(t) = s(t) + n(t)$ is the received noisy signal, $c(t)$ denotes the *quasi-recovered* carrier, $z_m$ is the observation variable and $\hat{b}_m$ denotes the recovered digital information. The appellation *quasi-recovered* carrier refers to the main characteristic of near-coherent detection: although both the $r_m(t)$ and $c(t)$ have the same angle, the former has Gaussian envelope while the latter is the output of a G-PLL therefore its envelope is constant. This mismatch in shape results in performance degradation. The information $\hat{b}_m$ can be recovered by correlating $r_m$ with the sinusoidal reference $c(t)$.

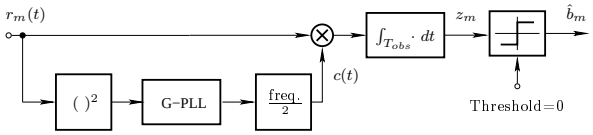



Figure 3: Block diagram of near-coherent UWB detector.

Important advantage of near-coherent detection is that $c(t)$ is a noiseless signal matched to the angle of $r_m(t)$

$$c(t) = \begin{cases} \sqrt{\frac{2}{T_{bin}}}\cos(\omega_C t), & \text{if } |t| < \frac{T_{obs}}{2} \\ 0, & \text{otherwise} \end{cases}$$

The received signal $r_m(t)$ is a modulated waveform. For the recovery of carrier first the modulation has to be eliminated that is performed by squaring the received signal. The spectrum of a UWB signal carrying a random modulation may have no carrier component. However squaring the modulated signal removes the modulation and generates a carrier component at $2f_C$

$$(\pm g(t))^2 = (\pm 1)^2 v^2(t)\cos^2(\omega_C t) = v^2(t)\frac{1+\cos(2\omega_C t)}{2} \tag{2}$$

This $2f_C$ frequency component is selected by a G-PLL and then a frequency divider is used to generate the $f_C$ frequency reference signal.

## IV. Noise performance of near-coherent detection

The model for investigation of the noise effects on the received signal is shown in Fig. 4. Since the transmitted signal is corrupted by white Gaussian noise in an additive manner in the telecommunication channel, the observation signal, a random variable that appears at the output of the correlator can be expressed as

$$z_m = \int_0^{T_{obs}} s_m(t)c(t)\mathrm{dt} + \int_0^{T_{obs}} n(t)c(t)\mathrm{dt} \tag{3}$$

Recall, that $c(t)$ is noiseless, consequently, $n(t)$ in (3) goes under a linear integral transformation. Therefore the distribution of the second term on right hand side remains Gaussian and its expected value is zero [3]. The noise performance is determined from the observation variable. For antipodal modulation scheme using one basis function, the probability of bit error is given in the literature [4]

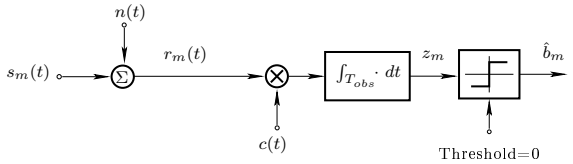



Figure 4: Model for investigation of the noise effects on the received signal.

$$P_e = \frac{1}{2}\mathrm{erfc}\left(\frac{\mu_m}{\sqrt{2\sigma_n^2}}\right) \tag{4}$$

where $\mu_m$ is the expected value of observation variable and $\sigma_n^2$ is the power of noise, i.e. the variance of $z_m$. To get the BER for near-coherent UWB detection $\mu_m$ and $\sigma_n^2$ have to be found.

Since the second term on the RHS of (3) has zero mean, the mean of $z_m$ is obtained as

$$\mu_m = \mathrm{E}\left[\int_0^{T_{obs}} s_m(t)c(t)\mathrm{dt}\right] = \sqrt{\frac{E_b\sqrt{\pi}u_B}{T_{bin}}}\mathrm{erf}\left(\frac{T_{obs}}{2u_B}\right) \tag{5}$$

where $E[.]$ is the expectation operator.

The channel noise $n(t)$ is modeled by a zero-mean stationary Gaussian process that has a uniform two-sided psd of $N_0/2$. The variance of the observation variable is obtained as [3]

$$\sigma_n^2 = \int_0^{T_{obs}}\int_0^{T_{obs}} R_n(t_1,t_2)c(t_1)c(t_2)dt_1dt_2 \tag{6}$$

where $R_n(t_1,t_2)$ denotes the autocorrelation function of $n(t)$. Since $n(t)$ is stationary and white, its autocorrelation function is

$$R_n(t_1,t_2) = R_n(t_2-t_1) = \frac{N_0}{2}\delta(t_2-t_1) = \frac{N_0}{2}\delta(\tau) \tag{7}$$

Substituting (7) into (6), the variance of $z_m$ is expressed as

$$\sigma_n^2 = \int_0^{T_{obs}}\int_0^{T_{obs}} \frac{N_0}{2}\delta(t-\tau)\frac{2}{T_{bin}}\cos(\omega_C t)\cos(\omega_C\tau)\,dt\,d\tau = \frac{N_0}{2}\frac{2}{T}\int_0^{T_{obs}}\cos^2(\omega_C t)\mathrm{dt} = \frac{N_0}{2}\frac{T_{obs}}{T_{bin}} \tag{8}$$

Substituting (8) and (5) into (4), we get the BER for near-coherent detection

$$BER = \frac{1}{2}\text{erfc}\left(\sqrt{\frac{\sqrt{\pi}u_B\left[\text{erf}\left(\frac{T_{obs}}{2u_B}\right)\right]^2}{T_{obs}}}\sqrt{\frac{E_b}{N_0}}\right) \tag{9}$$

Equations (5) and (8) show that both the mean and variance of observation variable depend on the observation time. Consequently, the BER given by (9) also depends on $T_{obs}$, its value may be minimized by choosing the optimum value of $T_{obs}$. Figure 5 shows the BER as a function of $T_{obs}$. Note, a global minimum exists at $T_{obs} = 1$ ns. This means that (9) can be minimized as a function of $T_{obs}$, its optimum value for our case is at $T_{obs} = 1$ ns.

Note, that any deviation from the optimum value of the observation time, results in a considerable performance degradation. For example, if the observation time is increased from 1 ns to 2 ns then BER also increases, from $10^{-8}$ to $10^{-6}$.

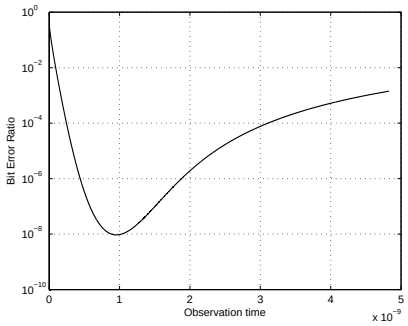



Figure 5: BER as a function of $T_{obs}$ for $E_B/N_0 = 14$ dB.

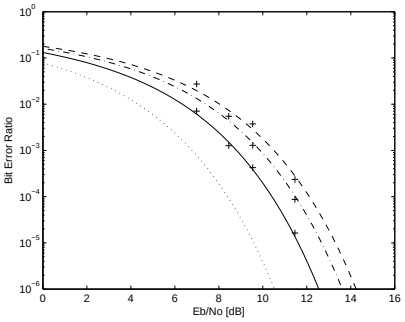



Figure 6: BER of BPSK (dotted), near-coherent detection for optimum $T_{obs}$ (solid), $2T_{obs}$ (dashed) and $0.5T_{obs}$ (dash-dotted).

The theoretical results have been validated by computer simulations. The results of BER simulations are marked by crosses in Fig. 6.

The noise performance of the new near-coherent detector is shown in Fig. 6 for optimum and increased/decreased observation times. The BER of BPSK is also shown for comparison. The transmitted carrier is not perfectly recovered in the proposed approach, instead, the recovered reference signal $c(t)$ is matched only in angle to the transmitted one. This mismatch error reduces the mean of $z_m$, therefore, the noise performance of near-coherent UWB detector always lags behind that of BPSK.

## V. Conclusions

A new near-coherent detection technique has been proposed for narrowband UWB impulse radio. A gated phase locked loop is used for the *quasi-recovery* of the basis function. An optimum $T_{obs}$ has been found that offers the best noise performance. Simulation results validate the closed form expression given for BER.

## References


[1] K. Siwiak and D. McKeown, *Ultra-Wideband Radio Technology*, Wiley, Chichester, UK, 2004.

[2] G. Kolumbán and T. Krébesz, "UWB radio: A real chance for application of chaotic communications," in *Proc. NOLTA'06*, pp. 475–478, Bologna, Italy, September 11–14 2006.

[3] J. S. Bendat and A. G. Piersol, *Measurement and Analysis of Random Data*, John Wiley & Sons, New York, 1966.

[4] S. Haykin, *Communication Systems*, Wiley, 3rd edition, 1994.

# IMAGE PROCESSING USING RECONFIGURABLE HARDWARE

**Tamás RAIKOVICH**
**Advisor: Béla FEHÉR**

## I. Introduction

FPGA based hardware accelerators have become more and more important for bioinformatics applications. These applications use wide range of algorithms, including searches in large databases, sequence alignment, statistical analysis and image processing. A part of these algorithms can be efficiently accelerated using FPGA devices.

Biological and biomedical experiments like microarray experiments [1], high-content screening (HCS) or cellular microscopy [2] result in large amount of image data. These data have to be processed to evaluate the experiments. There are different applications available for this purpose, for example the CellProfiler [3], which can process and analyze cellular microscopy images.

The final goal would be an FPGA based hardware accelerator for (biomedical) image processing purposes. This would include several reconfigurable execution units, which would enable the parallel processing of the data. By using partial dynamic reconfiguration [4], the execution units could be quickly reconfigured with the necessary algorithms.

This paper introduces a reconfigurable test system and two basic image processing functions implemented with FPGA. These functions are a 3 x 3 median filter and a 3 x 3 FIR filter, which can process 8 bit grayscale images.

## II. The reconfigurable system

### *A. Hardware*

The design is implemented on the Xilinx ML506 board. This board utilizes an XC5VSX50T FPGA (Virtex-5 SXT family), which is optimized for DSP and memory-intensive applications. It contains increased number of Block-RAMs and DSP48E slices and its capacity enables to implement more complex designs in the future. The ML506 board has wide range of communication interfaces to exchange data with the PC, for example serial ports, USB port, PCI-Express x1 interface and 10/100/1000 Ethernet interface. The Ethernet interface has been chosen because it requires less development effort and it provides enough communication speed. Fig. 1 shows the block diagram of the reconfigurable system. The system clock frequency is 100 MHz.

