UMR IRISA

Activity Report 2023

Team HYCOMES

ModÃ©lisation hybride & conception par contrats pour les systÃ¨mes embarquÃ©s multi-physiques

Joint team with Centre Inria de l'Université de Rennes

D4 – Language and Software Engineering

Université de Rennes | ENS rennes | INSA Institut National des Sciences Appliquées Rennes · CNRS · Inria · Inserm · CentraleSupélec · IMT Atlantique Bretagne-Pays de la Loire École Mines-Télécom · Université Bretagne Sud UBS

Team HYCOMES IRISA Activity Report 2023

Contents

Project-Team HYCOMES

Creation of the Project-Team: 2016 September 01

Keywords

Computer sciences and digital sciences

A2. – Software

A2.1. – Programming Languages

A2.1.1. – Semantics of programming languages

A2.1.5. – Constraint programming

A2.1.9. – Synchronous languages

A2.1.10. – Domain-specific languages

A2.2. – Compilation

A2.2.1. – Static analysis

A2.2.8. – Code generation

A2.3. – Embedded and cyber-physical systems

A2.3.1. – Embedded systems

A2.3.2. – Cyber-physical systems

A2.3.3. – Real-time systems

A2.4. – Formal method for verification, reliability, certification

A2.4.1. – Analysis

A2.4.3. – Proofs

A2.5. – Software engineering

A2.5.1. – Software Architecture & Design

A2.5.2. – Component-based Design

A6. – Modeling, simulation and control

A6.1. – Methods in mathematical modeling

A6.1.1. – Continuous Modeling (PDE, ODE)

A6.1.5. – Multiphysics modeling

A6.3. – Computation-data interaction

A6.3.4. – Model reduction

A8. – Mathematics of computing

A8.4. – Computer Algebra

Other research topics and application domains

- B4. – Energy
- B4.4. – Energy delivery
- B4.4.1. – Smart grids
- B5.1. – Factory of the future
- B5.2. – Design and manufacturing
- B5.9. – Industrial maintenance
- B8. – Smart Cities and Territories
- B8.1. – Smart building/home
- B8.1.1. – Energy for smart buildings
- B8.2. – Connected city
- B8.3. – Urbanism and urban planning

1 Team members, visitors, external collaborators

Research Scientists

- Benoit Caillaud [Team leader, INRIA, Senior Researcher, HDR]
- Albert Benveniste [INRIA, Emeritus, HDR]
- Yahao Chen [INRIA, ISFP, from Dec 2023]
- Khalil Ghorbal [INRIA, Researcher]

PhD Students

- Maxime Bridoux [INRIA]
- Joan Thibault [University of Rennes, until Apr 2023]

Technical Staff

- Mathias Malandain [Inria, Engineer, shared with the I4S project-team]

Administrative Assistant

- Armelle Mozziconacci [CNRS]

2 Overall objectives

Hycomes was created as a local team of the Rennes - Bretagne Atlantique Inria research center in 2013 and has been created as an Inria Project-Team in 2016. The team is focused on two topics in cyber-physical systems design:

- Hybrid systems modeling, with an emphasis on the design of modeling languages in which software systems, in interaction with a complex physical environment, can be modelled, simulated and verified. A special attention is paid to the mathematical rigorous semantics of these languages, and to the correctness (wrt. such semantics) of the simulations and of the static analyses that must be performed during compilation. The Modelica language is the main application field. The team aims at contributing language extensions facilitating the modeling of physical domains which are poorly supported by the Modelica language. The Hycomes team is also designing new structural analysis methods for hybrid (aka. multi-mode) Modelica models. New simulation and verification techniques for large Modelica models are also in the scope of the team.
- Contract-based design and interface theories, with applications to requirements engineering in the context of safety-critical systems design. The objective of our research is to bridge the gap between system-level requirements, often expressed in natural, constrained or semi-formal languages and formal models, that can be simulated and verified.

3 Research program

3.1 Hybrid Systems Modeling

Systems industries today make extensive use of mathematical modeling tools to design computer controlled physical systems. This class of tools addresses the modeling of physical systems with models that are simpler than usual scientific computing problems by using only Ordinary Differential Equations (ODE) and Difference Equations but not Partial Differential Equations (PDE). This family of tools first emerged in the 1980's with SystemBuild by MatrixX (now distributed by National Instruments) followed soon by Simulink by Mathworks, with an impressive subsequent development.

In the early 90's control scientists from the University of Lund (Sweden) realized that the above approach did not support component based modeling of physical systems with reuse [1]. For instance, it was not easy to draw an electrical or hydraulic circuit by assembling component models of the various devices. The development of the Omola language by Hilding Elmqvist was a first attempt to bridge this gap by supporting some form of Differential Algebraic Equations (DAE) in the models. Modelica quickly emerged from this first attempt and became in the 2000's a major international concerted effort with the Modelica Consortium. A wider set of tools, both industrial and academic, now exists in this segment [2]. In the Electronic Design Automation (EDA) sector, VHDL-AMS was developed as a standard [71] and also enables the use of differential algebraic equations. Several domain-specific languages and tools for mechanical systems or electronic circuits also support some restricted classes of differential algebraic equations. Spice is the historic and most striking instance of these domain-specific languages/tools [3]. The main difference is that equations are hidden and the fixed structure of the differential algebraic results from the physical domain covered by these languages.

Despite the fact that these tools are now widely used by a number of engineers, they raise a number of technical difficulties. The meaning of some programs, their mathematical semantics, is indeed ambiguous. A main source of difficulty is the correct simulation of continuous-time dynamics, interacting with discrete-time dynamics: How the propagation of mode switchings should be handled? How to avoid artifacts due to the use of a global ODE solver causing unwanted coupling between seemingly non interacting subsystems? Also, the mixed use of an equational style for the continuous dynamics with an imperative style for the mode changes and resets, is a source of difficulty when handling parallel composition. It is therefore not uncommon that tools return complex warnings for programs with many different suggested hints for fixing them. Yet, these "pathological" programs can still be executed, if wanted so, giving surprising results — See for instance the Simulink examples in [26], [19] and [20].

Indeed this area suffers from the same difficulties that led to the development of the theory of synchronous languages as an effort to fix obscure compilation schemes for discrete time equation based languages in the 1980's. Our vision is that hybrid systems modeling tools deserve similar efforts in theory as synchronous languages did for the programming of embedded systems.

3.2 Background on non-standard analysis

Non-Standard analysis plays a central role in our research on hybrid systems modeling [19, 26, 21, 20, 24], [3]. The following text provides a brief summary of this theory and gives some hints on its usefulness in the context of hybrid systems modeling. This presentation is based on our paper [2], a chapter of Simon Bliudze's PhD thesis [33], and a recent presentation of non-standard analysis, not axiomatic in style, due to the mathematician Lindström [80].

Non-standard numbers allowed us to reconsider the semantics of hybrid systems and propose a radical alternative to the *super-dense time semantics* developed by Edward Lee and his team as part of the Ptolemy II project, where cascades of successive instants can occur in zero time by using $\mathbb{R}_+ \times \mathbb{N}$ as a time index. In the non-standard semantics, the time index is defined as a set $\mathbb{T} = \{n\partial \mid n \in {}^*\mathbb{N}\}$, where ∂ is an *infinitesimal* and ${}^*\mathbb{N}$ is the set of *non-standard integers*. Remark that (1) $\mathbb{T}$ is dense in $\mathbb{R}_+$, making it "continuous", and (2) every $t \in \mathbb{T}$ has a predecessor in $\mathbb{T}$ and a successor in $\mathbb{T}$, making it "discrete". Although it is not effective from a computability point of view, the *non-standard semantics* provides a framework that is familiar to the computer scientist and at the same time efficient as a symbolic abstraction. This makes it an excellent candidate for the development of provably correct compilation schemes and type systems for hybrid systems modeling languages.

