Project: Verification of Complex Hierarchical Systems (VOCHS)

Principal Investigator: dr.ir. T.A.C. Willemse

Funding Organisation: NWO (2010–2014)

Summary of the Research Project

The Compact Muon Solenoid (CMS) research group at CERN is responsible for one of the four experiments in the Large Hadron Collider in Geneva. The software, controlling this experiment, consists of 20,000 finite state machines, which are strictly hierarchically organised. The complexity of the software has grown out of hand and it is not predictable anymore. This is becoming a nuisance. Consequently, the CMS research group currently invests in the verification thereof by translating the finite state machines to mCRL2, which is a behavioural specification and analysis methodolgy which we have developed.

Verification of small constellations of state machines is quite straightforward. However, the verification of the entire system consisting of 20,000 parallel components is of a different category, which cannot be done by CERN alone, but requires our skill and experience. We are convinced that the hierarchical architecture enables us to prove the correctness of this massive control system. As the Large Hadron Collider will become fully operational in the coming years, an urge is felt to get the software right.

In tackling this problem, we pursue two main approaches concurrently. The first approach is to employ (symbolic) PBES technology that we have developed for the verification of modal properties. Our second approach is to employ the sheer power of parallel processing using (distributed clusters of) multi-core cpus for solving BESs resulting from modularised PBESs.

This project is geared towards the verification challenge at CERN; its underlying purpose is to increase the verification capabilities as a whole.

Keywords

Hierarchical control systems, CERN, software correctness, parameterised Boolean equation systems, multi-core verification.

Composition of the Research Team

The research team consists of the following members in alphabetical order:

Name	Specialises in	Institute
Prof.dr. R. Cleaveland	Verification methods, Real-time	University of Maryland
Dr. F. Glege	Domain expert, stakeholder CMS	CERN
Prof.dr.ir. J.F. Groote	Verification methods, Real-time	TU/e
Prof.dr. J.C. van de Pol	Distributed verification methods	UT
Dr.ir. T.A.C. Willemse	Verification methods	TU/e
Ph.D. student 1	Verification methods	TU/e
Ph.D student 2	Distributed computing	UT
Scientific Programmer	Programming skills (C++)	TU/e

Prof.dr.ir. J.F. Groote and Prof.dr. J.C. van de Pol will act as advisors ("promotoren" in Dutch) for the Ph.D. students.

1 Description of the Research Project

1.1 Scientific Quality

Background

The CMS research group at CERN[1] is responsible for the *Compact Muon Solenoid (CMS)* experiment. This is one of the four groups at CERN responsible for particle detection experiments conducted using the *Large Hadron Collider (LHC)*, which will become operational in mid November 2009. The CMS detector has a diameter of 15 meters and weighs approximately 12,000 tonnes. The experiment is monitored and controlled in real-time by a complex hierarchical control system consisting of approximately 20,000 finite state machines that is primarily responsible for switching the experiment on and off.

The CMS research group observed that the complexity of this control system has grown out of hand. The system repeatedly lost track of parts of the experiment due to 'inadvertent' oversights in the software of these finite state machines. Realising that the problems that the CMS research group are facing could not be solved solely by increasing the effort in (re)programming the finite state machines, but are fundamental to the complexity of the entire system, we have been approached to analyse the control system.

A first experiment showed that the finite state machines could straightforwardly be modelled in mCRL2, see [32]. Moreover, relying on process algebraic reasoning and model checking, we were able to thoroughly analyse the behaviours of combinations of a small number of non-trivial finite state machines. Based on our findings, the CMS research group has decided to invest in a systematic translation of the finite state machines to mCRL2. To this end, it has assigned one (and soon maybe two) persons to this task; furthermore, they have started to make an inventory of functional and performance properties that the CMS control system should satisfy.

Clearly, the CMS research group is currently exploring which technologies can be used to bring their massive control system into check; they are willing to invest time and learn alien technology. A successful application at CERN will draw substantial attention to the behavioural modelling and analysis techniques developed within computer science.

[1] See `http://cms.cern.ch/`

The problem that we foresee is that current technology will fall short when verifying the correctness properties for the full control system consisting of approximately 20,000 finite state machines. Our current technology works well with moderately complex systems, but thus far, no systems of this scale have been verified. It is obvious that a successful verification of the properties will be a huge challenge.