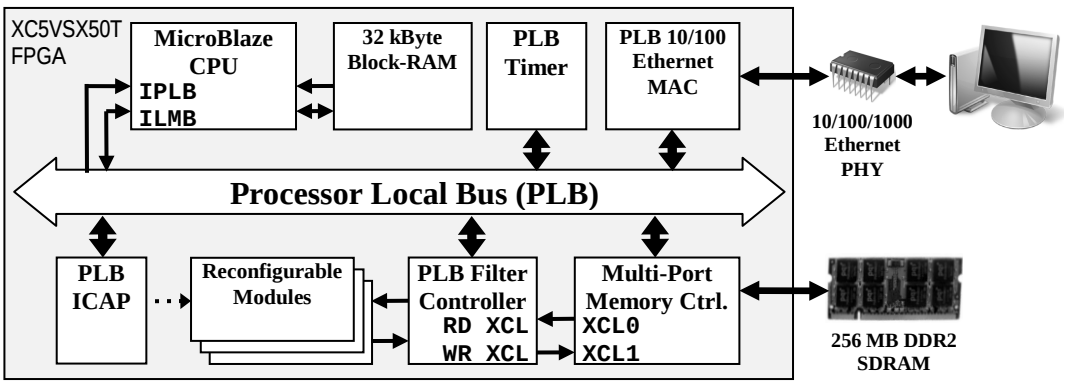



Figure 1: Block diagram of the reconfigurable system

The Virtex-5 SXT FPGAs don't contain embedded PowerPC processor therefore the MicroBlaze soft processor is used in the design. The 32 kB Block-RAM stores the software running on the

processor and it is connected to the MicroBlaze CPU through the LMB bus. The 256 MB external memory stores the original and the filtered images, as well as the configuration data for the reconfigurable modules. Because the limited number of the internal block-RAMs, using fast and high capacity external memory is the efficient way to store these data. The multi-port memory controller provides access to the external memory. The memory controller has 8 ports: one port is connected to the PLB bus, the other ports are available to the reconfigurable modules. The Ethernet MAC peripheral is necessary for the Ethernet communication. The PLB timer peripheral has two timer registers. One timer is required by the TCP/IP stack, the other timer is used to measure the reconfiguration and the filtering time.

The PLB ICAP peripheral is a master peripheral connected to the PLB bus. It handles the internal configuration access port (ICAP) of the Virtex-5 FPGA and can be used to update the configuration of the reconfigurable modules. Before starting the dynamic reconfiguration process, the start address and the data length have to be written into the proper registers. Then the peripheral reads the configuration data from the SDRAM and writes it into the ICAP. The maximum allowed clock frequency of the ICAP is 100 MHz. When the system clock frequency is greater than 100 MHz, the peripheral includes a DCM module to synthesize the 100 MHz ICAP clock and the synchronization logic required to connect the two sides working at different clock frequencies.

The PLB filter controller peripheral provides the connection between the system and the filter module in the reconfigurable region. The internal filter and peripheral registers can be accessed through the PLB slave interface. In the current design, the Xilinx CacheLink (XCL) interface has been chosen for data transfer between the filter and the external SDRAM. The XCL is a simple FIFO based point-to-point interface, it provides 1, 4, 8 or 16 word length read and write transfers. The filter controller has one read and one write XCL interface, which are connected to the multi-port memory controller.

### *B. Software applications*

The main function of the software for the MicroBlaze processor is to handle the TCP/IP communication and to control the reconfiguration and the filtering. This software uses the uIP 1.0 TCP/IP protocol stack. A simple Windows application has been created for the PC side. This application can be used to download the configuration data and the images to the target system, modify the filter coefficients and read back the filtered images. It also implements the software version of the filters. This allows comparing the hardware and the software filtering times.

## III. Implementation of the filters

The 2D median and the 2D FIR filtering algorithms are called local image processing algorithms, because they depend on data from a relatively small neighborhood compared to the image size. These algorithms use a sliding window (Fig. 2) and execute different operations on the pixels found in the window. The filters in this design use a 3 x 3 window and can process 8 bit grayscale images with a maximum size of 1024 x 1024 pixels. The image size is adjustable.

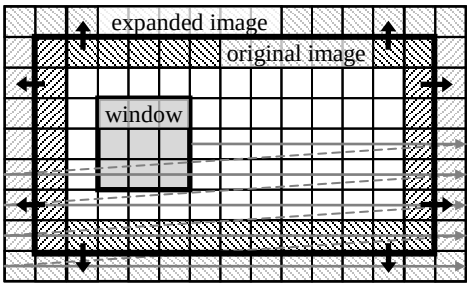



Figure 2: Moving the window on the image

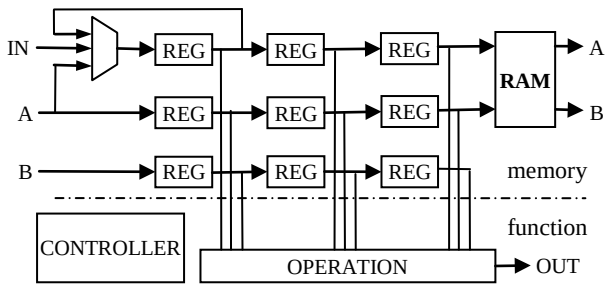



Figure 3: Block diagram of the filters

The sliding window image buffer can be easily implemented in hardware as shown in Fig. 3. Registers store the pixels in the window, the remaining pixels (image width – 3) of the rows are stored in a dual-port Block-RAM. When a new data is written into the buffer, the window is shifted right one position.

As the window moves through the image, its center covers a smaller area than the original image, which results a smaller filtered image size. This problem can be solved in different ways. In this design, the filters process an expanded image shown in Fig. 2. It contains the original image and its edges are the copy of the edges of the original image. This expanded image can be easily created in hardware by adding a multiplexer to the image buffer input, which can select the following write operations:

- reloading the previously written row into the buffer (at the 2nd and last rows)
- reloading the previously written pixel into the buffer (at the 2nd and last columns)
- writing the next pixel of the original image into the buffer (otherwise)

### *A. 2D median filter*

Median filtering can efficiently reduce or remove "salt & pepper" noise from the images. Its edge-preserving nature makes it useful in cases when edge blurring is undesirable. Median filters sort the pixels in the window by intensity and the center element of the window is selected as the output.

In this design, the median element is determined by the following way. At first, each row of the window is sorted (stage A). In the next step, each column of the window is sorted (stage B). At last, the elements in the secondary diagonal are sorted (stage C). The median element in the secondary diagonal will be the median element of the original inputs. The fully parallel and pipelined sorting network can be seen in Fig. 4.

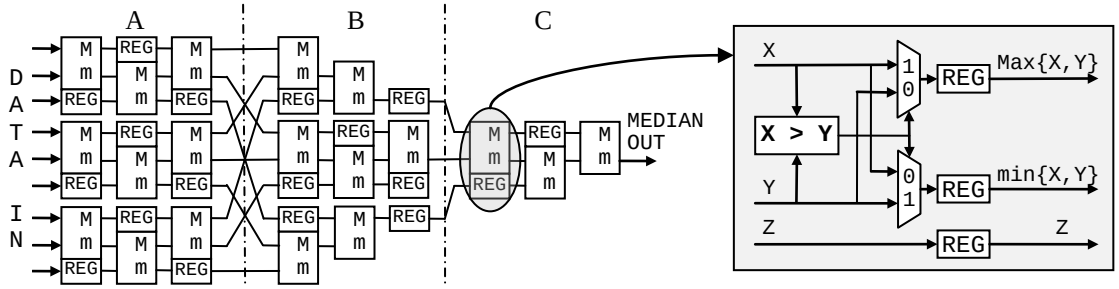



Figure 4: Network that determines the median element

### *B. 2D FIR filter*

The operation executed by the FIR filters is convolution, which is defined in Eq. (1). The output of the filter is the weighted sum of the elements in the window.

$$f(x,y) = \sum_{i=-1}^{1}\sum_{j=-1}^{1} W_{i,j} \cdot image(x+i, y+j) \qquad (1)$$

By selecting the appropriate weights, convolution can implement low-pass, high-pass and band-pass frequency domain filters. Low-pass filters use positive weights and are used for image smoothing. High-pass filters use a positive center weight and negative outer weights and are used to enhance high frequency components in an image such as edges and fine detail. All of the filter coefficients ($W_{i,j}$ weights) are adjustable.

The parallel implementation of a 3 x 3 FIR filter requires nine multipliers and eight adders, both are registered at the outputs. The adders are organized in tree form to reduce the pipeline stages. The last pipeline stage is the saturation unit, which saturates the output when overflow occurs. Negative output values are possible when there are negative $W_{i,j}$ weights. These output values are meaningless and are replaced with zeroes.

## IV. Results

Table 1 shows the logic utilization of the reconfigurable region of the FPGA. The reconfigurable region is larger than the required size, only 24% of the logic resources are used by the filters. Decreasing the reconfigurable region size would shorten the reconfiguration time. Unfortunately, in case of more complex functions, more resources are required and these functions may not fit into the smaller reconfigurable regions. The size of the reconfigurable regions can be optimized when more image processing modules will be available.

Table 1: Reconfigurable region utilization

| Module name | Configuration file size (bytes) | Reconfigurable region utilization | | | |
|---|---|---|---|---|---|
| | | LUT | Flip-flop | RAMB36 | DSP48E |
| Median filter | 146396 | 688 / 2880 (24%) | 551 / 2880 (19%) | 1 / 12 (8%) | 0 / 32 (0%) |
| FIR filter | 142779 | 473 / 2880 (16%) | 517 / 2880 (18%) | 1 / 12 (8%) | 9 / 32 (28%) |

Table 2 shows the reconfiguration and filtering time. As for the values in the "theoretical" columns of the table, it is assumed that every new input data is processed in one clock cycle. The actual values are about three times greater than the theoretical values. Because the filters are pipelined, they can process new data at every clock cycle. Therefore the bottleneck is the memory access and it have to be optimized in future designs to achieve better results. If the hardware and the software filtering times are compared, the FPGA design performs better. In case of 2D median filtering, the speed-up is 20x. In case of 2D FIR filtering, the speed-up is 8.6x. The ratio of the reconfiguration and the filtering times highly depends on the image size. To reduce the number of reconfigurations, it is suggested to process as many images as possible with a module before the module is reconfigured.