Non-standard analysis was proposed by Abraham Robinson in the 1960s to allow the explicit manipulation of "infinitesimals" in analysis [93, 58, 53]. Robinson's approach is axiomatic; he proposes adding three new axioms to the basic Zermelo-Fraenkel (ZFC) framework. While the need for non-standard analysis (in addition to the usual or standard analysis) has long agitated the mathematical community, it is not our purpose to debate such aspects. The important thing for us is that non-standard analysis allows the use of the non-standard discretization of continuous dynamics "as if" it was operational.

[1] Origins of Equation-Based Modeling
[2] SimScape by Mathworks, Amesim by LMS International, now Siemens PLM, and more.
[3] Such as the Spice3 electronic circuit simulator.

Not surprisingly, such an idea is not novel. Iwasaki et al. [73] first proposed using non-standard analysis to discuss the nature of time in hybrid systems. Bliudze and Krob [34, 33] have also used non-standard analysis as a mathematical support for defining a system theory for hybrid systems. They discuss in detail the notion of "system" and investigate computability issues. The formalization they propose closely follows that of Turing machines, with a memory tape and a control mechanism.

3.3 Structural Analysis of DAE Systems

The Modelica language is based on Differential Algebraic Equations (DAE). The general form of a DAE is given by:

$$F(t,x,x',x'',\ldots) \tag{1}$$

where F is a system of n_e equations $\{f_1,\ldots,f_{n_e}\}$ and x is a finite list of n_v independent real-valued, smooth enough, functions $\{x_1,\ldots,x_{n_v}\}$ of the independent variable t. We use x' as a shorthand for the list of first-order time derivatives of x_j, $j=1,\ldots,n_v$. High-order derivatives are recursively defined as usual, and $x^{(k)}$ denotes the list formed by the k-th derivatives of the functions x_j. Each f_i depends on the scalar t and some of the functions x_j as well as a finite number of their derivatives.

Let $\sigma_{i,j}$ denote the highest differentiation order of variable x_j effectively appearing in equation f_i, or $-\infty$ if x_j does not appear in f_i. The *leading variables* of F are the variables in the set

$$\left\{x_j^{(\sigma_j)} \mid \sigma_j = \max_i \sigma_{i,j}\right\}$$

The *state variables* of F are the variables in the set

$$\left\{x_j^{(\nu_j)} \mid 0 \leq \nu_j < \max_i \sigma_{i,j}\right\}$$

A leading variable $x_j^{(\sigma_j)}$ is said to be *algebraic* if $\sigma_j = 0$ (in which case, neither x_j nor any of its derivatives are state variables). In the sequel, v and u denote the leading and state variables of F, respectively.

DAE are a strict generalization of *ordinary differential equations* (*ODE*), in the sense that it may not be immediate to rewrite a DAE as an explicit ODE of the form $v = G(u)$. The reason is that this transformation relies on the Implicit Function Theorem, requiring that the Jacobian matrix $\frac{\partial F}{\partial v}$ to be full rank. This is, in general, not the case for a DAE. Simple examples, like the two-dimensional fixed-length pendulum in Cartesian coordinates [89], exhibit this behaviour.

For a square DAE of dimension n (i.e., we now assume $n_e = n_v = n$) to be solved in the neighborhood of some (v^*, u^*), one needs to find a set of non-negative integers $C = \{c_1,\ldots,c_n\}$ such that system

$$F^{(C)} = \{f_1^{(c_1)},\ldots,f_n^{(c_n)}\}$$

can locally be made explicit, i.e., the Jacobian matrix of $F^{(C)}$ with respect to its leading variables, evaluated at (v^*, u^*), is nonsingular. The smallest possible value of $\max_i c_i$ for a set C that satisfies this property is the *differentiation index* [45] of F, that is, the minimal number of time differentiations of all or part of the equations f_i required to get an ODE.

In practice, the problem of automatically finding a minimal solution C to this problem quickly becomes intractable. Moreover, the differentiation index may depend on the value of (v^*, u^*). This is why, in lieu of numerical nonsingularity, one is interested in the *structural nonsingularity* of the Jacobian matrix, i.e., its almost certain nonsingularity when its nonzero entries vary over some neighborhood. In this framework, the *structural analysis* (*SA*) of a DAE returns, when successful, values of the c_i that are independent from a given value of (v^*, u^*).

A renowned method for the SA of DAE is the *Pantelides method*; however, Pryce's Σ-*method* is introduced also in what follows, as it is a crucial tool for our works.

3.3.1 Pantelides method

In 1988, Pantelides proposed what is probably the most well-known SA method for DAE [89]. The main idea of his work is that the structural representation of a DAE can be condensed into a bipartite graph whose left nodes (resp. right nodes) represent the equations (resp. the variables), and in which an edge exists if and only if the variable occurs in the equation.

By detecting specific subsets of the nodes, called *Minimally Structurally Singular* (*MSS*) subsets, the Pantelides method iteratively differentiates part of the equations until a perfect matching between the equations and the leading variables is found. One can easily prove that this is a necessary and sufficient condition for the structural nonsingularity of the system.

The main reason why the Pantelides method is not used in our work is that it cannot efficiently be adapted to multimode DAE (mDAE). As a matter of fact, the adjacency graph of a mDAE has both its nodes and edges parametrized by the subset of modes in which they are active; this, in turn, requires that a parametrized Pantelides method must branch every time no mode-independent MSS is found, ultimately resulting, in the worst case, in the enumeration of modes.

3.3.2 Pryce's Sigma-method

Albeit less renowned that the Pantelides method, Pryce's Σ-method [90] is an efficient SA method for DAE, whose equivalence to the Pantelides method has already been established. This method consists in solving two successive problems, denoted by primal and dual, relying on the Σ-*matrix*, or *signature matrix*, of the DAE F.

This matrix is given by:

$$\Sigma = (\sigma_{ij})_{1 \le i,j \le n} \tag{2}$$

where σ_{ij} is equal to the greatest integer k such that $x_j^{(k)}$ appears in f_i, or $-\infty$ if variable x_j does not appear in f_i. It is the adjacency matrix of a weighted bipartite graph, with structure similar to the graph considered in the Pantelides method, but whose edges are weighted by the highest differentiation orders. The $-\infty$ entries denote non-existent edges.

The *primal problem* consists in finding a *maximum-weight perfect matching* (*MWPM*) in the weighted adjacency graph. This is actually an assignment problem for which several standard algorithms exist, such as the push-relabel algorithm [67] or the Edmonds-Karp algorithm [60] to only give a few. However, none of these algorithms are easily parametrizable, even for applications to mDAE systems with a fixed number of variables.

The *dual problem* consists in finding the component-wise minimal solution (C, D) where $C = \{c_1, \dots, c_n\}$ and $D = \{d_1, \dots, d_n\}$) to a given linear programming problem, defined as the dual of the aforementioned assignment problem. This is performed by means of a *fixpoint iteration* (*FPI*) that makes use of the MWPM found as a solution to the primal problem, described by the set of tuples $\{(i, j_i)\}_{i \in \{1,\dots,n\}}$:

1. Initialize $\{c_1, \dots, c_n\}$ to the zero vector.
2. For every $j \in \{1, \dots, n\}$,
$$d_j \leftarrow \max_i (\sigma_{ij} + c_i)$$
3. For every $i \in \{1, \dots, n\}$,
$$c_i \leftarrow d_{j_i} - \sigma_{i,j_i}$$
4. Repeat Steps 2 and 3 until convergence is reached.

From the results proved by Pryce in [90], it is known that the above algorithm terminates if and only if it is provided a MWPM, and that the values it returns are independent of the choice of a MWPM whenever there exist several such matchings. In particular, a direct corollary is that the Σ-method succeeds as long as a perfect matching can be found between equations and variables.

Another important result is that, if the Pantelides method succeeds for a given DAE F, then the Σ-method also succeeds for F and the values it returns for C are exactly the differentiation indices for the equations that are returned by the Pantelides method. As for the values of the d_j, being given by $d_j = \max_i(\sigma_{ij} + c_i)$, they are the differentiation indices of the leading variables in $F^{(C)}$.

Working with this method is natural for our works, since the algorithm for solving the dual problem is easily parametrizable for dealing with multimode systems, as shown in our recent paper [42].