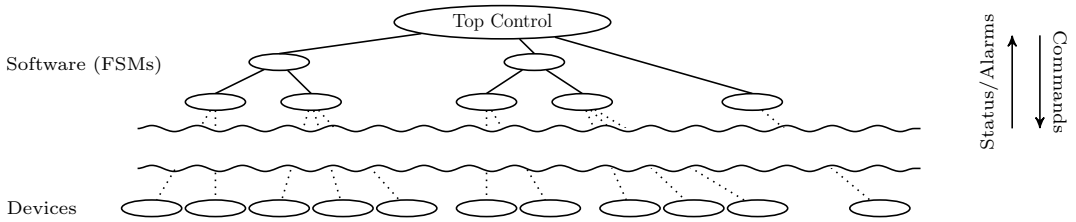

Figure 1: Architecture of the real-time monitoring and control system of the CMS experiment

The architecture of the CMS control system follows the SMI++ Framework [19]. In fact, this architecture is used in all four LHC experiments, and is quite common in other large scale experiments. A characteristic of this architecture is the regularity and relatively low complexity of individual finite state machines; the main source of complexity is in the cooperation of these finite state machines. Cooperation is strictly hierarchical, consisting of several layers, see Figure 1 for a schematic overview. At the top, there is only a single controlling finite state machine. Devices, which can be hardware or software, are typically found only at the bottom. Commands are refined and propagated down the hierarchy and status and alarms are sent upwards. It is our conviction that a successful verification of the full control system can only be achieved by taking advantage of this hierarchical architecture.

Methodology

The verification of systems as complex as the ones at CERN can be achieved only through the use of an expressive and flexible mathematical framework. The framework of *Parameterised Boolean Equation Systems (PBESs)*, first studied in [42] is ideally suited for the task at hand. This framework provides the theoretical backbone to successful verification tool sets, including the CADP tool set[2] and the mCRL2 tool set[3].

The general approach to verification using PBESs is illustrated in Figure 2: it is a modular process in which first several specifications (e.g., a behavioural specification and a modal formula) are combined to yield a PBES. The second step in the process is to use a suitable solution technique for PBESs. As a third step, the PBESs can be mapped to the simpler formalism of *Boolean Equation Systems (BESs)* [40]. The innovation in this methodology is that all artefacts can be manipulated with dedicated techniques at each stage.

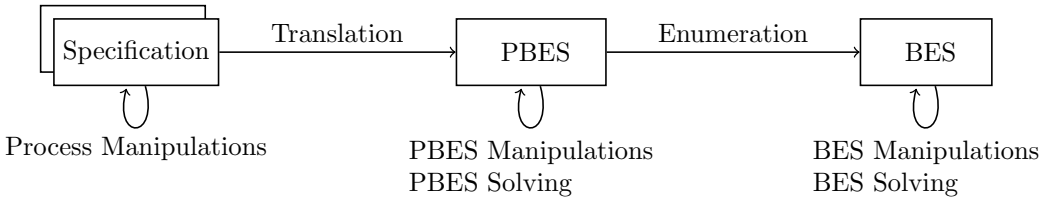

Figure 2: Verification approach using Parameterised Boolean Equation Systems

[2]See `http://www.inrialpes.fr/vasy/cadp/`
[3]see `http://www.mcrl2.org`

Many complex verification problems can be mapped onto the above verification methodology. Prime examples thereof are the verification of first-order modal μ-calculus formulae for process algebras such as LOTOS, μCRL and mCRL2, see [43, 23, 27, 29]; the problem of deciding behavioural process equivalences such as strong, weak and branching bisimilarity for infinite state systems [39, 14]; and static analysis of program code [22].

A study into the foundations of the PBES framework was conducted by Groote and Willemse, see [28, 30]. The theory of PBESs revolves around the concept of a *solution*. Automated solution techniques for PBESs, based on *symbolic approximation* and *Gauß Elimination* have been investigated by Groote and Willemse in [27, 29]. Further approaches to solving and manipulating PBESs are *pattern-matching* [28, 30], *weakening*, *strengthening*, see [30], *invariants*, see [45, 46] and automated *static analysis* techniques [44].

In practice, an important step towards computing the solution of a PBES is through *enumeration*, see [42, 15, 43]. This reduces the problem of solving a PBES to the problem of solving a BES. The latter problem is known to coincide with solving a *Parity Game*, see *e.g.* [40]. Algorithms for solving the Parity Games can therefore be used for solving BESs, and *vice versa*[4]. A bunch of solving algorithms are known from the literature, *e.g. Zielonka's algorithm (ZA)* [53], *Small Progress Measures (SPM)* [35], and *Strategy Improvement (SI)* [51].

Interestingly, all known solving algorithms use exponential time in the worst case. However, the question whether a polynomial algorithm exists at all, is a long standing open problem (the problem is known to be both in NP and in co-NP).