Table 2: Reconfiguration and filtering times

| Module name | Reconfig. time @ 100 MHz | | Filtering time (512 x 512 image size) | | |
|---|---|---|---|---|---|
| | | | FPGA @ 100 MHz | | PC @ 2,8 GHz |
| | theoretical | actual | theoretical | actual | |
| Median filter | 0.366 ms | 1.05 ms | 2.62 ms | 8 ms | 160 ms |
| FIR filter | 0.357 ms | 1.03 ms | 2.62 ms | 7.2 ms | 62 ms |

Fig. 5 shows some images used for testing the filters. The task is to enhance the first noisy image. This process is very sensitive to the noise. When the noise is not removed, it will be also enhanced as shown in the second image. The median filter can be used to remove the noise from the first image as shown in the third image. The FIR filtered third image (final result) can be seen in the last image.

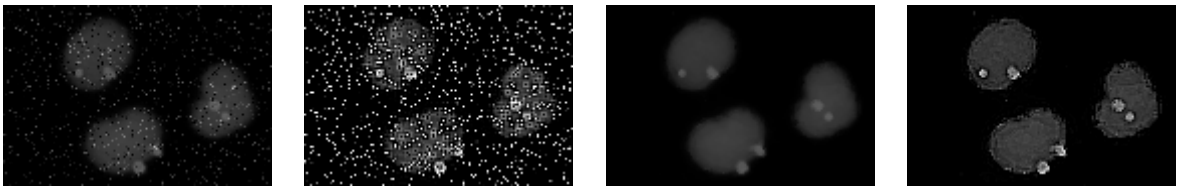


Figure 5: Input and filtered images

## References


[1] E. Wit, J. McClure, *Statistics for Microarrays*, Wiley, 2004.

[2] Q. Wu, F. A. Merchant, K. R. Castleman, *Microscope Image Processing*, Elsevier, 2008.

[3] CellProfiler cell image analysis software manual
URL: http://www.cellprofiler.org

[4] T. Raikovich, "Dynamic Reconfiguration of FPGA devices," in *Proc. of the 15th PhD Mini-Symposium*, pp. 72-73, Budapest, Hungary, February 4–5 2008.

# TESTING OF AUTOMATIC TRANSMISSION CONTROLLER SOFTWARE USING ARTIFICIAL INTELLIGENCE METHODS

**Balázs SCHERER**
**Advisor: Gábor HORVÁTH**

## I. Introduction

Software systems are among the most complex and least reliable technological products. Software engineers are currently not able to test software enough to ensure its correctness. Exhaustive software testing is impossible and currently there isn't a way for non-exhaustive testing that ensures that a software system will perform as intended. Developing software that can be marked reliable requires much effort. Statistics showed that projects should expect 50% of their resources to be expended on testing, in the case of safety-critical systems, this amount could rise to 80% [1].

Automotive embedded software modules are typical examples of safety-critical systems. Subsystem providers like Robert Bosch GmbH. have to ensure the trustiness of their products for 10 to 15 years in a very harsh environment. The software development of Bosch automatic transmission control ECU-s (Electronic Control Units) or with other words TCU-s (Transmission Control Unit) [2] is done in Budapest at the GS-TC/ENC-Bp department. To satisfy the above reliability goals this department employs more Test engineers than Software engineer.

The goal of my PhD. work is to find methods to improve software quality of embedded systems by using artificial intelligence methods. The methods developed in this PhD. work will be used at GS-TC/ENC-Bp for testing the software of TCU modules. Therefore the special features of this environment are noted in this paper.

## II. Software testing and Artificial Intelligence

Test methods are sorted into three main categories: White Box, Black Box and Grey Box (mixture of Black- and White Box tests) tests. White Box methods are based on the source code of software, and these are the most effective ones. Most of these tests are coverage tests [3], like instruction coverage, branch coverage, and decision coverage. These coverage tests ensures that every instruction of the code is executed, each control structure (such as an IF statement) evaluated both to true and false etc.

However White Box methods are very effective ones, but also have their limits. Companies like Bosch usually use separate test centers, or test teams; therefore the tester in most of the cases has no clear overview about the tested code. Complex embedded software systems like TCU software are developed usually by more than one manufacturer, and these manufacturers do not share their source codes, which limit the scope of these tests. A drawback of White Box methods that these tests are very time demanding ones, and therefore the most complex coverage tests like decision coverage and path coverage are only recommended, but not mandatory steps, even in very high reliability systems such as SIL3 [4] systems (typical SIL3 systems are brakes, airbag, etc.).

Black Box tests do not need the internal structure of the system, and are used widespread to complement White Box methods. These methods are based on functional specification of unit under test and examine whether or not the software conforms to its specifications. User acceptance tests always take this form, and it is also useful for developers. Typical Black Box tests examine the input and output variables of the systems, and use limit checking or process identification methods to detect faults and errors. Artificial intelligence methods are often used as Black Box tests, like Fuzzy clustering for limit checking and Artificial Neural Networks for process identification [5], [6].

Despite the relatively complex literature of this field there are many uncovered problem as will be shown later.

## III. Black Box testing of a typical embedded system software

At GS-TC/ENC-Bp White Box and Black Box tests are also used in the software development process. A typical Black Box test is the Regression test. Regression test check the differences to previous software release. Regression test at GS-TC/ENC-Bp consists of the following main sub-module tests: I/O, CAN, Diag, Comm-diag, OMM, ROS.

The I/O test is responsible for testing the power supply and the analog and digital interfaces of the TCU. A general TCU has about 60 such I/O lines. Test measurements made on the power supplies of the sensor and actuator units, the proper working of the solenoids and ignition controls are also investigated.

The CAN test contains measurements for the CAN (Controller Area Network) [7] communication of the TCU. This test step checks the presence and periodicity of CAN messages (there are about 10 CAN messages containing more than 20 important signals), and whether the system is able to notice the absence of these messages.

Diag. test deals with the so called filtering and de-filtering of different errors, this test checks whether the system is able to notice and store errors.

Comm-diag subprocess provides test steps for the diagnostic communication. In modern cars it is possible to read out some performance and malfunction related notes from the ECUs in a repair station using a simple diagnostic device. This step ensures that every diagnostic function (like KWP2000 [8] or UDS [9] commands) works properly.

The OMM (Operating Mode Management) test step is responsible for checking the transition between the main operating states of the TCU, like initialization, drive, limp-home, shut-down etc.

The ROS (Realtime Operating System) tests the behavior of the RTOS by checking the stack usage, the periodicity of tasks and so on.

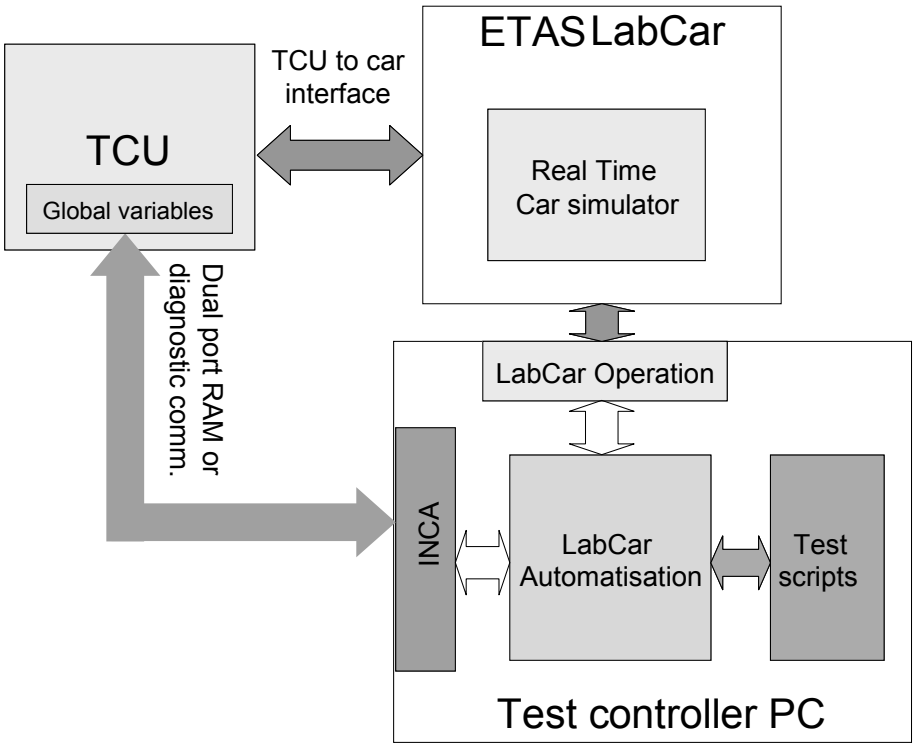



Figure 1: TCU testing environment

Each of the tests above is done in nearly the same environment. This environment contains a TCU, a so called LabCar, which simulates the behavior of the car's other ECU-s, and an ETAS INCA interface and software (Fig. 1.). The most important part of this set-up is the ETAS INCA [10] software and hardware. Briefly this tool provides an interface to the TCU (based on a dual port RAM or diagnostic communication), and by using this interface, every important internal variables of the TCU software can be examined on the fly. Most of the above tests are done by checking, whether these internal variables contain the right values.

## IV. Goal and application space of a self learning based test system

As the previous chapter has shown there are extremely complex test processes at GS-TC/ENC-Bp, and many of these processes are under automation. The Regression tests are Black box tests focusing on one individual software module.

The main advantage of the self learning test could be the opportunity to handle the whole system, and test fast any unwanted interactions between individual modules. Other complement to traditional tests could be that these tests are rather static ones focusing only to a change of one variable in a given conditions. Self learning test can introduce a more dynamic drive cycle based test vector, where the system modules and their interactions can be investigated in a different situation. Therefore the goal of the self learning system is to make faster or reliable the Regression tests.

## V. Difficulties of testing complex embedded software systems

As the first step to create a Black Box based self learning test, the inputs and outputs of the system should be identified. The first problem of the TCU testing example is that the collections of these variables are very difficult. At GS-TC/ENC-Bp TCUs are developed for more than 5 car manufacturers, and there are several TCU versions for each manufacturer. Unfortunately the set of variables and their mapping are highly TCU version dependent.

The next step of designing such a test system is to identify features, which represents the system. The correctness of the system could be determined by measuring these features. In the case of the TCU example, the names and precisions and sometimes the amounts of the TCU software input, output variables differ, but the main functionality of a TCU is to switch gears. The way of switching gears is representative to one TCU model and version. So features like Up switching characteristic, that depends on the vehicle speed and the gas pedal (HPD pedal) state (and other not noted parameters) can be collected and measured (Fig 2.).