3.3.3 Block triangular decomposition

Once structural analysis has been performed, system $F^{(C)}$ can be regarded, for the needs of numerical solving, as an algebraic system with unknowns $x_j^{(d_j)}$, $j = 1 \ldots n$. As such, (inter)dependencies between its equations must be taken into account in order to put it into block triangular form (BTF). Three steps are required:

1. the *dependency graph* of system $F^{(C)}$ is generated, by taking into account the perfect matching between equations $f_i^{(c_i)}$ and unknowns $x_j^{(d_j)}$;

2. the *strongly connected components* (*SCC*) in this graph are determined: these will be the *equation blocks* that have to be solved;

3. the *block dependency graph* is constructed as the condensation of the dependency graph, from the knowledge of the SCC; a BTF of system $F^{(C)}$ can be made explicit from this graph.

3.4 Contract-Based Design, Interfaces Theories, and Requirements Engineering

System companies such as automotive and aeronautic companies are facing significant difficulties due to the exponentially raising complexity of their products coupled with increasingly tight demands on functionality, correctness, and time-to-market. The cost of being late to market or of imperfections in the products is staggering as witnessed by the recent recalls and delivery delays that many major car and airplane manufacturers had to bear in the recent years. The root causes of these design problems are complex and relate to a number of issues ranging from design processes and relationships with different departments of the same company and with suppliers, to incomplete requirement specification and testing.

We believe the most promising means to address the challenges in systems engineering is to employ formal design methodologies that seamlessly and coherently combine the various viewpoints of the design space (behavior, time, energy, reliability, ...), that provide the appropriate abstractions to manage the inherent complexity, and that can provide correct-by-construction implementations. The following issues must be addressed when developing new approaches to the design of complex systems:

- The overall design flows for heterogeneous systems and the associated use of models across traditional boundaries are not well developed and understood. Relationships between different teams inside a same company, or between different stake-holders in the supplier chain, are not supported by precise mathematical specifications of the components each party is expected to deliver.

- System requirements capture and analysis is in large part a heuristic process, where informal text and natural language-based techniques in use today are facing significant challenges [76]. Formal requirements engineering is in its infancy: mathematical models, formal analysis techniques and links to system implementation must be developed.

- Dealing with variability, uncertainty, and life-cycle issues, such as extensibility of a product family, are not well-addressed using available systems engineering methodologies and tools.

The challenge is to address the entire process and not to consider only local solutions of methodology, tools, and models that ease part of the design.

Contract-based design has been proposed as a new approach to the system design problem that is rigorous and effective in dealing with the problems and challenges described before, and that, at the same time, does not require a radical change in the way industrial designers carry out their task as it cuts across design flows of different types. Indeed, contracts can be used almost everywhere and at nearly all stages of system design, from early requirements capture, to embedded computing infrastructure and detailed design involving circuits and other hardware. Intuitively, a contract captures two properties, respectively representing the assumptions on the environment and the guarantees of the system under these assumptions. Hence, a contract can be defined as a pair $C = (A, G)$ of assumptions and guarantees characterizing in a formal way 1) under which context the design is assumed to operate, and 2) what its obligations are. Assume/Guarantee reasoning has been known for a long time, and has been used mostly in software engineering [86]. However, contract-based design is not limited to types and values in a piece of software. It can also be used to capture its performances (time, memory consumption, energy) and reliability. This amounts to enrich a component's interface with, on one hand, formal specifications of the behavior of the environment in which the component may be instantiated and, on the other hand, of the expected behavior of the component itself. To leverage contract-based reasoning as a technique of choice for system engineers, we aim to develop:

- mathematical foundations of contracts, that enable the design of formal verification frameworks;
- System engineering methodologies and tools, that focus on requirements modeling, contract specification and verification, at multiple abstraction levels.

A detailed bibliography on contract and interface theories for embedded system design can be found in [5]. In a nutshell, contract and interface theories fall into two main categories:

Assume/guarantee contracts. By explicitly relying on the notions of assumptions and guarantees, A/G-contracts are intuitive. This makes them appealing for the engineer. In A/G-contracts, assumptions and guarantees are just properties regarding the behavior of a component and of its environment. The typical case is when these properties are formal languages or sets of traces. This includes the class of safety properties [77, 49, 84, 17, 51]. Contract theories were initially developed as specification formalisms able to refuse some inputs from the environment [59]. A/G-contracts were advocated in [27] and are is still a very active research topic, with several contributions dealing with the timed [31] and probabilistic [43, 44] viewpoints in system design, and even hybrid systems design [88].

Automata theoretic interfaces. Interfaces combine assumptions and guarantees in a single, automata theoretic specification. Most interface theories are based on Lynch's Input/Output Automata [83, 82]. Interface Automata [13, 12, 14, 47] focus primarily on parallel composition and compatibility: two interfaces are compatible if there exists at least one environment where they can work together. The idea is that the resulting composition exposes as an interface the needed information to ensure that incompatible pairs of states cannot be reached. This can be achieved by using the possibility, for an Interface Automaton, to refuse some inputs from the environment in a given state. This amounts to the implicit assumption that the environment will never produce any of the refused inputs, when the interface is in this state. Modal Interfaces [91] inherit from both Interface Automata and the originally unrelated notion of Modal Transition System [79, 16, 35, 78]. Modal Interfaces are strictly more expressive than Interface Automata by decoupling the I/O orientation of an event and its deontic modalities (mandatory, allowed or forbidden). Informally, a *must* transition is offered in every component that realizes the modal interface, while a *may* transition is optional. Research on interface theories is still very active. For instance, timed [15, 28, 30, 56, 55, 29], probabilistic [43, 57] and energy-aware [48] interface theories have been proposed recently.

Requirements Engineering is one of the major concerns in large systems industries today, particularly so in sectors where certification prevails [94]. Most requirements engineering tools offer a poor structuring of the requirements and cannot be considered as formal modeling frameworks today. They are nothing less, but nothing more than an informal structured documentation enriched with hyperlinks.

We see Contract-Based Design and Interfaces Theories as innovative tools in support of Requirements Engineering. The Software Engineering community has extensively covered several aspects of Requirements Engineering, in particular:

- the development and use of large and rich *ontologies*; and
- the use of Model Driven Engineering technology for the structural aspects of requirements and resulting hyperlinks (to tests, documentation, PLM, architecture, and so on).

Behavioral models and properties, however, are not properly encompassed by the above approaches. This is the cause of a remaining gap between this phase of systems design and later phases where formal model based methods involving behavior have become prevalent. We believe that our work on contract-based design and interface theories is best suited to bridge this gap.

3.5 Efficient Symbolic Computation for Sparse Systems

This project consists in exploiting the parsimony of sparse systems to accelerate their symbolic manipulation (quantifiers elimination [52], differential-algebraic reductions [95] etc.). Let us cite two typical examples as a motivation: Boolean functions ($a \vee b \wedge \neg c$) and polynomial systems with inequalities ($x^2 + y \leq 1 \wedge x + y = 0$). We seek precisely to decompose these systems, automatically, in order to be able to manipulate them at an advantageous computational cost (in time and in memory) by attacking the pieces thus obtained rather than considering the system as a single monolithic block.

The current algorithms suffer from a theoretical complexity that is at best exponential (in the size of the input) limiting their use to instances of very modest size. The classic approach to overcome this problem is to develop/use numerical methods (with their limits and intrinsic problems) when possible of course. We aim to explore a different avenue.

In this project, we wish to exploit the structure of sparse systems to push the symbolic approach beyond its theoretical limits. The a priori limited application of our methods for dense systems is compensated by the fact that in practice, the problems are very often structured (in this regard, let us content ourselves with quoting the SAT solvers which successfully tackle industrial instances of a theoretically NP-complete problem).