Challenges

The main challenge in verifying the CMS control system is in tackling the sheer size of the verification problem. This, we are convinced, is where we can employ the full strength and flexibility of the PBES framework. Unlike many frameworks, the PBES framework allows for two orthogonal strategies to address verification problems, which we address below in detail.

a) Symbolic Strategies A PBES typically is a finite, symbolic representation of a verification problem. As such, it is open to symbolic solution preserving or reflecting manipulations that aim at reducing the computational complexity that is associated to solving the PBES. Inspired by the verification problems we expect to address in the CMS control system, we foresee the following lines of research in this area:

1. The hierarchical structure of the CMS control system will lead to high degree of confluence in the state space. Confluence reduction at the level of a symbolic description of the state space has been reported to lead to exponential reductions in state space size, see *e.g.* [9], although typical reductions are often less extreme.

   However, there are two twists to using this technique. A first disadvantage of this technique is that most confluence reductions only preserve a subclass of all possible properties one wishes to express. For instance, *livelock-freedom* is not reflected under many confluence reductions.

   Second, confluence reductions based on state spaces are not sufficiently powerful, since they are blind to the specific property that is to be verified. The latter dynamically characterises when actions are important to the property and when they are not. For instance, distinguishing between a transition $p_0 \xrightarrow{a} q$ and a transition $p_1 \xrightarrow{b} q$ is needed only for (sub)properties that refer explicitly to actions a and/or b.

   Both problems disappear in the framework of PBESs: the translations to PBES typically code all relevant process and property information, and prune all irrelevant information. The challenge remains to formalise a notion of confluence for PBESs. Starting point for this will be the work on confluence reductions for processes, *e.g.* [9, 26].

[4] Henceforth, wherever we write BESs, one should also read Parity Games.

2. Real-time verification using PBES technology is currently not well supported, meaning that it is unlikely that we can verify the performance properties for the CMS control system that are currently being defined. A major problem is the fact that enumerative strategies do not work on dense domains. However, several starting points are available for solving this in the framework of PBESs.

   Of particular interest are the *regions* and *zones* concepts that can be found in the setting of Timed Automata [1], which have inspired the development of KRONOS [12] and UPPAAL [38, 6], and Parametric Timed Automata [2, 33], which have inspired tools such as TReX [4] and IMITATOR [3]. Moreover, we expect that the high degree of hierarchy in the overall CMS control system, will lead to effective combinations of the techniques for computing zones and efficient symbolic reachability algorithms. Here, *e.g.* [10, 11] serve as a starting point.

   A further challenge is to define abstract interpretation techniques for soundly approximating the solution of a PBES, and apply these to verify performance properties. An inspiration to this approach is the work conducted in [31, 34].

3. Proving the correctness of symbolic manipulations on PBESs ultimately relies on showing a mathematical correspondence between PBESs before and after these manipulations. Such proofs are notably hard to give due to the complexity of the concept of solution of a PBES, see *e.g.* [30, 15, 45, 44], but may become easier by employing finer-grained equivalences.

   This calls for a study of equivalence relations for PBESs that are finer than the equivalence relation that is induced by the *de facto* concept of solution in PBESs. Note that the equivalence relations themselves are not the objective, contrary to the process theory setting. Instead, they facilitate correctness proofs, avoiding tiresome and errorprone deductions. Moreover, symbolic manipulations are often intimately related to equivalence relations; *e.g.*, confluence reductions are related to branching bisimilarity, see *e.g.* [9], and discretisations of real-time systems are related to time-abstract bisimilarity, see *e.g.* [1].

   A by-product of such studies is an additional machinery of efficient reduction techniques: as recently argued in [37, 48], bisimilarity-inspired reduction techniques can lead to remarkable reductions in the times required to solve BESs. In a similar vein, equivalence relations have been shown to reduce the complexity class of difficult problems, see [21].

b) Brute force strategies. Symbolic manipulations generally do not solve a PBES completely. Enumeration, *i.e.*, computing a BES from a PBES, is a proven strategy for doing so. Recall that the resulting BES can be viewed as a Parity Game, so Parity Game solvers can be applied to the result directly. Despite the reductions we can achieve using symbolic manipulations, we will still be confronted with massively large BESs. This is especially the case for PBESs encoding real-time verification problems. Below, we provide details for tackling this issue.

1. The process of enumerating a PBES is akin to that of generating a state space from a symbolic representation. Given this similarity, this process is a likely candidate for parallelisation on (clusters of) multi-core workstations. Starting point will be to apply techniques for distributed LTS generation from [7] to BES generation.

   In most cases, we are interested in a partial solution: A truth value for one particular variable occurring in the PBES. The generation of BESs can be combined with on-the-fly solving strategies. Basically, using partial solutions, generating the entire BES can be avoided. Doing so in a parallel setting, however, is extremely challenging, because the optimal scheduling of subtasks starts to depend on their actual solution. We expect that the hierarchical structure of the CMS control system will allow for a perfect distribution of the enumeration process over the available processors.