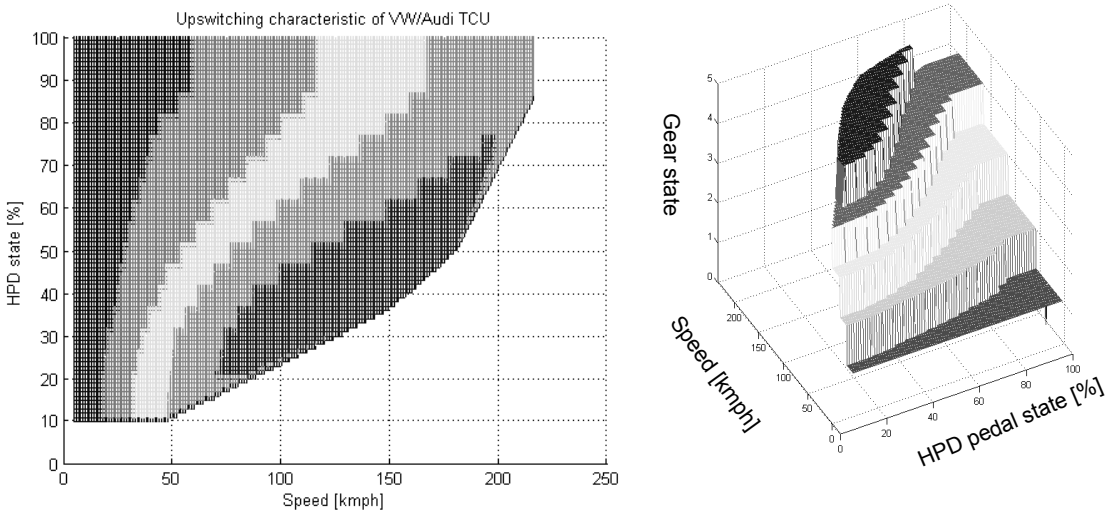



Figure 2: A typical Up switching characteristic

Errors in any sub module of the system will propagate to this feature level, so in this traditional way the main problem is solved. The remaining work is to select as many features as possible then choose appropriate methods for teaching and fault detections.

Unfortunately this high level feature-based approach cannot be applied in the case of TCU testing and many other embedded system software testing. The reason is, why these methods are not useable, that Robert Bosch GmbH does not own the whole development of the TCU software. The layers representing these higher layer functionalities belong to another company: ZF. Therefore most of the testing processes are done without having these higher layers.

## VI. Conclusions, problems to solve

The examination of the testing process of a TCU software, and the investigation of using traditional artificial intelligence based methods, as a new testing process has shown many problems. Most of these problems can be considered ordinary problems in the field of embedded software system testing.

The first and most important problem is getting information about the system. Measuring method shown above for the TCU model testing is a very typical and tool intensive one. These kinds of measurements always have their drawbacks. These drawbacks are delays, jitters, and the limitation of samples. Such measurement often modifies the measured system's performance.

The second problem is that embedded systems often has many versions, and therefore the selection of inputs and outputs for a Black Box test is very difficult.

The third problem is that many of the complex embedded systems are developed by more than one company, and therefore the self learning test cannot use the whole system as a Black Box, and perform traditional feature-based fault detection. To make this problem much harder, significant part of the embedded systems software are hardware drivers and there is nearly impossible to identify high level features in hardware handling.

There are other significant problems to solve in this area like finding representative faults for the teaching, determining the reliability of such a test system and so one. Taking into account the special features of the investigated system is also important. For example even the same TCU version could have different behavior based on its so called calibration, and unfortunately this calibration can be changed even on the fly.

As a conclusion, embedded system software testing by artificial intelligence methods is possible and can improve the quality of these products. Black Box methods like Limit checking and process identification are applicable, but the conditions are highly differs comparing to a traditional environment.

## References


[1] Brooks, F.P. Jr., "*The Mythical Man-Month: Essays on Software Engineering"*, Anniversary Edition, Addison-Wesley Pub. Co., 1995.

[2] The Bosch Yellow Jackets: *"Electronic Transmission Control ETC"* Bosch, 2004.

[3] Wikipedia: *"Code Coverage"*, http://en.wikipedia.org/wiki/Code_coverage

[4] Jorg Schauffele*: "Automotive Software Engineering: Principles, Processes, Methods, and Tools*", SAE International, 2005.

[5] R. Isermann.: *"Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance"* Springer 2006.

[6] S. Dick, A. Kandel: "*Computational Intelligence in Software Quality Assurance*". Series in Machine Perception Artificial Intelligence. Volume 63. Word Scientific Publishing Co. 2005.

[7] Robert Bosch GmbH: *"CAN Specification Version 2.0"* 1991.

[8] ISO 14230:1999. Road vehicles -- Diagnostic systems -- Keyword Protocol Part 1-4

[9] ISO 14229:2006. Road vehicles -- Unified diagnostic services (UDS)

[10] ETAS, INCA homepage: http://www.etas.com/en/products/inca_software_products.php

# NUMERICAL DISPERSION IN FINITE DIFFERENCE MEMBRANE MODELS

**Zsolt GARAMVÖLGYI**
**Advisors: Balázs BANK, László SUJBERT**

## I. Introduction

The finite difference method (FDM) is one of the most widespread approaches for numerical solution of partial differential equations (PDEs), mainly due to its general applicability and simplicity. As the computational power of modern processors makes real-time simulation possible, the FDM became a viable alternative for physics-based modeling of musical instruments in sound synthesis applications.

Mathematical models of musical instruments are based on various forms of the wave equation. It is known, that the analytical solution of these equations can be written as a superposition of a set of functions, determined by the boundary conditions, called *normal modes* ([1]). Normal modes are standing waves, whose frequencies are important characteristics of the model. Due to the properties of human hearing, in sound synthesis applications, the main goal is to construct a numerical model, whose modal frequencies approximate well those of the wave equation.

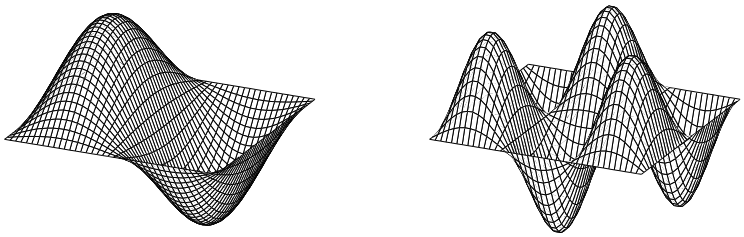


Fig. 1: The (2,1) and (3,2) normal modes of a square membrane.

It is known (see, e.g.,[2]) that a distinctive property of the FDM is the presence of *numerical dispersion*, which causes the waves of different frequencies in the model to propagate with different speed. Moreover, in the 2D and 3D case, the dispersion is direction-dependent. This phenomenon is related to the error of the modal frequencies of the finite difference model with respect to their theoretical values, however, it is rarely investigated from this point of view.

The vibrations of a membrane can be modeled by the 2D wave equation, for which various finite difference models with different dispersion properties can be constructed. In this paper, we will only deal with the simplest case, i.e., the square membrane discretized in Cartesian coordinates. An analytical expression will be shown for the modal frequencies of the model, which will form the basis of further investigations with respect to modal properties of various finite difference membrane models.

## II. Modal Frequencies of the Membrane Model

The boundary-value problem (BVP) of a square membrane with a side length $L$ can be written as

$$u_{tt} = c^2 \left(u_{xx} + u_{yy}\right), \qquad u(0,y,t) = u(L,y,t) = u(x,0,t) = u(x,L,t) = 0 \tag{1}$$

where $u = u(x,y,t)$ denotes the displacement of the membrane as a function of space-coordinates and time, $c$ is the propagation speed, and the lower indices denote second partial derivatives. For this BVP, the analytical solution can be written as a sum of normal modes, which are characterized by two indices

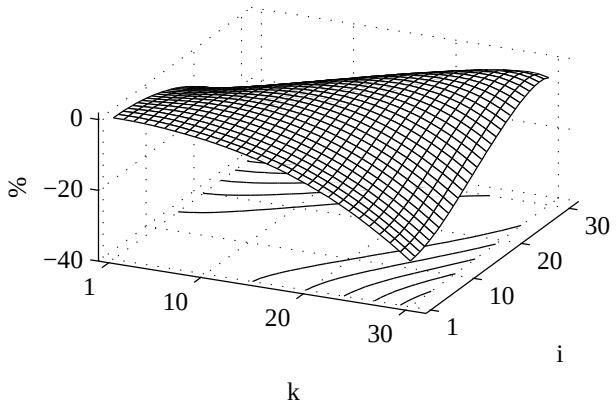



Fig. 2: Relative modal frequency error in the discretized model ($M = 31$, $c\,\frac{\Delta t}{\Delta x} = 0.703$).

($k$ and $i$). These indices correspond to the number of half-waves in $x$ and $y$ directions, respectively (see Fig. 1). Equation (1) can be discretized by approximating the derivatives by differences. In the simplest case, this yields an explicit difference equation of three variables. Basically, the model is a program that recursively solves this equation. However, this recursion formula can also be solved analytically, which provides information on how well the modal frequencies are approximated. The following formulae can be derived for the modal frequencies of the continuous-time model ($\omega$) and those of the finite difference scheme ($\tilde{\omega}$):

$$\omega_{k,i} = c\,(\pi/L)^2\,\sqrt{k^2+i^2}, \qquad \tilde{\omega}_{k,i} = \tfrac{1}{\Delta t}\,\arccos\left[\left(c\,\tfrac{\Delta t}{\Delta x}\right)^2\left(\cos\tfrac{k\pi}{M}+\cos\tfrac{i\pi}{M}-2\right)+1\right] \qquad (2)$$

The parameters $\Delta t$ and $\Delta x$ are the sampling periods with respect to time and space, and the model consists of $(M+1)\times(M+1)$ points. The relative error of $\tilde{\omega}$ for each mode of a $32\times 32$ membrane is shown in Fig. 2. It can be seen that the frequency error of the normal modes for which $k$ and $i$ are nearly equal is small. These modes correspond to waves travelling in diagonal direction. The rest of the modal frequencies are lower than expected, which means that the waves propagate slower in non-diagonal directions. The largest error can be observed for $i << k$ and $k << i$. These cases correspond to wave propagation in $x$ and $y$ directions (see, e.g., [3]).

## III. Conclusion

The effect of numerical dispersion in the model of the square membrane was presented. It was shown that due to this phenomenon, the modal frequencies of the model are generally lower than in the theoretical ideal case. A formula describing this effect was derived. This formula, as opposed to numerical simulation, is a more convenient way for error calculation.