The idea of exploiting the structure to speed up calculations that are a priori complex is not new. It has notably been developed and successfully used in signal processing via Factor Graphs [81], where one restricts oneself to local propagation of information, guided by an abstract graph which represents the structure of the system overall. Our approach is similar: we basically seek to use expensive algorithms sparingly on only subsystems involving only a small number of variables, thus hoping to reduce the theoretical worst case. One could then legitimately wonder why it is not enough to apply what has already been done on Factor Graphs? The difficulty (and the novelty for that matter) lies in the implementation of this idea for the problems that interest us. Let's start by emphasizing that the propagation of information has a significantly different impact depending on the operator (or quantifier) to be eliminated: a minimization or a summation do not look like a projection at all! This will obviously not prevent us from importing good ideas applicable to our problems and vice versa.

More related to symbolic computation, to our knowledge, at least two recent attempts exist: chordal networks [50] which propose a representation of the ideals of the ring of polynomials (therefore algebraic sets), and triangular block shapes [97], initiated independently and under development in our team and which tackle Boolean functions, or, if you will, the algebraic sets over the field of Booleans. The similarity between the two approaches is striking and suggests that there is a common way of doing things that could be exploited beyond these two examples. It is this unification that interests us in the first place in this project.

We identify three research problems to explore: **T1.** Unify several optimization problems on graphs as a single problem parameterized by a cost function, we coin such a problem WAP, for weighted adjacency propagation. **T2.** Adapt (and possibly improve) the algorithm of [96] to WAP and consequently to all instances of the single problem detailed in T1. **T3.** Propose a unified and modular method consisting of: (1) an elimination algorithm, (2) a data structure and (3) an efficient algorithm to solve the problem (with an adequate cost function).

The work on chordal networks and our work on Boolean functions immediately become special cases. For example, for Boolean functions, one could use Binary Decision Diagrams (BDDs) [40] to represent each piece of the initial system. In fact, the final representation will no longer be a single monolithic BDD as is currently the case, but rather a graph of BDDs. In the same way, an algebraic set will be represented by a graph where each node is a Gröbner basis (or any other data structure used to represent systems of equations).

The structure of the system becomes thus apparent and is exploited to optimize the used representation, opening the way to a better understanding and therefore to a more efficient and better targeted manipulation. Let's remember a simple fact here: symbolic manipulation often solves the problem exactly (without approximation or compromise). Therefore, pushing the limits of applicability of these techniques to scale them can only be appreciated and will undoubtedly have a significant impact on all the areas where they apply and the list is as long as it is varied. (compilation, certification, validation, synthesis, etc.).

4 Application domains

The Hycomes team contributes to the design of mathematical modeling languages and tools, to be used for the design of cyberphysical systems. In a nutshell, two major applications can be clearly identified: (i) our work on the structural analysis of multimode DAE systems has a sizeable impact on the techniques to be used in Modelica tools; (ii) our work on the verification of dynamical systems has an impact on the design methodology for safety-critical cyberphysical systems. These two applications are detailed below.

4.1 Modelica

Mathematical modeling tools are a considerable business, with major actors such as MathWorks, with Matlab/Simulink, or Wolfram, with Mathematica. However, none of these prominent tools are suitable for the engineering of large systems. The Modelica language has been designed with this objective in mind, making the best of the advantages of DAEs to support a component-based approach. Several industries in the energy sector have adopted Modelica as their main systems engineering language.

Although multimode features have been introduced in version 3.3 of the language [61], proper tool support of multimode models is still lagging behind. The reason is not a lack of interest from tool vendors and academia, but rather that multimode DAE systems poses several fundamental difficulties, such as a proper definition of a concept of solutions for multimode DAEs, how to handle mode switchings that trigger a change of system structure, or how impulsive variables should be handled. Our work on multimode DAEs focuses on these crucial issues [25].

Thanks to our IsamDAE software [42, 41], a larger class of Modelica models are expected to be compiled and simulated correctly. This should enable industrial users to have cleaner and simpler multimode Modelica models, with dynamically changing structure of cyberphysical systems. On the longer term, our ambition is to provide efficient code-generation techniques for the Modelica language, supporting, in full generality, multimode DAE systems, with dynamically changing differentiation index, structure and dimension.

The Hycomes team also focuses on scalability problems related to the compilation and simulation of large Modelica models. Digital twins developed by industrial Modelica users in the energy sector tend to be extremely large models, with up to 10^6 equations. State-of-the-art Modelica compilers can not handle such models and users are forced to partition their model into smaller parts and use complex co-simulation techniques to produce executable digital twins. This puts a heavy burden on digital twin developers, since both the partitioning and the implementation of cosimulation methods are manual, finely tailored to the model, and require a high degree of expertise.

The Hycomes team is working on a new generation of algorithms for the compilation of the Modelica language, that can scale up to large models. The key contributations are modular index-reduction [9] and block-triangular equation sorting algorithms, that can be applied to incomplete (rectangular) DAE systems.

4.2 Dynamical Systems Verification

In addition to well-defined operational semantics for hybrid systems, one often needs to provide formal guarantees about the behavior of some critical components of the system, or at least its main underlying logic. To do so, we are actively developing new techniques to automatically verify whether a hybrid system complies with its specifications, and/or to infer automatically the envelope within which the system behaves safely. The approaches we developed have been already successfully used to formally verify the intricate logic of the ACAS X, a mid-air collision avoidance system that advises the pilot to go upward or downward to avoid a nearby airplane which requires mixing the continuous motion of the aircraft with the discrete decisions to resolve the potential conflict [74]. This challenging example is nothing but an instance of the kind of systems we are targeting: autonomous smart systems that are designed to perform sophisticated tasks with an internal tricky logic. What is even more interesting perhaps is that such techniques can be often "reverted" to actually synthesize missing components so that some property holds, effectively helping the design of such complex systems.

5 Social and environmental responsibility

The expected impact of our research is to allow both better designs and better exploitation of energy production units and distribution networks, enabling large-scale energy savings. At least, this is what we could observe in the context of the FUI ModeliScale collaborative project (2018–2021), focused on electric grids, urban heat networks and building thermal modeling.

The rationale is as follows: system engineering models are meant to assess the correctness, safety and optimality of a system under design. However, system models are still useful after the system has been put in operation. This is especially true in the energy sector, where systems have an extremely long lifespan (for instance, more than 50 years for some nuclear power plants) and are upgraded periodically, to integrate new technologies. Exactly like in software engineering, where a software and its model co-evolve throughout the lifespan of the software, a co-evolution of the system and its physical models has to be maintained. This is required in order to maintain the safety of the system, but also its optimality.

Moreover, physical models can be instrumental to the optimal exploitation of a system. A typical example are model-predictive control (MPC) techniques, where the model is simulated, during the exploitation of the system, in order to predict system trajectories up to a bounded-time horizon. Optimal control inputs can then be computed by mathematical programming methods, possibly using multiple simulation results. This has been proved to be a practical solution [64], whenever classical optimal control methods are ineffective, for instance, when the system is non-linear or discontinuous. However, this requires the generation of high-performance simulation code, capable of simulating a system much faster than real-time.

The structural analysis techniques implemented in IsamDAE [42] generate a conditional block dependency graph, that can be used to generate high-performance simulation code : static code can be generated for each block of equations, and a scheduling of these blocks can be computed, at runtime, at each mode switching, thanks to an inexpensive topological sort algorithm. Contrarily to other approaches (such as [63]), no structural analysis, block-triangular decompositions, or automatic differentiation has to be performed at runtime.

6 Highlights of the year

The most notable result of the Hycomes team, for 2023, is the design and implementation of a modular structural analysis algorithm for multimode DAE systems, that can scale up to systems with the order 10^{12} equations. This important breakthrough has been presented at the Modelica'23 conference [9]. It lifts in effect the bottleneck of the structural analysis in the workflow of Modelica compilers.

7 New software, platforms, open data

7.1 New software

7.1.1 IsamDAE

Name: Implicit Structural Analysis of Multimode DAE systems

Keywords: Structural analysis, Differential algebraic equations, Multimode, Scheduling, Consistent initialization, Code generation

Scientific Description: Modeling languages and tools based on Differential Algebraic Equations (DAE) bring several specific issues that do not exist with modeling languages based on Ordinary Differential Equations. The main problem is the determination of the differentiation index and latent equations. Prior to generating simulation code and calling solvers, the compilation of a model requires a structural analysis step, which reduces the differentiation index to a level acceptable by numerical solvers.