   Another challenging prospect is to apply BDD-techniques during the generation and maybe even solution of large BESs. In [10], we have shown that BDD techniques can be greatly beneficial, even when only an enumerative interface to a Labelled Transition System is

present. This can dramatically decrease memory and time requirements of the generation process. Again, it is a challenge to combine this technology with solving the BES on-the-fly.

2. Combining generation and fully solving BESs may sound attractive, but it may come at the cost of some undesirable overhead. Hence we want to restrict on-the-fly partial solution computations to linear-time preprocessing. This means that in the end, a fully generated BES, encoding the core verification problem, still has to be solved.

   Practically, recent findings, see *e.g.* [20, 36], report that the SPM algorithm due to Jurdzinski [35], is outperformed by other algorithms, such as ZA and SI, but this is of course dependent on the application domain. Also, typically these algorithms have a high degree of non-determinism, and a heuristic search strategy could have dramatic impact on the practical effectiveness.

   Parallelising algorithms for solving BESs, is a promising direction. Being able to take full advantage of contemporary multi-core cpu processing power can reduce the time to solve the BES substantially. We have made an initial step in this direction with the parallelisation of the SPM algorithm, see [47].

   Parallel implementations and new parallel algorithms can shed new light on the practical applicability of parity game solvers, and reopen the debate of what the most effective (parallelisable) algorithm is, given a particular application domain. In the end, parallel algorithms may even shed new light on the true complexity of solving parity games.

3. The last line of challenges is to minimise the BES after generation, but before solving it. In particular, we want to investigate if the bisimilarity-inspired reduction techniques of [37, 48] can be made to run effectively on (a distributed cluster of) multi-core cpus. Again, the hierarchical structure of the CMS control system is expected to lead to huge size reductions. Also, it is worthwhile to investigate whether the heuristics of [20] can be readily adapted to a distributed or multi-core setting. Here our experience with distributed bisimulation reduction for Labelled Transition Systems and Markov chains [8] and the distributed detection of strongly connected components [5] will be the starting point of our research.

Addressing the abovementioned topics is pressing for solving the verification problems at CERN. However, it is clear that the outcomes of the project serve a larger problem domain and will add to the applicability of model checking for verifying industrial systems in general.

Related Work

Systematic investigations of the foundations of the PBES framework have been conducted only recently, see [28, 30, 15, 45, 46, 44]; many of these investigations were done in the context of the Dutch *COMFORTS* project that revolves around the PBES framework. The number of application areas for PBESs, as mentioned earlier, has been steadily increasing, see in particular the works of Chen *et al* [14] and Lin [39] on process equivalence relations; the initiating works by Mateescu [42, 41], Groote and Mateescu [23], Groote and Willemse [27, 29], Van Dam, Ploeger and Willemse [15], and Mateescu and Thivolle [43] on model checking data-intensive systems. The inspiring work by Zhang and Cleaveland [52] is the first in which a formalism based on BESs is used for verifying a formalism in which both data and time play a role. The mCRL2 tool suite offers a range of tools for constructing, manipulating and solving PBESs.

Closely related to the works on PBESs is the steadily increasing body of work on verifying μ-calculus formulae using other frameworks. Among these, we find Parity Games [17], which have been studied in their own right, and for which many tantalising results have appeared over the years, see [51, 35, 49].

Local, National and International Collaboration

The research conducted in the project will take place in the *Design and Analysis of Systems* (DAS) research group at the Eindhoven University of Technology (TU/e), headed by Prof.dr.ir.

J.F. Groote, and the *Formal Methods and Tools* (FMT) research group at the University of Twente (UT), headed by Prof.dr. J.C. van de Pol.

The DAS group specialises in developing verification methods and analysis techniques for complex reactive systems. The theory, developed in this group, is consolidated in the mCRL2 specification language, see [24, 25], and its verification tools are based on PBES technology. The FMT group of the University of Twente develops and uses formal techniques and tools as a means to support the development of software. This includes the development of formal theories of concurrency, design methodologies for distributed systems, and correctness assessment using verification or validation techniques.

Some projects currently running in these groups that are relevant to the research are the following:

1. *VeriGem* (TU/e and UT): addressing the verification problem with brute-force methods by using multi-core computers and clusters of computers. Among others, BESs, generated from PBESs, serve as input to the project.

2. *COMFORTS* (TU/e): translating verification problems, such as various process equivalences and model checking problems, into the PBES framework, and study techniques to analyse these within the PBES framework.

3. *MCMC* (UT): addresses distributed algorithms for model checking, investigating, among others, non-standard solutions, such as the use of external memory (disk) or specialised hardware (stream processors).