Only the simplest case was studied but based on the above results, a similar analysis is planned to be performed for other types of membrane models in order to provide a detailed comparison with respect to modal properties. The further investigation is justified by the fact that, in spite of the extensive research in this field, no optimal solution for reducing the error caused by this undesirable effect has been found. Nor is it clarified, to what extent this phenomenon degrades the quality of the sound generated by the model. In order to answer this question, psychoacoustic considerations are necessary.

## References

[1] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*, Springer-Verlag, 1998.

[2] J. C. Strikwerda, *Finite Difference Schemes and Partial Differential Equations*, Wadsworth and Brooks, 1989.

[3] S. Bilbao and J. O. Smith, "Finite difference schemes and digital waveguide networks for the wave equation: Stability, passivity, and numerical dispersion," 11(3):255–266, May 2003.

# OBJECT RECOGNITION USING RADIUS PROFILES

**Sándor I. JUHÁSZ**
**Advisors: Gábor HORVÁTH (BUTE-MIT), Atsushi SUGAHARA (Toshiba-RDC)**

## I. Introduction

Several different approaches can be used for image classification and object detection. In the first case the goal is only to decide whether a certain object is present on the image, in the latter the position and orientation of the objects should be found too.

Our goal was to build a real-time object detector. Real-time applications don't allow the use of complicated algorithms due to processing limitations. The detector should be able to detect objects with shiny metallic surfaces too. For such objects it is difficult to define a certain texture. The visible texture changes rapidly as the object, the camera or the lights change position or orientation. The objects on the images can be in any position and orientation. They can be in partial occlusion by other objects and they can cover each other too. The algorithm should be able to handle complex, concave objects too. It should find the objects on the images, calculate the centers and orientations.

A common way to find objects is to use image patches. Many methods use such features [1, 2, 3]. These can be combined with other types of features, for example contour information [4]. Unfortunately these models are not fast enough to be used real-time.

Psychology experiments show that contours alone can be used effectively for object identification even in noisy and partially occluded cases [5]. This fact led to several methods using only the contour of the object for recognition purposes. Some of these can learn object category features semi-automatically using only the boundary rectangle of the object as additional information [6, 7]. The model closest to our algorithm is described in [8].

Contour fragments are good choices for features as they can be computed easily on the fly and they work well in cases where no texture is available too. They can represent inner structures of the object as well. The similarity of these fragments and the image contour at a certain position can be defined by the Chamfer-distance. The common way to speed up this calculation is to generate the distance transformed image. This can be still too slow for real-time applications if the distance transformation is used many times.

To avoid speed problems and to achieve rotation-invariance we used a different feature: local radius profiles (see section II.). Distance transformation is no longer needed with the new feature and global searching can be avoided too by fitting the features only to feature points where the contour curvature have extremum.

## II. Local Radius Profiles

Local radius profiles can be calculated at a high speed and only the contour of the object is needed for these calculations thus making them ideal features for this problem. Canny edge detector with some Gaussian smoothing is a good choice for robust contour detection. The short contour fragments should be deleted to speed up the search algorithm and to reduce noise in the results.

Features are extracted at curvature extrema points of the contour. These positions are scale- and rotation invariant. This property makes multi-scale matching easier. Feature points with curvatures above a high limit and under a low limit should be deleted as these represent very high curvature points which are sensitive to noise and very straight parts of lines with no stable maximum position.

*A Definition of Local Radius Profiles*

Figure 1 shows an example of how to calculate radius profiles. P is curvature extremum point of the contour. C is the center of the osculating circle at P with the radius *r*. It is used as the center for the calculations. We measure the distance between C and the analyzed contour at certain directions. A possible choice is to use a 60° interval on both sides of the CP axis. If the angle between the sampling directions is 2° then a 61-dimension vector will be the result. A bounding circle with a radius of *3r* can be used to limit the results. It makes the calculation of the radius possible at directions where the line starting from C cannot cross the contour. Radii are normalized so that CP distance equals 1. This choice makes the feature vectors scale-invariant. Rotational invariance is already guaranteed by the definition of the sampling directions.

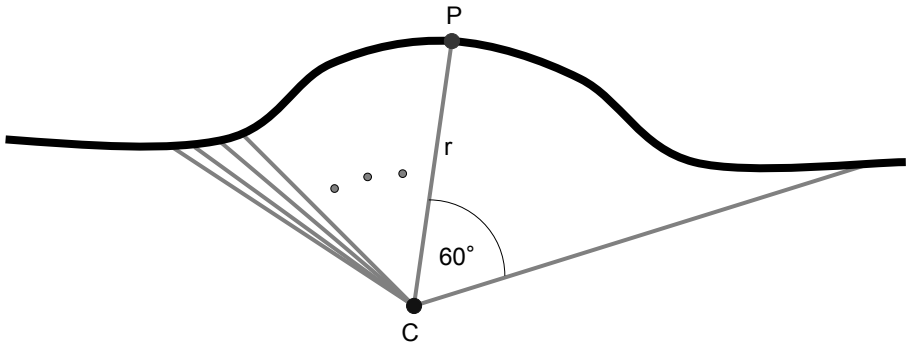



Figure 1: Structure of a radius profile

*B Distance of radius profiles*

With the above choice radius profiles are vectors of the 61 dimension space. We use the usual Euclidean distance to measure the distance between two such vectors $\gamma_1$ and $\gamma_2$:

$$D(\gamma_1, \gamma_2) = \sqrt{\sum_{i=1}^{61} (\gamma_{1,i} - \gamma_{2,i})^2} \,. \qquad (1)$$

## III. Training

*A Choosing Features*

The radius profiles or features used for recognition are acquired from the training images. The object boundaries should be clearly visible and no occlusion should be present. Boundary boxes, rotation values and the centers of the objects are also stored to help future identification of the orientation.

In the first phase all the features are acquired. We use agglomerative clustering to reduce the number of features. Radius profiles that are similar enough and point to the same object center are merged into a cluster. Each cluster is represented by only one feature. ANN method [9] is used to speed up distance calculations.

The validation images can contain other objects too and occlusion is allowed. There should be images with only background objects (negative validation images) and images with the object to be recognized too (positive validation images).

*B Building Weak Detectors*

Radius profiles alone are features not good enough to reliably detect objects. We combine them to create better detectors. We combine *k* of them to form a weak detector. *k=2* and *k=3* are typically used. Every possible combination is generated except the ones where the features are on the same curve and are very close to each other as these represent the same feature place. The classification output of a weak detector $h_i$ on image $I$ is defined as

$$h_i(I) = \max_{p_1, p_2 \dots p_k} (h_i(I_{p_1, p_2 \dots p_k})). \quad (2)$$

It is the maximum value of the classifications considering every possible $p_1, p_2 \dots p_k$ position. The possible locations are the curvature extrema of the contours here too. The classification at the position $p_1, p_2 \dots p_k$ is defined as

$$h_i(I_{p_1, p_2 \dots p_k}) = \begin{cases} 1 & if \quad D_{h_i}(I_{p_1, p_2 \dots p_k}) < th_{h_i} \\ 0 & otherwise \end{cases}. \quad (3)$$

$th_{h_i}$ is the learned threshold for weak detector $h_i$. If the result is 1 then the weak detector detected the object, otherwise it found nothing at that position. The distance inside the formula is a combination of the feature distances at the appropriate positions:

$$D_{h_i}(I_{p_1, p_2 \dots p_k}) = \frac{1}{m_s^2} \sum_{j=1}^{k} D(\gamma_j, \gamma_{I_{p_j}}) \quad (4)$$

where $\gamma_j$ is a feature (radius profile) of the weak detector and $\gamma_{I_{p_j}}$ is a radius profile acquired from the image $I$ at position $p_j$. The weight $m_s$ is used to measure the compactness of the features' central votes (see [8]). Figure 2 shows a *k=3* example of a weak detector's object center guess. The big circle's center is the real object center. The small circle centers are the weak detector's features' object center guesses. $m_s = k$ and it is decreased by 0.5 each time the votes of two features in a weak detector for the center of the object have a distance more than *2r*. In the example $m_s$ equals 2.

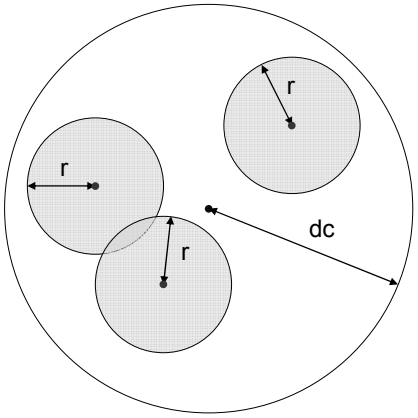



Figure 2: Feature votes for the object center in a weak detector

We have to evaluate the validation images to estimate the optimal $th_{h_i}$ values. On the positive validation image set $D_{h_i}(I_{p_1, p_2 \dots p_k})$ is considered infinity if the respective object center estimations are not close enough to the real object center. *dc* is the maximum distance allowed.

*C Building a Strong Detector*

After defining the weak detectors $h_i$ we should combine them to form a strong detector. The output of the strong detector *H* is the combination of the of the weak detectors' outputs:

$$H(I) = sign(\sum_{i=1}^{T} h_i(I) \cdot w_{h_i} - th). \quad (5)$$

T is the number of weak detectors, *th* is the threshold of detection. Adaboost [10] is used to calculate the $w_{h_i}$ weights.

## IV. Object Detection

Object detection tries to fit the weak detectors to the feature positions in every possible combination. A weak detector will fire if all of the following conditions are satisfied:

- equation (3) is 1
- the object center guesses are within a *dc* radius circle of their center of gravity
- the center guesses fit in their yaw rotation values too.

A Hough-style voting is used to calculate the object positions. Multiple planes are defined, each of them represents a certain yaw rotation of the object, and another series of planes are responsible for the different pitch rotations. A 2-D Gaussian is added to the adequate plane at the object center estimation weighted by the *w* weights each time a weak detector detects an object. Neighboring planes get votes too with smaller weights to compensate rounding errors.

Mean-shift algorithm is used to calculate local maxima of the sum of planes. These will be the object centers if they are above the *th* threshold (see equation 5). The rotational position of the object can be calculated by searching the plane that gives the highest value at the object center's position.

## V. Conclusion

With the help of local radius profiles we were able to build a quick real-time object recognition system that can be used for any object, even for objects without any surface texture. Despite the low complexity of the algorithm it is able to detect complex objects with any 3D orientation and can operate successfully on images where the objects are partially occluded too.