The Modelica language, among others, allows hybrid models with multiple modes, mode-dependent dynamics and state-dependent mode switching. These Multimode DAE (mDAE) systems are much harder to deal with. The main difficulties are (i) the combinatorial explosion of the number of modes, and (ii) the correct handling of mode switchings.

The IsamDAE software aims at providing a compilation chain for mDAE-based modeling languages that make it possible to efficiently generate correct simulation code for multimode models. Novel structural analysis methods for mDAE systems were designed and implemented, based on an implicit representation of the varying structure of such systems. Several standard algorithms, such as J. Pryce's Sigma-method and the Dulmage-Mendelsohn decomposition, were adapted to the multimode case, using Binary Decision Diagrams (BDD) to represent the mode-dependent structure of an mDAE system.

IsamDAE determines, as a function of the mode, the set of latent equations, the leading variables and the state vector. This is then used to compute a conditional dependency graph (CDG) of the system, that can be used to generate simulation code with a mode-dependent scheduling of the blocks of equations. The software is also fit for generating simulation code for the hybrid dynamical system simulation tool Siconos, as well as handling the structural analysis of the multimode consistent initialization problem associated with an mDAE system.

Functional Description: IsamDAE (Implicit Structural Analysis of Multimode DAE systems) is a software library implementing new structural analysis methods for multimode DAE systems, based on an implicit representation of incidence graphs, matchings between equations and variables, and block decompositions. The input of the software is a variable dimension multimode DAE system consisting in a set of guarded equations and guarded variable declarations. It computes a mode-dependent structural index reduction of the multimode system and is able to produce a mode-dependent graph for the scheduling of blocks of equations in long modes, check the structural nonsingularity of the associated consistent initialization problem, or generate simulation code for the nonsmooth dynamical system simulation tool Siconos.

IsamDAE is coded in OCaml, and uses the following packages: GuaCaml by Joan Thibault, ML-BDD by Arlen Cox, Menhir by François Pottier and Yann Régis-Gianas, Pprint by François Pottier, Snowflake by Joan Thibault, XML-Light by Nicolas Cannasse and Jacques Garrigue.

Release Contributions: New features:

* XML representations of the structure of a multimode DAE model are accepted as inputs by the IsamDAE tool, in order to enable weak coupling with tools based on existing DAE-based languages. IsamDAE distinguishes between MEL and XML inputs based on the extension of the input file (.mel versus .mdae.xml).

Bug fixes:

* A better handling of the model structure for consistent initialization prevents subtle bugs that were observed for a few models and initial events. Specific error messages are returned when initial equations involve variables that are not active in the corresponding modes.

Performance improvement:

* Better handling of sets of equations/variables labeled with propositional formulas, thanks to an adapted data structure.

Various:

* Verbosity option -v now takes as a parameter an integer ranging from 0 ("quiet") to 5 ("deep debug"). The detailed output of CoSTreD is only available in "deep debug" mode.

URL: https://team.inria.fr/hycomes/software/isamdae/

Publications: hal-03768331, hal-02572879, hal-03320499, hal-02476541

Contact: Benoit Caillaud

Participants: Benoit Caillaud, Mathias Malandain, Joan Thibault, Alexandre Rocca, Bertrand Provot

7.1.2 Snowflake

Name: Snowflake : A Generic Symbolic Dynamic Programming framework

Keywords: Ocaml, Symbolic computation, Binary decision diagram

Scientific Description: Complex systems (either physical or logical) are structured and sparse, that is, they are build from individual components linked together, and any component is only linked to rather small number of other components with respects to the size of the global system.

RBTF exploits this structure, by over-approximating the relations between components as a tree (called decomposition tree in the graph literature) each node of this tree being a set of components of the initial systems. Then, starting from leaves, each sub-system is solved and the solutions are projected as a new constraints on their parents node, this process is iterated until all sub-systems are solved. This step allows to condensate all constraints and check their satisfiability. We call this step the **Forward Reduction Process** (FRP).

Finally, we can propagate all the constraints back into their initial sub-system by performing those same projections in the reverse direction. That is, each sub-system updates its set of solutions given the information from its parent then sends the information to its children sub-systems (possibly none, if its a leaf). We call this step the **Backward Propagation Process** (BPP).

Functional Description: Snowflake interfaces a WAP-solver (Weighted Adjacency Propagation problem), a functor-based implementation of CoSTreD (Constraint System Tree Decomposition), along with a minimalist MLBDD (Arlen Cox's BDD package) toolbox.

Release Contributions: 2022/07 : published Research Report 9478 (https://hal.archives-ouvertes.fr/hal-03740562/) 2022/06/30 : renamed RBTF into CoSTreD 2022/06/19 : added basic constraint system export 2022/06/02 : add small graphviz interface 2022/06/02 : added small graphviz interface 2022/06/02 : added sorted test on input to MlbddUtils.subst

URL: https://gitlab.com/boreal-ldd/snowflake/-/wikis/home

Author: Joan Thibault

Contact: Joan Thibault

7.1.3 modularSigma

Name: A modular Sigma-method for the structural analysis of large DAE systems

Keywords: Differential algebraic equations, Modularity

Scientific Description: A key feature of the Modelica language is its object-oriented nature: components are instances of classes and they can aggregate other components, so that extremely large models can be efficiently designed as "trees of components". However, the structural analysis of Modelica models, a necessary step for generating simulation code, often relies on the flattening of this hierarchical structure, which undermines the scalability of the language and results in widely-used Modelica tools not being able to compile and simulate such large models. This software implements a new algorithm for the modular structural analysis of Modelica models. An adaptation of Pryce's Sigma-method for non-square DAE systems, along with a carefully crafted notion of component interface, make it possible to fully exploit the object tree structure of a model. The structural analysis of a component class can be performed once and for all, only requiring the information provided by the interface of its child components. The resulting method alleviates the exponential computation costs that can be yielded by model flattening, hence, its scalability makes it ideally suited for the modeling and simulation of large cyber-physical systems.

Algorithms implemented in modularSigma are based on the Sigma-method, which reduces the DAE structural index-reduction problem to two complementary linear programs: the primal problem amounts to the computation of a maximal-weight perfect matching of the equation-variable incidence graph of the DAE, while the dual problem consists in the computation of the minimal solution of a difference bound matrix (DBM). Modularity is achieved thanks to a decomposition of both problems, using dynamic programming principles (akin to message passing techniques, that are often used in statistical estimation) and memoization of the intermediate results.

Functional Description: The software performs the index reduction and the bloc-triangular decomposition of large DAE systems, defined as the composition, hiding and renaming of incomplete (rectangular) DAE systems.

News of the Year: The initial purpose of modularSigma has been to benchmark the algorithms detailed in the paper presented at the Modelica'23 conference.

Publication: hal-04295096

Contact: Benoit Caillaud

Participant: Benoit Caillaud

7.1.4 PosInvSet

Name: Positive Invariant Sets

Keywords: Symbolic computation, Semi-algebraic set, Differential equations

Functional Description: Given a semi-algebraic set S, that is a Boolean combination of equations and inequalities of polynomials, and a polynomial differential equation, we show that an algorithm can effectively decide whether S is a positive invariant set for the considered dynamic, that is, if the initial condition is in S, then the entire trajectory defined by the dynamics belongs to S.

We implemented in Mathematica two different procedures. Both require a backend algorithm for real quantifiers elimination (like the Cylindrical Algebraic Decomposition). One procedure form a monolithic request for the entire problem. The other chop the problem into small pieces following the Boolean structure of the input S.

Release Contributions: Adaptation of the generic procedures to the linear case for scalability. The linear case means linear differential equations and semi-linear sets for the set S.

Contact: Khalil Ghorbal

8 New results

8.1 A Modular Structural Analysis of DAE Systems

Participants: Albert Benveniste, Benoît Caillaud, Mathias Malandain, Joan Thibault.