Both groups have strong ties with national and international research groups and institutes. These include groups in Nijmegen, Amsterdam, Aachen, Reykjavik, Grenoble, Maryland. The research will be conducted in close collaboration with Prof.dr. R. Cleaveland of the University of Maryland, and director of the Fraunhofer USA institute. The collaborations with the CMS research group at CERN are facilitated through our cooperations with dr. F. Glege.

1.2 Application Perspective

The basic motivation for studying and improving behavioural analysis techniques is to improve computer controlled systems. Constructing correct programs, especially in the context of parallel and distributed systems, is a tremendously complex task. This leaves little room for doubt on the importance of behavioural analysis techniques, and, especially improving upon these, as they typically contribute to the quality of programmed systems and the effectiveness to construct them.

Throughout the years there has been a growing interest, albeit slowly, in the application of formal techniques. This slow uptake has a number of reasons, *viz.* the gap between the workforce of programmers that expect out of the box tools for verification and the required knowledge and skills to fruitfully use contemporary behavioural analysis techniques. The fact that most tools are 'academic', without too much concern for usage by the average programmer does not help, as does the lack of desire in companies to change their current way of working. Tool sets that support practical applications, *e.g.* mCRL2 and CADP, typically have a steep learning curve and require a thorough training to use. Furthermore, the lack of a commonly agreed upon, standardised language that is sufficiently expressive and a suite of accompanying analysis methods, does not help.

Our primary concern is that technology is being developed from a scientific perspective. We want to know its strengths and weaknesses, and use this knowledge to come up with more effective and conceptually cleaner methodologies. We also find it important that the tools, methods and techniques are freely published to foster a community of researchers. Therefore, we take great care in publishing both scientific and introductory information about our methods. Moreover, all our software is provided in an accessible form, free for anyone to use and contribute to, even commercially. So far, this attitude has led to a slow but steady uptake of the mCRL2 toolset

in the academic world (*e.g.*, Luxembourg, Darmstadt, Teheran, Amsterdam, Eindhoven, Twente, Leiden, Geneva, Copenhagen, *etc*).

In recent years, several autonomous market parties in the Netherlands have started to develop and deploy formal analysis tools geared towards programming embedded systems (notably, Verum with the ASD toolkit, and Imtech with I-Matic). While these companies have their proprietary specification languages, verification issues are typically delegated to academic toolsets, including FDR and mCRL2. Besides that, they invest heavily in model based code generation and they go to great lengths to guarantee customers satisfaction. The existence of such companies made it possible for multinationals like Philips to prescribe the use of formal methods in some development trajectories of their products (*e.g.*, Philips Healthcare in Best).

In view of the above developments, we have little concern that the technology developed as a result of the research project can find its way in society. In isolation, our projected achievements at CERN will already be an effective outlet.

2 Project Planning

The four-year planning of the project is as detailed by the Gantt Chart in Figure 3. Numbers in the bars correspond to the research directions listed under *a* (page 4) and *b* (page 5); dashed lines indicate common issues between activities. The main tasks are clustered horizontally in three work packages; WP1, assigned to Eindhoven, and WP2, assigned to Twente, will be allocated to Ph.D. students working in the project, and can be run independently. Training of the Ph.D. students is facilitated through the research school IPA. WP3 will be allocated to a part-time scientific programmer. The scheme allows for flexibility and switching of tasks in the work packages if so required. The dependencies of tasks a.1 and a.2 on a.3 is weak and can be coped with in case a.3 should not lead to success; task a.3 is not a full-time task.

Research conducted within the project will combine advancements in the PBES theory with periods of reflection on these advances. Reflection is achieved via applications to the CMS control system. Programming is taken care of chiefly in WP3, with guidance from WP1 and WP2; we expect programming tasks to be relatively small given the facilities that are currently available.

Given the priority of verification of functional properties of the CMS control system, we prioritise research that will lead to results in these domains most quickly.

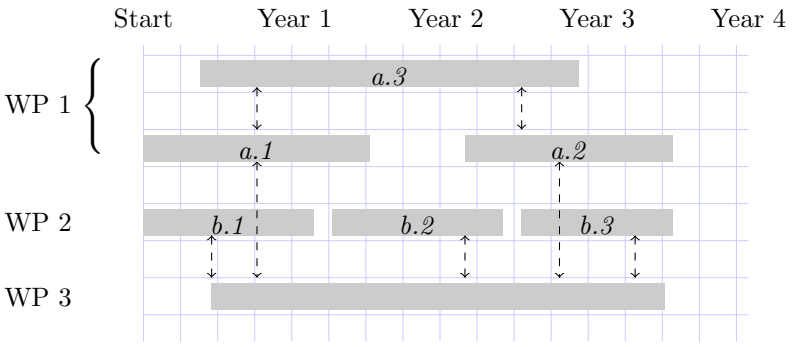

Figure 3: Four-year workplan for the project.