## Acknowledgement

I would like to acknowledge my advisor Atsushi Sugahara for his helpful ideas and endless patience during the development of the object recognition system. I also would like to thank Toshiba Corporation for making this research possible.


## References


[1] R. Fergus, P. Perona, and A. Zisserman, "Object Class Recognition by Unsupervised Scale-invariant Learning," in *CVPR*, pp. 264-271, 2003.

[2] A. Torralba, K. P. Murhy, and W. T. Freeman, "Sharing features: efficient boosting procedures for multiclass object detection," in *CVPR*, pp. 762-769, 2004.

[3] B. Leibe, A. Leonardis, and B. Schiele, "Combined object categorization and segmentation with an implicit shape model," in *ECCV workshop on statistical learning in computer vision*, pp. 17-32, 2004.

[4] A. Opelt, A. Pinz, and A. Zisserman, "Fusing shape and appearance information for object category detection," in *Proceedings of the British Machine Vision Conference*, 2006.

[5] J. De Winter and J. Wagemans, "Contour-based object identification and segmentation: Stimuli, norms and data, and software tools," *Behavior Research Methods, Instruments, & Computers*, 36 (4): 604-624, 2004.

[6] J. Shotton, A. Blake, and R. Cipolla, "Multi-Scale Categorical Object Recognition Using Contour Fragments," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 30 (7): 1270-1281, 2008.

[7] V. Ferrari, F. Jurie, and C. Schmid, "From images to shape models for object detection," *Inria Technical Report*, *RR-6600*, 2008.

[8] A. Opelt, A. Pinz, and A. Zisserman, "A Boundary-Fragment-Model for Object Detection," in *ECCV*, pp. 575-588, 2006.

[9] S. Arya, T. Malamatos, and D. M. Mount, "Space-Time Tradeoffs for Approximate Spherical Range Counting," *16th Ann. ACM-SIAM Symposium on Discrete Algorithms*, pp. 535-544, 2005.

[10] Y. Freund, and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Computer and System Sciences*, 55 (1): 119-139, 1997.

# 60 GHz BAND: OFDM OR SINGLE CARRIER TRANSMISSION

**Zsolt KOLLÁR**
**Advisor: Gábor PÉCELI and Reiner THOMÄ (TU-Ilmenau)**

## I. Introduction

As the need for wireless communication networks is increasing, the interest in higher unlicensed frequency bands is emerging – especially for the frequency band at 60 GHz. At these high carrier frequencies the non-linearity of the High Power Amplifier (HPA) becomes a real problem. The engineers have to decide between two commonly used wireless communication systems: the Orthogonal Frequency Division Multiplexing (OFDM) [1] and the Single Carrier with Cyclic Prefix (SC-CP) [2]. The OFDM is a well studied modulation technique but it has a relatively high sensitivity to Carrier Frequency Offset (CFO) and phase noise and also to nonlinear effects of the HPA. On the other hand the SC-CP systems are less studied in the literature, but the transmission signal is less sensitive to CFO and nonlinearity. In this paper we will briefly introduce the two systems and compare them.

## II. The OFDM system

The Orthogonal Frequency Division Multiplexing (OFDM) is one of the most popular multicarrier modulation techniques (used in ADSL, WLAN, DVB-T standards), where the digital data are modulated on a large number of orthogonal subcarriers and the modulation and demodulation of the subcarriers can be easily performed using Fast Fourier Transform (FFT) and Inverse FFT.

A general structure of an OFDM system is shown in Figure 1. At first, the binary information pro-

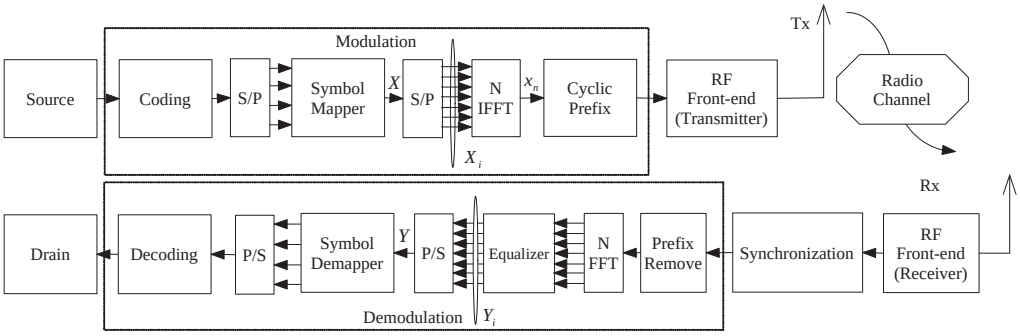



Figure 1: Block diagram of an OFDM system

vided by the source is coded. Channel coding is used to introduce redundancy into the bit stream to reduce the error probability due to the influence of the radio channel. The coded bit stream is partitioned into M blocks by a serial/parallel (S/P) converter, which are then transformed to a complex symbol $X$ by the symbol mapper. In multicarrier modulation the available bandwidth $W_B$ is divided between among N subcarriers. The complex data stream $X$ is partitioned into blocks of $N$ data symbols $X_i$, that are transmitted in parallel by modulating the $N$ subcarriers (in practical systems not all subcarriers are used). In OFDM systems, due to the orthogonality, the IFFT/FFT algorithms can be used as modulators and demodulators, i.e. orthogonality means that there is no cross-talk between among the sub-channels as long as no distortion is present. The time-domain representation of one discrete OFDM symbol can be expressed with the aid of the N-point IFFT of the set of N discrete

complex symbols $X_i$. To overcome the effect of multipath propagation in the channel a cyclic prefix is usually added to the time domain OFDM symbol. It is a repeated part of the end of the OFDM symbol as an extension at the beginning of the symbol to maintain time domain continuity of the OFDM symbol. The RF front-end is responsible for upconverting the complex base-band signal to the radio frequency. After the transmission through the physical medium (e.g. air, cable,...), the receiver RF front-end downconverts the received signal again to the baseband. Synchronization at the receiver is needed to eliminate the possible frequency and timing offsets. After removing the CP, the N-point FFT is used for each received OFDM symbol to demodulate the subcarriers. The equalizer tries to compensate the channel influence for each received OFDM subcarrier based on a channel estimation. The channel estimation is calculated based on OFDM pilot symbols and the pilot subcarriers in the data symbols. The pilot symbols, which were sent before the data symbols, are also known at the receiver. After demodulation and equalization the complex symbols $Y$ are demapped to a serial bitstream which can be decoded and passed on to the drain.

The OFDM signal suffers from a well know problem, the high Peak-to-Average Power Ratio (PAPR), which adversely affects the HPA and D/A-A/D converters. If the HPA has a smaller linear range than required, nonlinear effects decrease the system performance. Many methods have already been proposed to reduce the PAPR of OFDM signals. Each method has its own advantages and disadvantages, the engineers should choose the best method according to the system parameters and requirements.

## III. The Single Carrier system

From a signal processing point of the SC-CP system has the same blocks as the OFDM system, the only difference is that the IFFT block is moved from the transmitter to the receiver. The received symbols are converted to frequency domain, then equalized, and at the end they are turned back to time domain before the decisions are made. The order of the other blocks remain unchanged. The energy of one data symbol is spread throughout the entire available frequency band. This effect can lead to the fact that the SC-CP system can deal better with frequency selective channels. This is why the SC-CP system outperforms the OFDM system for uncoded case. A spectral disadvantage of SC-CP is that it is not as compact as the OFDM signal, which can be important in case of a disturbance occurs in the neighboring frequency bands.

The PAPR of an OFDM signal is higher than for SC signals. So if cheap HPA-s are available which do not have high linearity, SC-CP systems are preferred. Another advantage of SC-CP over OFDM is that the packet size can be changed more dynamically and the synchronization problems caused by CFO are not so crucial. On the other hand, until the last few years, less attention was paid to the SC systems so it is less studied in the literature and the synchronization methods are not so developed.

## IV. Conclusion

In this paper, we have shown that the OFDM and the SC-CP systems are built up of similar signal processing blocks. The following parameters and design concepts have to be taken into account:

- PAPR – linearity of the HPA.
- Synchronization – especially for SC systems.
- Coding parameters and channel characteristics.
- Spectral compactness – design of front-end filters.

## References


[1] R. van Nee and R. Prasad, Eds., *OFDM for Wireless Multimedia Communication*, Artech House, 2000.

[2] P. F. Vitetta, G.Kalbasi, R.Al-Dhahir, N.Uysal, and H. M.Mheidat, "Single-carrier frequency domain equalization," *IEEE Signal Processing Magazine*, 25(5):37–56, Sept. 2008.

# ANALYSIS OF THE SIGN-ERROR FXLMS ALGORITHM

**György OROSZ**
**Advisors: Gábor PÉCELI and László SUJBERT**

## I. Introduction

Sign-error algorithms are mainly used due to their simple realization and low computational demand [1, 2]. These algorithms have extensive literature, however, the sign-error adaptive controller algorithm hasn't been investigated yet. This paper introduces the analysis of the Sign-Error Filtered-x Least-Mean Square (SE-FxLMS) adaptive controller algorithm. The advantage of the sign-error adaptive controller, above the reduced computational complexity, is that it makes possible *data compression* in the control loop *in a very simple way*. The signal compressing feature of sign-error algorithms is demonstrated in [4] that introduces a wireless active noise control system where a sign-error resonator based adaptive controller is used for noise control. In spite of the sign-error algorithm introduced in [4], the SE-FxLMS can also be applied in the case of general stochastic reference and control signals, and it preserves the capability of signal compression.

The paper is structured as follows. Section II provides a brief description of the SE-FxLMS algorithm. In Section III, it is shown that the mean-absolute error of the algorithm is bounded for any value of the convergence parameter. The characteristic properties of the sign-error FxLMS algorithm are demonstrated with simulation in Section IV.

## II. Introduction to the SE-FxLMS Algorithm

FxLMS [3] is one of the most well-known adaptive controller algorithms. It has simple structure and ensures high degree of freedom due to the large number of free parameters so it is capable of the controlling of systems with complicated transfer function (e.g. acoustic and complex mechanical systems) [5].