System modeling tools are key to the engineering of safe and efficient Cyber-Physical Systems (CPS). Although ODE-based languages and tools, such as Simulink [85], are widely used in industry, there are two main reasons why DAE-based modeling is best suited to the modeling of such systems: it enables a modeling based on first principles of the physics; it is physics-agnostic, and consequently accomodates arbitrary combinations of physics (mechanics, electrokinetics, hydraulics, thermodynamics, chemical reactions, etc.).

The pioneering work by Hilding Elmqvist [62] led to the emergence of the Modelica community in the 1990s, and the DAE-based modeling language of the same name [87] has become a *de facto* standard, with its object-oriented nature enabling a component-based modeling style. Its combined use with the port-Hamiltonian paradigm [92] results in a methodology that is instrumental to the scalable modeling of large systems, additionally ensuring that the model architecture preserves the system architecture, in stark contrast to ODE-based modeling [22, 23].

Consequently, DAE-based modeling requires that Modelica tools properly scale up to very large models. However, although Modelica enables the modeling of extremely large systems, its implementations [54, 66] are often not capable of compiling and simulating such large models. Scaling has been and still is a subject for sustained effort by the Modelica community [46], and although HPC issues belong to the landscape [36], a more specific issue is of uttermost importance for the Modelica language.

In the first steps of the compilation of a Modelica model, its hierarchical structure is flattened, thanks to a recursive syntactic inlining of the objects composing it. See [87], Section 5.6 for a complete definition of this flattening process. The result is an unstructured DAE that can be exponentially larger than the source model. The *structural analyses* that are required for the generation of simulation code (namely, the *index reduction* of the DAE system, followed by a *block-triangular form* transformation of the reduced-index system) are then performed on this monolithic DAE model. As the compilation process does not fully take advantage of the hierarchical nature of the models it has to handle, the modeling capabilities offered by the Modelica language are undermined by performance issues on the structural analysis itself [70, 69]. Additionally, model flattening poses a challenge when attempting to extend DAE-based modeling to higher-order modeling or dynamically changing systems [39, 38, 37].

In [9], a new modular structural analysis algorithm is proposed that takes full advantage of the object tree structure of a DAE model. The bedrock of this method is a novel concept of *structural analysis-aware interface* for components. The essence of a component interface is to capture the necessary information about a Modelica class that needs to be exposed, in order to perform the structural analysis of a component comprizing instances of the former class, while hiding away useless information regarding the equations and all *protected* features it may contain.

In order to compute a component interface, one has to be able to perform the structural analysis of the possibly non-square DAE system that this component encapsulates, and to use the interfaces of the components it aggregates in this analysis. We base our algorithm on Pryce's Σ-method for index reduction [90], which essentially consists in the successive solving of two dual linear integer programs. The striking difference with Pryce's algorithm is that these problems are solved by parts, in a scalable manner.

Putting all of this together, it is then possible to perform a *modular structural analysis*, in which structural analysis is performed at the class level, and the results can then be instantiated for each component of the system model, knowing its context. Hence, structural information at the system level is derived from composing the result of component-level analysis. Modular structural analysis yields huge gains in terms of memory usage and computational costs, as the analysis of a single large-scale DAE is replaced with that of multiple smaller subsystems. Moreover, the analysis is performed at the class level, meaning that a single structural analysis is needed for all system components that are instances of the same class.

8.2 Fault Diagnosability Analysis of Multi-Mode Systems

Participants: Benoît Caillaud, Mathias Malandain.

A new collaboration between the Hycomes and the University of Linköping (Sweden) has started this year on the topic of system diagnosis, based on multimode DAE systems.

Fault detection and diagnosis are important for the health monitoring of physical systems. Model-based approaches for single-mode, smooth, systems is a well-established field, supported by a large body of literature covering various approaches like structural methods [32], parity space techniques, and observer-based methods [72].

While single-mode systems are often described using differential algebraic equations (DAEs), the modeling of non-smooth physical systems yields switched DAEs, also known as multimode DAEs (mDAEs), which combine continuous behaviors, defined as solutions of a set of DAE systems, with discrete mode changes [98, 25]. Direct application of traditional fault diagnosis methods to all possible configurations of multi-mode systems quickly becomes intractable, as the number of modes tends to be exponential in the size of the system. The method proposed by [75] works around this issue by coupling a mode estimation algorithm with a single-mode diagnosis methodology, akin to just-in-time compilation in computer science. This approach unfortunately puts the burden on solving mode estimation problems, which often turn out to be intractable for the same reason.

Structural fault detectability and isolability is a graph-based method to evaluate diagnosability properties on DAEs [65]. It is based on the Dulmage-Mendelsohn decomposition (DM), a building block of the structural analysis of equation systems. In [11], we show how its extension to multimode systems, introduced in [4], can be applied in the context of structural fault detectability and isolability of mm-DAEs [68]. Building upon our previous research studies, the methods presented in this paper represent advancements in diagnostic methodologies for multi-mode systems, providing novel ways to study the diagnosability of multi-mode systems without enumerating their modes.

The case study used throughout this article is a model of a reconfigurable battery system, in which switching strategies enable to produce an AC output without relying on a central inverter [18]. This model is parametrized by the number of battery cells, so that both the inherent complexity associated with the diagnostics of such systems and the scalability of our approaches can be addressed.

8.3 Mixed Nondeterministic-Probabilistic Automata

Participants: Albert Benveniste.

Graphical models in probability and statistics are a core concept in the area of probabilistic reasoning and probabilistic programming-graphical models include Bayesian networks and factor graphs. For modeling and formal verification of probabilistic systems, probabilistic automata were introduced. A coherent suite of models consisting of Mixed Systems, Mixed Bayesian Networks, and Mixed Automata is proposed in [8]. This framework extends factor graphs, Bayesian networks, and probabilistic automata with the handling of nondeterminism. Each of these models comes with a parallel composition, and we establish clear relations between these three models. Also, we provide a detailed comparison between Mixed Automata and Probabilistic Automata.

8.4 On Continuous Solutions for Linear Complementarity Systems

Participants: Khalil Ghorbal.

Hybrid systems are dynamical systems alternating between continuous-time dynamics, called *modes*, and nonsmooth transitions between modes. Linear complementarity systems (LCS) form a special class of hybrid systems with an exponential number of modes and a linear differential algebraic equation in each mode. LCS are for instance used to describe mechanical and electrical systems featuring perfect contacts or ideal switches. For example, the ideal (Zener) diode is a 1-dimensional LCS with two modes: a passing mode in one direction and a blocking mode in the other direction. While seemingly simple, little is known about the existence, and eventually uniqueness, of continuous solutions (in the state space). The only known sufficient condition is too strong as it requires the existence and uniqueness of solutions for the underlying linear complementarity problem (LCP) which, for a fixed matrix M and a given vector q, asks whether there exists a pair of vectors (w, z) satisfying $w - Mz = q$, $w, z \geq 0$, and $w.z = 0$. M is said to be a *Q-matrix* when a solution exists for all q. It's worth noting that characterizing Q-matrices is an open problem since the sixties even for low dimensions. Motivated by generalizing the known sufficient conditions for the existence of continuous solutions for LCS, we were naturally led to better understand Q-matrices. In this work, we focused on the regions where no solution for a given LCP exists. We showed that such *holes* occur only in specific locations. We then exploited this property to fully characterize Q-matrices for $n \leq 3$.

Characterizing Q-matrices for any finite dimension n is still an open problem despite a large palette of attempts ranging from linear algebra to convex analysis all the way to the homology of simplicial sets. The novelty of our approach [10] relies on using geometric and topological intuitions to locate the regions for which the LCP *doesn't have a solution.* This property allowed us to reduce the spatial case to finite planar problems that we were able to enumerate and solve. Our characterization is a program enumerating a long list of (symbolic) constraints on the entries of the matrix M. The matrix is a Q-matrix if and only if all the constraints are satisfied. Such approach is for instance useful to generate examples (or counter-examples) to either solve existing conjectures or to improve our current understanding of the problem. For instance, we were able to find an example of a non-regular Q-matrix in dimension 3 (the smallest dimension for which such an example was known was $n = 5$). This is a joint work with Christelle Kozaily.