3 Literature References

Key Publications of the Team

[1] J.C. van de Pol and M. Weber. A Multi-Core Solver for Parity Games. In Electron. Notes Theor. Comput. Sci., 220(2):19–34, 2008.

[2] S.C.C. Blom, B. Lisser, J.C. van de Pol and M. Weber. A Database Approach to Distributed State-Space Generation. *Journal of Logic and Computation*, 2009 (to appear; available online).

[3] J.F. Groote and T.A.C. Willemse. Parameterised boolean equation systems. *Theoretical Computer Science*, volume 343:332–369, 2005.

[4] S.M. Orzan, J.W. Wesselink, and T.A.C. Willemse. Static analysis techniques for Parameterised Boolean Equation Systems. In S. Kowalewski and A. Philippou, editors, *TACAS 2009*, volume 5505 of *Lecture Notes in Computer Science*, pages 230–245, 2009.

[5] D. Zhang and R. Cleaveland. Fast Generic Model-Checking for Data-Based Systems. In F. Wang, editor, *Formal Techniques for Networked and Distributed Systems - FORTE 2005*, volume 3731 of *Lecture Notes in Computer Science*, pp. 83–97, 2005

Bibliography

[1] R. Alur and D.L. Dill. A theory of timed automata. In *Theoretical Computer Science*, volume 126:183–235, 1994.

[2] R. Alur, T.A. Henzinger, and M.Y. Vardi. Parametric Real-Time Reasoning. In *Proceedings of the 25th Annual Symposium on Theory of Computing (STOC)*, pages 592–601. ACM Press, 1993.

[3] É. André, T. Chatain, L. Fribourg and E. Encrenaz. An inverse method for parametric timed automata. *Electron. Notes Theor. Comput. Sci.*, 223:29–46, 2008.

[4] A. Annichini, A. Bouajjani, and Mihaela Sighireanu. TReX: A Tool for Reachability Analysis of Complex Systems. In *Proceedings of Computer Aided Verification*, volume 1855 of *LNCS*, pages 368–372. Springer-Verlag, 2001.

[5] J. Barnat, J. Chaloupka and J.C. van de Pol. Distributed Algorithms for SCC Decomposition. *Journal of Logic and Computation*, 2009 (to appear).

[6] G. Behrmann, A. David and K.G. Larsen. A Tutorial on Uppaal. In *SFM-RT Revised Lectures*, volume 3185, pages 200–236, *LNCS*, Springer, 2004.

[7] S.C.C. Blom, B. Lisser, J.C. van de Pol and M. Weber. A Database Approach to Distributed State-Space Generation. *Journal of Logic and Computation*, 2009 (to appear).

[8] S.C.C. Blom, B.R.H.M. Haverkort, G.W.M. Kuntz and J.C. van de Pol. Distributed Markovian Bisimulation Reduction aimed at CSL Model Checking. *Electr. Notes Theor. Comput. Sci.*, volume 220(2):35–50, 2008.

[9] S.C.C Blom and J.C. van de Pol. State Space Reduction by Proving Confluence. In E. Brinksma and K.G. Larsen, editors, *Proceedings CAV*, volume 2404 of *LNCS*, pages 596–609. Springer, 2002.

[10] S.C.C Blom and J.C. van de Pol. Symbolic Reachability for Process Algebras with Recursive Data Types. In *Proceedings ICTAC*, volume 5160 of *LNCS*, pages 81–95. Springer, 2008.

[11] S.C.C Blom, J.C. van de Pol and M. Weber. Bridging the Gap between Enumerative and Symbolic Model Checkers. *Technical Report TR-CTIT-09-30*, University of Twente, 2009.

[12] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, and S. Yovine. Kronos: A Model-Checking Tool for Real-Time Systems. In A.J. Hu and M.Y. Vardi, editors, *Proceedings of the 10th International Conference on Computer Aided Verification, Vancouver, Canada*, volume 1427 of *LNCS*, pages 546–550. Springer-Verlag, 1998.

[13] R.E. Bryant, S.K. Lahiri, and S.A. Seshia. Modeling and verifying systems using a logic of counter arithmetic with lambda expressions and uninterpreted functions. In *CAV 2002*, volume 2404 of *Lecture Notes in Computer Science*, pages 78–92. Springer-Verlag, 2002.

[14] T. Chen, B. Ploeger, J. van de Pol, and T.A.C. Willemse. Equivalence checking for infinite systems using parameterized boolean equation systems. In *Proc. 18th Int'l Conference on CONCUR*, LNCS 4703, pages 120–135. Springer, 2007.