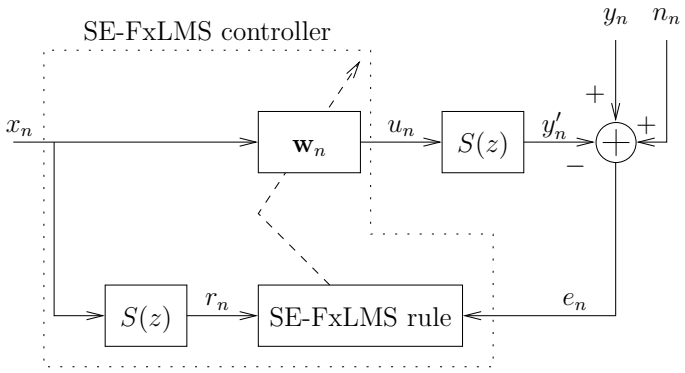



Figure 1: Block diagram of the SE-FxLMS algorithm

The block diagram of the sign-error variant of the FxLMS algorithm can be seen in Fig. 1. In the figure, $S(z)$ denotes the plant to be controlled. The input of $S(z)$ i.e. the output of the controller is $u_n$ and the output of $S(z)$ is denoted by $y'_n$. $y_n$ and $e_n$ denote the desired value of $y'_n$ and the error signal respectively. $n_n$ denotes the noise. $x_n$ is the so-called reference signal that is correlated with $y_n$.

A typical application of the FxLMS algorithm is the active noise control (ANC) [5]. In the ANC systems, $y_n$ is the noise to be suppressed, $S(z)$ is an acoustic system and the control signal $u_n$ is the so-called anti-noise that is radiated by a loudspeaker—that is the actuator. The superposition of the noise and the anti-noise is the residual noise i.e. the error signal $e_n$ that is sensed by a microphone. $u_n$ should be inverted in order to achieve subtraction at the microphone as shown in Fig. 1. In a practical application, it can easily be ensured that $x_n$ is correlated with $y_n$.

The control signal $u_n$ is produced by filtering $x_n$ with the adaptive transversal filter $\mathbf{w}_n$:

$$u_n = \sum_{i=0}^{N-1} w_{i,n} x_{n-i} = \mathbf{x}_n^{\mathrm{T}} \mathbf{w}_n, \tag{1}$$

where $\mathbf{w}_n = [w_{0,n} \dots w_{N-1,n}]^{\mathrm{T}}$ and $\mathbf{x}_n = [x_n \dots x_{n-N+1}]^{\mathrm{T}}$. In the case of the sign-error algorithms, the weights of $\mathbf{w}_n$ are updated by the steepest descent method so that the absolute value of the error i.e. $|e_n|$ is minimized [2]. The SE-FxLMS algorithm can be derived from the "conventional" FxLMS by replacing the error with its sign in the updating term of $\mathbf{w}_n$. Since the updating rule of the FxLMS is $\mathbf{w}_{n+1} = \mathbf{w}_n + \mu e_n \mathbf{r}_n$ [3] so the SE-FxLMS algorithm is as follows:

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \operatorname{sign}(e_n)\mathbf{r}_n, \tag{2}$$

where $\operatorname{sign}(\cdot)$ denotes the sign function, $\mu$ is the positive convergence parameter and $\mathbf{r}_n = [r_n \dots r_{n-N+1}]^{\mathrm{T}}$ is the vector of the filtered reference signal:

$$r_n = \sum_{k=0}^{F-1} s_k x_{n-k} \leftrightarrow \mathbf{r}_n = \sum_{k=0}^{F-1} s_k \mathbf{x}_{n-k}, \tag{3}$$

where $\{s_k, k = 0 \dots F-1\}$ is the impulse response of $S(z)$.

Note that (2) uses only the sign of the error signal, which makes the computation simple since the multiplication by a $\operatorname{sign}$ function means only the manipulation of the sign of the multiplicand. Another advantage of the algorithm emerges when in the control system, the central controller—that implements the SE-FxLMS algorithm—and the sensor are separated and they are connected over a low bandwidth communication channel. If the sensor transmits only the sign of the error signal, significant reduction in the amount of data can be achieved [4]. The error signal can be measured directly by the sensor—e.g. in the ANC systems the residual noise—or it can be calculated if the desired signal $y_n$ is known by the sensor that measures $y'_n$.

## III. Upper Bound of the Mean-Absolute Error

As (2) shows, the parameter $\mathbf{w}_n$ is updated irrespectively of the magnitude of the error signal even in the tight region of the optimum where the error is small. This causes the fluctuation of the parameters around the optimum so the error signal can't be set to zero even in optimal case. This residual error is one of the most commonly investigated property of sign-error algorithms [2] and it is generally characterized by the mean-absolute error (MAE) that will be derived for the SE-FxLMS in this section.

According to (2), one can obtain a general updating rule for $\mathbf{w}_n$:

$$\mathbf{w}_n = \mathbf{w}_{n-k} + \sum_{q=1}^{k} \mu \operatorname{sign}(e_{n-q})\mathbf{r}_{n-q}, \quad k \geq 1. \tag{4}$$

In order to obtain a formula for $e_n$, first, $y'_n$ should be calculated:

$$y'_n = \sum_{k=0}^{F-1} s_k u_{n-k} = \sum_{k=0}^{F-1} s_k \mathbf{x}_{n-k}^{\mathrm{T}} \mathbf{w}_{n-k} = s_0 \mathbf{x}_n^{\mathrm{T}} \mathbf{w}_n + \sum_{k=1}^{F-1} s_k \mathbf{x}_{n-k}^{\mathrm{T}} \left[ \mathbf{w}_n - \sum_{q=1}^{k} \mu \operatorname{sign}(e_{n-q})\mathbf{r}_{n-q} \right], \tag{5}$$

where (1) and (4) are used. Due to (3), (5) can be rewritten as follows:

$$y'_n = \mathbf{r}_n^{\mathrm{T}} \mathbf{w}_n - \sum_{k=1}^{F-1} s_k \mathbf{x}_{n-k}^{\mathrm{T}} \sum_{q=1}^{k} \mu \operatorname{sign}(e_{n-q})\mathbf{r}_{n-q} = \mathbf{r}_n^{\mathrm{T}} \mathbf{w}_n - h_n, \tag{6}$$

$$h_n = \mu \sum_{k=1}^{F-1} \sum_{q=1}^{k} \operatorname{sign}(e_{n-q}) s_k \mathbf{x}_{n-k}^{\mathrm{T}} \mathbf{r}_{n-q}. \tag{7}$$

Let $\mathbf{w}_{opt}$ denote the optimal value of $\mathbf{w}_n$—that makes the absolute error minimal—so the error of the parameter vector is $\widetilde{\mathbf{w}}_n = \mathbf{w}_n - \mathbf{w}_{opt}$. It is assumed that $y_n$ is of the form:

$$y_n = \mathbf{r}_n^{\mathrm{T}} \mathbf{w}_{opt} + \nu_n, \tag{8}$$

where $\nu_n$ is the component of $y_n$ that cannot be tracked even in optimal case. In the following, $\nu_n$ will be included into the noise and a resultant noise $\varepsilon_n = n_n + \nu_n$ will be used. According to the definition of $y_n$, the error signal can be written as follows:

$$e_n = y_n - y'_n + n_n = \mathbf{r}_n^{\mathrm{T}} \mathbf{w}_{opt} - \mathbf{r}_n^{\mathrm{T}} \mathbf{w}_n + h_n + \varepsilon_n = -\mathbf{r}_n^{\mathrm{T}} \widetilde{\mathbf{w}}_n + h_n + \varepsilon_n. \tag{9}$$

Since (2) and (9) are formally similar to the equations that describe simple sign-error LMS (SE-LMS) algorithm which was investigated in [1] hence, those results can be applied for the SE-FxLMS algorithm as well—SE-LMS is a special case of SE-FxLMS with $S(z) = 1$. The following equations describe the SE-LMS algorithm and they were used for the derivation of the MAE of the SE-LMS [1]:

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \operatorname{sign}(e_n) \mathbf{x}_n \quad \text{and} \quad e_n = -\mathbf{x}_n^{\mathrm{T}} \widetilde{\mathbf{w}}_n + \varepsilon_n, \tag{10}$$

and the MAE—that is denoted by $E^a$—of the SE-LMS is:

$$E^a = \sum_{k=1}^{n} E\left\{|e_k|\right\} \leq \frac{\widetilde{\mathbf{w}}_1}{2\mu n} + \frac{1}{2}\mu N R_{xx}(0) + E\{|\varepsilon_n|\}, \tag{11}$$

where $E\{|\varepsilon_n|\}$ denotes the expected value of the absolute value of the noise and $R_{xx}(0) = E\left\{x_n^2\right\}$ is the autocorrelation function of $x_n$ in the origin, i.e. the variance of $x_n$ if $E\left\{x_n\right\} = 0$.

Comparing (10) with (2) and (9), it can be noted that in the SE-FxLMS, $\mathbf{r}_n$ is used instead of $\mathbf{x}_n$ so in the calculation of the MAE of the SE-FxLMS, $R_{rr}(0)$ should be used instead of $R_{xx}(0)$.

On the other hand, the effect of the dynamic system appears in the error as an additive term—see $h_n$ in (9)—hence, it can be handled as the part of the noise. Since the noise increases the MAE (11) by its mean-absolute value i.e. by $E\left\{|\varepsilon_n|\right\}$ thus the dynamic system increases the MAE by $E\left\{|h_n|\right\}$ i.e. by the expected value of $|h_n|$.

An upper bound of $E\left\{|h_n|\right\}$ can be calculated according to (7) using that $|a+b| \leq |a| + |b|$ and $E\left\{|\mathbf{x}_{n-k}^{\mathrm{T}} \mathbf{r}_{n-q}|\right\} \leq E\left\{\sum_{i=0}^{N-1} |x_{n-k-i}||r_{n-q-i}|\right\} = N R_{|x||r|}(k-q)$ where $R_{|x||r|}(k-q)$ is the cross-correlation of $|x_n|$ and $|r_n|$ if it is assumed that they are stationer. Finally, one obtains:

$$E\left\{|h_n|\right\} \leq \mu \sum_{k=1}^{F-1} \sum_{q=1}^{k} |s_k| N R_{|x||r|}(k-q) = \mu\eta. \tag{12}$$

Using (11), (12) and the above discussion, the MAE of the SE-FxLMS can be calculated:

$$E^a \leq \frac{\widetilde{\mathbf{w}}_1}{2\mu n} + \frac{1}{2}\mu N R_{rr}(0) + E\left\{|h_n|\right\} + E\left\{|\varepsilon_n|\right\} \leq \frac{\widetilde{\mathbf{w}}_1}{2\mu n} + \frac{1}{2}\mu N R_{rr}(0) + \mu\eta + E\left\{|\varepsilon_n|\right\}. \tag{13}$$

(13) can be partitioned into three groups:

- $\frac{1}{2\mu n}\widetilde{\mathbf{w}}_1$ is the effect of the transient and it vanishes if $n \to \infty$. Since it is inversely proportional to $\mu$, the smaller the $\mu$ is, the longer the transient phase is.
- $\mu\left[\frac{1}{2} N R_{rr}(0) + \eta\right]$ is the effect of the constant step size, i.e. the parameters are modified irrespectively of the magnitude of the error. This term is proportional to $\mu$ so the MAE can be decreased by decreasing $\mu$.
- $E\left\{|\varepsilon_n|\right\}$ is the effect of the noise, it isn't influenced by the convergence parameter.