9 Dissemination

9.1 Promoting scientific activities

9.1.1 Scientific events: selection

Member of the conference program committees Albert Benveniste has served on the program committee of the Modelica'23 conference, that was held in October 2023, in Aachen (Germany).

Khalil Ghorbal has served program committee of the TACAS'23 conference.

9.1.2 Journal

Member of the editorial boards Benoît Caillaud has been appointed member of the editorial board of *Research Directions: Cyber-Physical Systems*, a new open-access journal published by Cambridge University Press. He is also serving on the board of the *MDPI Computation* journal.

Reviewer - reviewing activities Benoît Caillaud has reviewed papers for the *Discrete Event Dynamic Systems* and the *IEEE Transactions on Automatic Control* journals.

9.1.3 Scientific expertise

Albert Benveniste is member of the French National Academy of Technology. He also serves of the Scientific Advisory Board of the aeronautic company Safran.

Benoît Caillaud has evaluated proposals submitted for funding to the ANR (the French national research funding agency).

9.1.4 Research administration

Khalil Ghorbal is the main organizer of 68NQRT, the seminar on formal methods, programming languages and software engineering of the Inria center of the University of Rennes and of the Language and Software Engineering department of the Irisa UMR.

9.2 Teaching - Supervision - Juries

9.2.1 Teaching

- Master degree in computer science: Khalil Ghorbal, Category Theory, Monads, and Computation, at ENS Rennes, France ;
- *Agregation informatique*: Khalil Ghorbal and Maxime Bridoux, oral examination and lecture preparation, at ENS Rennes, France.

9.2.2 Supervision

- Benoît Caillaud and Khalil Ghorbal are cosupervising the PhD work of Joan Thibault. Joan Thibault is expected to defend is PhD thesis in 2024, on efficient and scalable data-structures for solving Boolean constraint systems and some optimization problems on them.
- Khalil Ghorbal is supervising the PhD work of Maxime Bridoux, on the broad topic of efficient symbolic computation methods for sparse algebraic systems.

10 Scientific production

10.1 Major publications

[1] A. Benveniste, T. Bourke, B. Caillaud, J.-L. Colaço, C. Pasteur and M. Pouzet. 'Building a Hybrid Systems Modeler on Synchronous Languages Principles'. In: *Proceedings of the IEEE*. Design Automation for Cyber-Physical Systems 106.9 (Sept. 2018), pp. 1568–1592. DOI: 10.1109/JPROC.2018.2858016. URL: https://hal.inria.fr/hal-01879026.

[2] A. Benveniste, T. Bourke, B. Caillaud and M. Pouzet. 'Non-standard semantics of hybrid systems modelers'. English. In: *Journal of Computer and System Sciences* 78.3 (2012). This work was supported by the SYNCHRONICS large scale initiative of INRIA, pp. 877–910. DOI: 10.1016/j.jcss.2011.08.009. URL: http://hal.inria.fr/hal-00766726.

[3] A. Benveniste, B. Caillaud and M. Malandain. 'The mathematical foundations of physical systems modeling languages'. In: *Annual Reviews in Control* 50 (2020), pp. 72–118. DOI: 10.1016/j.arcontrol.2020.08.001. URL: https://hal.inria.fr/hal-03045498.

[4] A. Benveniste, B. Caillaud, M. Malandain and J. Thibault. 'Algorithms for the Structural Analysis of Multimode Modelica Models'. In: *Electronics* 11.17 (1st Sept. 2022), pp. 1–63. DOI: 10.3390/electronics11172755. URL: https://inria.hal.science/hal-03768331.

[5] A. Benveniste, B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. Sangiovanni-Vincentelli, W. Damm, T. Henzinger and K. G. Larsen. 'Contracts for System Design'. In: *Foundations and Trends in Electronic Design Automation* 12.2-3 (2018), pp. 124–400. DOI: 10.1561/1000000053. URL: https://hal.inria.fr/hal-01971429.

[6] J.-B. Jeannin, K. Ghorbal, Y. Kouskoulas, A. Schmidt, R. Gardner, S. Mitsch and A. Platzer. 'A Formally Verified Hybrid System for Safe Advisories in the Next-Generation Airborne Collision Avoidance System'. In: *International Journal on Software Tools for Technology Transfer* 19.6 (Nov. 2017), pp. 717–741. DOI: 10.1007/s10009-016-0434-1. URL: https://hal.archives-ouvertes.fr/hal-01232365.

[7] A. Sogokon, K. Ghorbal and T. T. Johnson. 'Operational Models for Piecewise-Smooth Systems'. In: *ACM Transactions on Embedded Computing Systems (TECS)* 16.5s (Oct. 2017), 185:1–185:19. DOI: 10.1145/3126506. URL: https://hal.inria.fr/hal-01658196.

10.2 Publications of the year

International journals

[8] A. Benveniste and J.-B. Raclet. 'Mixed Nondeterministic-Probabilistic Automata: Blending graphical probabilistic models with nondeterminism'. In: *Discrete Event Dynamic Systems* 2023 (Oct. 2023), pp. 1–58. DOI: 10.1007/s10626-023-00375-x. URL: https://ut3-toulouseinp.hal.science/hal-04276789.

International peer-reviewed conferences

[9] A. Benveniste, B. Caillaud, M. Malandain and J. Thibault. 'Towards the separate compilation of Modelica: modularity and interfaces for the index reduction of incomplete DAE systems'. In: Modelica 2023 - 15th International Modelica Conference. Aachen, Germany, 2023, pp. 1–10. URL: https://inria.hal.science/hal-04295096.

Reports & preprints

[10] K. Ghorbal and C. Kozaily. *On Covering Euclidean Spaces with Q-arrangements of Cones*. 8th Feb. 2024. URL: https://inria.hal.science/hal-04444572.

[11] F. Hashemniya, B. Caillaud, E. Frisk, M. Krysander and M. Malandain. *Fault Diagnosability Analysis of Multi-Mode Systems*. 2023. DOI: 10.48550/arXiv.2312.14030. URL: https://inria.hal.science/hal-04361934.

10.3 Cited publications

[12] L. de Alfaro. 'Game Models for Open Systems'. In: *Verification: Theory and Practice*. Vol. 2772. Lecture Notes in Computer Science. Springer, 2003, pp. 269–289. DOI: 10.1007/978-3-540-39910-0_12.

[13] L. de Alfaro and T. A. Henzinger. 'Interface automata'. In: *Proc. of the 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE'01)*. ACM Press, 2001, pp. 109–120. DOI: 10.1145/503271.503226.

[14] L. de Alfaro and T. A. Henzinger. 'Interface-based design'. In: *In Engineering Theories of Software Intensive Systems, proceedings of the Marktoberdorf Summer School*. Kluwer, 2004. DOI: 10.1007/1-4020-3532-2_3.

[15] L. de Alfaro, T. A. Henzinger and M. Stoelinga. 'Timed Interfaces'. In: *Proc. of the 2nd International Workshop on Embedded Software (EMSOFT'02)*. Vol. 2491. Lecture Notes in Computer Science. Springer, 2002, pp. 108–122. DOI: 10.1007/3-540-45828-X_9.

[16] A. Antonik, M. Huth, K. G. Larsen, U. Nyman and A. Wasowski. '20 Years of Modal and Mixed Specifications'. In: *Bulletin of European Association of Theoretical Computer Science* 1.94 (2008). URL: https://dblp.org/rec/journals/eatcs/AntonikHLNW08.bib.

[17] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, Cambridge, 2008. URL: https://mitpress.mit.edu/9780262026499/principles-of-model-checking/.

[18] A. Balachandran, T. Jonsson and L. Eriksson. 'Design and Analysis of Battery-Integrated Modular Multilevel Converters for Automotive Powertrain Applications'. In: *2021 23rd European Conference on Power Electronics and Applications (EPE'21 ECCE Europe)*. IEEE. 2021, P–1.

[19] A. Benveniste, T. Bourke, B. Caillaud, J.-L. Colaço, C. Pasteur and M. Pouzet. 'Building a Hybrid Systems Modeler on Synchronous Languages Principles'. In: *Proceedings of the IEEE*. Design Automation for Cyber-Physical Systems 106.9 (Sept. 2018), pp. 1568–1592. DOI: 10.1109/JPROC.2018.2858016. URL: https://hal.inria.fr/hal-01879026.