[15] A. van Dam, B. Ploeger, and T.A.C. Willemse. Instantiation for Parameterised Boolean Equation Systems. In J.S. Fitzgerald, A.E. Haxthausen, and H. Yenigün, editors, *ICTAC 2008*, volume 5160 of *Lecture Notes in Computer Science*, pages 440–454, 2008.

[16] L. Doyen. Robust parametric reachability for timed automata. *Inf. Process. Lett.*, 102(5):208–213, 2007.

[17] E.A. Emerson and C.S. Jutla. Tree Automata, Mu Calculus and Determinacy (extended abstract). In *Foundations of Computer Science*, pages 368–377. IEEE, 1991.

[18] M.V. Espada and J.C. van de Pol. An abstract interpretation toolkit for μCRL. *Formal Methods in System Design*, 30(3):249–273, 2007.

[19] B. Franek and C. Gaspar. SMI++ Object-Oriented Framewrok for Designing and Implementing Distributed Control Systems. *IEEE Transactions on Nuclear Science*, 52(4):891–895, 2005.

[20] O. Friedmann and M. Lange. Solving Parity Games in Practice. In *Proceedings ATVA*, 2009, to appear.

[21] C. Fritz and T. Wilke. Simulation Relations for Alternating Parity Automata and Parity Games. In O.H. Ibarra and Z. Dang, editors, *Proceedings DLT 2006*, volume 4036 of *LNCS*, pages 59–70. Springer-Verlag, 2006.

[22] M.M. Gallardo, C. Joubert, and P. Merino. Implementing influence analysis using parameterised boolean equation systems. In *Proceedings of ISOLA'06*. IEEE Computer Society Press, November 2006.

[23] J.F. Groote and R. Mateescu. Verification of temporal properties of processes in a setting with data. In A.M. Haeberer, editor, *AMAST'98*, volume 1548 of *LNCS*, pages 74–90. Springer-Verlag, 1999.

[24] J.F. Groote, A. Mathijssen, M.A. Reniers, Y.S. Usenko and M. van Weerdenburg. The Formal Specification Language mCRL2. In *Methods for Modelling Software Systems (MMOSS)*, volume 06351 of *Dagstuhl Seminar Proceedings*, 2007.

[25] J.F. Groote, A. Mathijssen, M.A. Reniers, Y.S. Usenko and M. van Weerdenburg. Analysis of Distributed Systems with mCRL2. In M. Alexander and W. Gardner, editors, *Process Algebra for Parallel and Distributed Processing*. Chapman & Hall/CRC Computational Science, 2008.

[26] J.F. Groote and M.P.A. Sellink. Confluence for Process Verification. *Theoretical Computer Science*, 170(1-2):47–81, 1996.

[27] J.F. Groote and T.A.C. Willemse. A checker for modal formulas for processes with data. In F.S. de Boer, M.M. Bosangue, S. Graf, and W.-P. de Roever, editors, *Proceedings of FMCO 2003*, volume 3188 of *LNCS*, pages 223–239. Springer-Verlag, 2004.

[28] J.F. Groote and T.A.C. Willemse. Parameterised boolean equation systems (extended abstract). In P. Gardner and N. Yoshida, editors, *Proceedings of CONCUR 2004*, volume 3170 of *LNCS*, pages 308–324. Springer-Verlag, 2004.

[29] J.F. Groote and T.A.C. Willemse. Model-checking processes with data. *Science of Computer Programming*, 56:251–273, 2005.

[30] J.F. Groote and T.A.C. Willemse. Parameterised boolean equation systems. In *Theoretical Computer Science*, 343:332–369, 2005.

[31] N. Halbwachs, Y.-E. Proy, and P. Roumanoff. Verification of Real-Time Systems using Linear Relation Analysis. In *Formal Methods in System Design*, 11(2):157–185, 1997.

[32] Y.L. Hwong, A. Racz, B. Beccati, C. Deldicque, C. Schwick, D. Gigi, E. Cano, E. Meschi, F. Glege, F. Meijers, H. Sakulin, J.A. Coarasa, J.F. Laurens, J. Gutleber, L. Orsini, M. Ciganek, M. Simon, M. Zanetti, R. Gomez-Reino, R. Moser, S. Cittolin, J.F. Groote, T.A.C. Willemse, A. Meyer, D. Hatton, U. Behrens, D. Shpakov, H. Cheung, J.A. Lopez-Perez, K. Biery, R.K. Mommsen, V. O'Dell, A.S. Yoon, C. Loizides, C. Paus, F. Ma, G. Bauer, J.F. Serrano Margaleff, K. Sumorok, S. Erhan, A. Petrucci, J. Branson, M. Pieri and M. Sani. An Analysis of the Control Hierarchy Modelling of the CMS Detector Control System. In *Proceedings ICALEPCS*, 2009 (to appear).