As (13) shows, the SE-FxLMS preserves the characteristic property of the sign-error algorithms that in ideal case, the steady state error can be set to an arbitrary small value, however, at the expense of the settling time [1, 2].

## IV. Simulation results

The properties of the sign-error FxLMS algorithm were also investigated with simulation and it was compared with the original FxLMS algorithm as well. The plant in the simulation was a second order system that has relatively high dynamics:

$$S(z) = 2\frac{z^2 + 0.6627z + 0.6214}{z^2 - 0.3373z + 0.81} \tag{14}$$

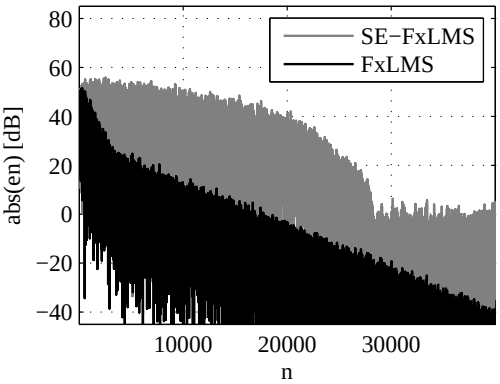



Figure 2: Transients of the SE-FxLMS and the FxLMS algorithms

In Fig. 2, the transients of the SE-FxLMS (gray line) and FxLMS (black line) algorithms can be seen. The parameters of the algorithms were: $\mu = 10^{-3}$ for SE-FxLMS and $\mu = 10^{-4}$ for FxLMS, $N = 10$ and the reference signal $x_n$ was Gaussian random process with the variance of $R_{xx}(0) = 1 \rightarrow R_{rr}(0) = 15.6$. Comparing the transients of the algorithms, one can observe the most characteristic property of the SE-FxLMS algorithm: in spite of the FxLMS that ensures exponentially decreasing error, in the case of the SE-FxLMS, the average absolute error doesn't decrease below a certain level that is determined by the convergence parameter.

The degradation of the residual error in the case of the SE-FxLMS is the result of the fact that the magnitude of the error is neglected during the adaptation. However, this simplification of the algorithm makes also possible the implementation of the algorithm in systems with limited resources [4].

The steady state MAE ($\mathrm{MAE}_{\mathrm{ss}}$) of the sign-error algorithms is generally an important design parameter that can be estimated according to (13) when $n \rightarrow \infty$. In the figure, the SE-FxLMS is in steady state for $n > 28000$. In the simulation, the $\mathrm{MAE}_{\mathrm{ss}}$ was $0.303$ and the estimated $\mathrm{MAE}_{\mathrm{ss}}$ was $3.44$, which shows the validity of (13). If the $\mathrm{MAE}_{\mathrm{ss}}$ were calculated according to (11), i.e. neither the filtering of the reference signal nor the effect of the plant $S(z)$ were taken into account, the $\mathrm{MAE}_{\mathrm{ss}}$ would be $0.005$ so the dynamic system in the feedback loop of the sign-error algorithm has significant effect.

## V. Conclusions

In this paper, the SE-FxLMS algorithm was introduced and its analysis was presented. It was shown that the bound of the MAE of the SE-FxLMS algorithm can be given for any value of the convergence parameter. Both the analytical and simulation results show that the dynamic plant in the feedback loop of the sign-error algorithm can significantly influence the MAE.

In the future, the extension of the results on multiple-input multiple-output case can be expected.

## References


[1] A. Gersho, "Adaptive Filtering with Binary Reinforcement," *IEEE Transactions on Information Theory,* Vol. IT-30, No. 2, Mar. 1984., pp. 191–199.

[2] O. Macchi, "Advances in Adaptive Filtering," in *Digital Communications,* Amsterdam, The Netherlands: North-Holland, 1986., pp. 41–57.

[3] B. Widrow, *Adaptive inverse control*, Prentice-Hall, Inc. 1996, pp. 160–208.

[4] Gy. Orosz, L. Sujbert, G. Péceli, "Spectral Observer with Reduced Information Demand," *Proc. of the IEEE International Instrumentation and Measurement Technology Conference*, Victoria, Canada, May 12-15. 2008., pp. 2155–2160.

[5] M.O. Tokhi, S.M. Veres, *Active Sound and Vibration Control: theory and applications*, Bath, UK, The Institution of Electrical Engineers, 2002.

# IMPLEMENTING A GLOBAL OPTIMIZATION ALGORITHM RELATED TO BIOINFORMATICS WITH A HIGH-PERFORMANCE FPGA

**Imre PECHAN**
**Advisor: Béla FEHÉR**

## I. Introduction

Bioinformatics includes several computationally demanding problems that are based on modeling and simulation of biochemical processes. One of these is the molecular docking, whose aim is to simulate the interaction of two given protein molecules and to predict if they can bind to each other.

The common drawback of the existing docking algorithms and softwares is their low speed. That is why the possibility of an FPGA-based implementation of molecular docking is worth investigating. The purpose of this short paper is to introduce and evaluate such an implementation, which mimics a popular and well-known docking software called AutoDock 4, as well as to suggest possible future improvements of the implemented algorithm.

## II. The Docking Algorithm

The two molecules that are involved in the docking problem are called the receptor and the ligand. During the docking algorithm the receptor is fixed in space while the ligand can be moved and rotated, and the aim is to find the energetically most favorable position of the ligand relative to the receptor. A scoring function is used to estimate the free energy of the different arrangements and to evaluate them. If the free energy is sufficiently negative for the ideal position of the molecules, they might be able to bind to each other in reality as well.

Molecular docking can be considered as a global optimization problem, where the ideal arrangement of the molecules, that is, the global minimum of the scoring function has to be found. The parameters of the problem are the three translational and the three rotational degrees of freedom describing the ligand's position and orientation. Besides, the ligand is often treated flexible, which means that it includes rotatable bonds whose torsional angle can change. Each of these bonds represents an additional parameter, thus the degrees of freedom of the problem can easily be 15-20 depending on the size of the ligand. Optimization methods used for solving the docking problem include simulated annealing, swarm-based optimization algorithms and genetic algorithms.

### *A. The Implementation*

The implementation has a pipelined structure where different stages are responsible for implementing the optimization algorithm, calculating the coordinates of the ligand in the current position and for evaluating the scoring function. The implementation uses the AutoDock 4 scoring function and applies a simple genetic algorithm, which is quite different from the one used in AutoDock 4. These differences are that the implemented algorithm is steady state [1] and not generational, it uses a totally random parent selection instead of a proportional one, and it does not include a genetic operator for local search. These simplifications seem to be reasonable considering that the algorithm is implemented on FPGA [1], however, their influence on the efficiency of the algorithm is to be investigated.

### *B. Evaluation of the Implementation*

When evaluating the implementation, two aspects have to be taken into account: the speed and the rate of successful dockings. A docking is considered successful if the position of the docked ligand is

sufficiently close to the expected one, that is, the root mean square deviation of the two structures is below 2 Angström.

For two medium size ligands, the XK263 and the sialic acid, a flexible docking consisting of 500.000 energy evaluations lasts 5,72s and 2,26s in the FPGA, respectively. Running AutoDock 4 on a standard PC with a 2.4 GHz Intel Core2 Duo CPU, the same task requires 45,65s and 32,16s, thus the FPGA implementation has a speedup factor of ×8 and ×14 over the CPU for these molecules. Further significant speedup could be achieved with improving the implementation.

If XK263 and sialic acid are treated as rigid, the rate of successful dockings is 77% and 47%, respectively. Since the molecules have 10 and 11 rotatable bonds, treating them as flexible makes the problem significantly harder. In this case the rate of successful dockings decreases to 3% and 16%, which suggests that the implemented steady state genetic algorithm cannot deal with too many degrees of freedom. This is confirmed by the fact that AutoDock 4 can find the proper docking position for both the flexible XK263 and the flexible sialic acid in about 30% of the cases.

## III. Possible Improvement of the Genetic Algorithm

AutoDock 4 uses a hybrid search method, which includes an ordinary genetic algorithm extended by a genetic operator for local search, that is, with a certain probability the generated offspring are subjected to local search after crossover and mutation. This method proved to be quite successful, improving the accuracy of docking in the case of AutoDock 4 [2].

The applied local search is a hill climbing method, where in each iteration step the current value of the parameters is altered by adding a normally distributed random variable v with mean b and deviation ρ to it. If the new solution is better, the movement is considered to be a success, if not, a failure. Value of b for the next step is the weighted sum of the current b and the current v, and ρ is altered adaptively after a certain number of consecutive successes or failures.

The whole docking algorithm was implemented in C as well, along with the local search method described above. When using the steady state genetic algorithm extended with the local search, the rate of successful dockings increased from 3% and 16% to 21% and 41%, respectively, for the flexible XK263 and sialic acid. These values are quite close to the 30% success rate of AutoDock 4. The rate of successful dockings in different cases can be seen in Table 1. Since the local search is quite simple and clearly improves the efficiency of the algorithm, adopting it in the FPGA seems to be a reasonable decision.

Table 1: Efficiency of the steady state genetic algorithm

| **Molecule** | | **Degrees of freedom** | **Rate of successful dockings** | |
|---|---|---|---|---|
| | | | Without local search | With local search |
| **XK263** | Rigid | 6 | 77% | 97% |
| | Flexible | 16 | 3% | 21% |
| **Sialic acid** | Rigid | 6 | 47% | 82% |
| | Flexible | 17 | 16% | 41% |


## Acknowledgement

I would like to give thanks to Dr. Béla Fehér for his persistent and useful advices and help.


## References


[1] B. Shackleford and G. Snider, "A High-Performance, Pipelined, FPGA-Based Genetic Algorithm Machine," *Genetic Programming and Evolvable Machines*, 2(1):33–60, March. 2001.

[2] C. D. Rosin and R. S. Halliday, "A Comparison of Global and Local Search Methods in Drug Docking," in *Proc. of the Seventh International Conference on Genetic Algorithms*, Michigan, USA, July 19–23 1997.