[33] T.S. Hune, J.M.T. Romijn, M.I.A. Stoelinga, and F.W. Vaandrager. Linear parametric model checking of timed automata. *Journal of Logic and Algebraic Programming*, 52-53:183–220, 2002.

[34] B. Jeannet. Dynamic Partitioning In Linear Relation Analysis. Application To The Verification Of Reactive Systems. *Formal Methods in System Design*, 23(1):5-37, 2003.

[35] M. Jurdzínski. Small Progress Measures for Solving Parity Games. In *Proceedings STACS 2000*, volume 1770 of *LNCS*, pages 290–301. Springer, 2000.

[36] J. Keiren. An experimental study of algorithms and optimisations for parity games, with an application to Boolean Equation Systems. MSc thesis, Eindhoven University of Technology, 2009.

[37] J. Keiren and T.A.C. Willemse. Bisimulation minimisations for Boolean equation systems, 2009. In *Proceedings HVC 2009*, *LNCS*. To appear.

[38] K. Larsen, P. Pettersson, and W. Yi. UPPAAL in a nutshell. *Springer International Journal of Software Tools for Technology Transfer*, 1, 1997.

[39] H. Lin. Symbolic transition graph with assignment. In U. Montanari and V. Sassone, editors, *Proceedings of CONCUR 1996*, volume 1119 of *LNCS*, pages 50–65. Springer-Verlag, 1996.

[40] A. Mader. *Verification of Modal Properties Using Boolean Equation Systems.* PhD thesis, Technical University of Munich, 1997.

[41] R. Mateescu. Local Model-Checking of an Alternation-Free Value-Based Modal Mu-Calculus. In *Proceedings of the 2nd International Workshop on Verification, Model Checking and Abstract Interpretation VMCAI'98 (Pisa, Italy)*, September 1998.

[42] R. Mateescu. *Vérification des propriétés temporelles des programmes parallèles.* PhD thesis, Institut National Polytechnique de Grenoble, 1998.

[43] R. Mateescu and D. Thivolle. A model checking language for concurrent value-passing systems. In *Proceedings of FM 2008*, LNCS 5014. Springer-Verlag, 2008.

[44] S.M. Orzan, J.W. Wesselink, and T.A.C. Willemse. Static analysis techniques for Parameterised Boolean Equation Systems. In S. Kowalewski and A. Philippou, editors, *TACAS 2009*, volume 5505 of *Lecture Notes in Computer Science*, pages 230–245, 2009.

[45] S.M. Orzan and T.A.C. Willemse. Invariants for Parameterised Boolean Equation Systems (extended abstract). In F. van Breugel and M. Chechik, editors, *CONCUR 2008*, volume 5201 of *Lecture Notes in Computer Science*, pages 187–202, 2008.

[46] S.M. Orzan and T.A.C. Willemse. Invariants for Parameterised Boolean Equation Systems. *Theor. Comp. Sc.*, to appear.

[47] J.C. van de Pol and M. Weber. A Multi-Core Solver for Parity Games. In Electron. Notes Theor. Comput. Sci., 220(2):19–34, 2008.

[48] M.A. Reniers and T.A.C. Willemse. Analysis of Boolean Equation Systems through Structure Graphs. In *Proc. of SOS 2009*, *EPTCS*, to appear.

[49] S. Schewe. An Optimal Strategy Improvement Algorithm for Solving Parity and Payoff Games. In M. Kaminski and S. Martini, editors, *Proceedings of CSL*, volume 5213 of *LNCS*, pp 369–384. Springer, 2008.

[50] C. Stirling. Local Model Checking Games. In I. Lee and S.A. Smolka, editors, *Proceedings CONCUR '95*, volume 962 of *LNCS*, pp. 1–11. Springer, 1995.

[51] J. Vöge and M. Jurdzinski. A Discrete Strategy Improvement Algorithm for Solving Parity Games. In E.A. Emerson and A.P. Sistla, editors, *Proceedings of CAV 2000*, volume 1855 of *LNCS*, pp. 202–215. Springer, 2000.

[52] D. Zhang and R. Cleaveland. Fast Generic Model-Checking for Data-Based Systems. In F. Wang, editor, *Formal Techniques for Networked and Distributed Systems - FORTE 2005*, volume 3731 of *Lecture Notes in Computer Science*, pp. 83–97, 2005

[53] W. Zielonka. Infinite Games on Finitely Coloured Graphs with Applications to Automata on Infinite Trees. *Theoretical Computer Science*, 200(1-2):135–183, 1998.

4 Requested Budget

It is not expected that the project will require additional computer resources. The project will require the standard salary and bench fees for two PhD students for four years (2.0 fte) and the salary of a scientific programmer for the duration of three years (0.3 fte). The total budget is 440k€.