[20] A. Benveniste, T. Bourke, B. Caillaud, B. Pagano and M. Pouzet. *A Type-Based Analysis of Causality Loops In Hybrid Systems Modelers*. Deliverable D3.1_1 v 1.0 of the Sys2soft collaborative project "Physics Aware Software". Dec. 2013. URL: https://hal.inria.fr/hal-00938866.

[21] A. Benveniste, T. Bourke, B. Caillaud and M. Pouzet. *Semantics of multi-mode DAE systems*. Deliverable D.4.1.1 of the ITEA2 Modrio collaborative project. Aug. 2013. URL: https://hal.inria.fr/hal-00938891.

[22] A. Benveniste, B. Caillaud, H. Elmqvist, K. Ghorbal, M. Otter and M. Pouzet. 'Multi-Mode DAE Models - Challenges, Theory and Implementation'. In: *Computing and Software Science: State of the Art and Perspectives*. Vol. 10000. Lecture Notes in Computer Science. Springer, Oct. 2019, pp. 283–310. DOI: 10.1007/978-3-319-91908-9_16. URL: https://hal.inria.fr/hal-02333603.

[23] A. Benveniste, B. Caillaud and M. Malandain. 'From Hybrid Automata to DAE-Based Modeling'. In: *Principles of Systems Design*. Vol. 13660. Lecture Notes in Computer Science. Springer Nature Switzerland, Dec. 2022, pp. 3–20. DOI: 10.1007/978-3-031-22337-2_1. URL: https://inria.hal.science/hal-03921708.

[24] A. Benveniste, B. Caillaud and M. Malandain. *Structural Analysis of Multimode DAE Systems: summary of results*. Research Report RR-9387. Inria Rennes – Bretagne Atlantique, Jan. 2021, p. 27. URL: https://hal.inria.fr/hal-03104030.

[25] A. Benveniste, B. Caillaud and M. Malandain. 'The mathematical foundations of physical systems modeling languages'. In: *Annual Reviews in Control* 50 (2020), pp. 72–118. DOI: 10.1016/j.arcontrol.2020.08.001. URL: https://hal.inria.fr/hal-03045498.

[26] A. Benveniste, B. Caillaud, B. Pagano and M. Pouzet. 'A type-based analysis of causality loops in hybrid modelers'. In: *HSCC '14: International Conference on Hybrid Systems: Computation and Control*. Proceedings of the 17th international conference on Hybrid systems: computation and control (HSCC '14). Berlin, Germany: ACM Press, Apr. 2014, p. 13. DOI: 10.1145/2562059.2562125. URL: https://hal.inria.fr/hal-01093388.

[27] A. Benveniste, B. Caillaud, A. Ferrari, L. Mangeruca, R. Passerone and C. Sofronis. 'Multiple Viewpoint Contract-Based Specification and Design'. In: *Proceedings of the Software Technology Concertation on Formal Methods for Components and Objects (FMCO'07)*. Vol. 5382. Revised Lectures, Lecture Notes in Computer Science. Amsterdam, The Netherlands: Springer, Oct. 2008. DOI: 10.1007/978-3-540-92188-2_9.

[28] N. Bertrand, A. Legay, S. Pinchinat and J.-B. Raclet. 'A Compositional Approach on Modal Specifications for Timed Systems.' In: *11th International Conference on Formal Engineering Methods (ICFEM'09)*. Vol. 5885. LNCS. Rio de Janeiro, Brazil: Springer, Dec. 2009, pp. 679–697. URL: https://hal.inria.fr/inria-00424356.

[29] N. Bertrand, A. Legay, S. Pinchinat and J.-B. Raclet. 'Modal event-clock specifications for timed component-based design'. In: *Science of Computer Programming* 77 (2012), pp. 1212–1234. DOI: 10.1016/j.scico.2011.01.007. URL: https://hal.inria.fr/hal-00752449.

[30] N. Bertrand, S. Pinchinat and J.-B. Raclet. 'Refinement and Consistency of Timed Modal Specifications.' In: *3rd International Conference on Language and Automata Theory and Applications (LATA'09)*. Vol. 5457. LNCS. Tarragona, Spain: Springer, Apr. 2009, pp. 152–163. DOI: 10.1007/978-3-642-00982-2_13. URL: https://hal.inria.fr/inria-00424283.

[31] P. Bhaduri and I. Stierand. 'A proposal for real-time interfaces in SPEEDS'. In: *Design, Automation and Test in Europe (DATE'10)*. IEEE, 2010, pp. 441–446. DOI: 10.1109/DATE.2010.5457163.

[32] M. Blanke, M. Kinnaert, J. Lunze and M. Staroswiecki. 'Diagnosis and Fault-Tolerant Control'. In: Springer Berlin, Heidelberg, Sept. 2006, pp. 109–188. DOI: https://doi.org/10.1007/978-3-540-35653-0.

[33] S. Bliudze. 'Un cadre formel pour l'étude des systèmes industriels complexes: un exemple basé sur l'infrastructure de l'UMTS'. PhD thesis. Ecole Polytechnique, 2006.

[34] S. Bliudze and D. Krob. 'Modelling of Complex Systems: Systems as Dataflow Machines'. In: *Fundamenta Informaticae* 91.2 (2009), pp. 251–274. DOI: 10.3233/FI-2009-0043. URL: https://hal.science/hal-02561099.

[35] G. Boudol and K. G. Larsen. 'Graphical versus logical specifications'. In: *Theoretical Computer Science* 106.1 (1992), pp. 3–20. DOI: https://doi.org/10.1016/0304-3975(92)90276-L. URL: https://www.sciencedirect.com/science/article/pii/030439759290276L.

[36] W. Braun, F. Casella and B. Bachmann. 'Solving large-scale Modelica models: new approaches and experimental results using OpenModelica'. In: *Proceedings of the 12th International Modelica Conference*. 2017.

[37] D. Broman. 'Interactive Programmatic Modeling'. In: *ACM Trans. Embed. Comput. Syst.* 20.4 (2021), 33:1–33:26. DOI: 10.1145/3431387.

[38] D. Broman. 'Meta-Languages and Semantics for Equation-Based Modeling and Simulation'. PhD thesis. Linköping University, Sweden, 2010. URL: https://nbn-resolving.org/urn:nbn:se:liu:diva-58743.

[39] D. Broman and P. Fritzson. 'Higher-Order Acausal Models'. In: *Proceedings of the 2nd International Workshop on Equation-Based Object-Oriented Languages and Tools, EOOLT 2008, Paphos, Cyprus, July 8, 2008*. Vol. 29. Linköping Electronic Conference Proceedings. Linköping University Electronic Press, 2008, pp. 59–69. DOI: 10.11128/sne.19.tn.09921.

[40] R. E. Bryant. 'Graph-Based Algorithms for Boolean Function Manipulation'. In: *IEEE Trans. Comput.* 35.8 (Aug. 1986), pp. 677–691. DOI: 10.1109/TC.1986.1676819. URL: http://dx.doi.org/10.1109/TC.1986.1676819.

[41] B. Caillaud, M. Malandain and J. Thibault. *Demo: IsamDAE, an Implicit Structural Analysis Tool for Multimode DAE Systems*. HSCC 2020 - 23rd ACM International Conference on Hybrid Systems: Computation and Control. Poster. Apr. 2020. URL: https://hal.inria.fr/hal-02545380.

[42] B. Caillaud, M. Malandain and J. Thibault. 'Implicit structural analysis of multimode DAE systems'. In: *HSCC 2020 - 23rd ACM International Conference on Hybrid Systems: Computation and Control*. Sydney New South Wales Australia, France: ACM, Apr. 2020, pp. 1–11. DOI: 10.1145/3365365.3382201. URL: https://hal.inria.fr/hal-02572879.

[43] B. Caillaud, B. Delahaye, K. G. Larsen, A. Legay, M. L. Pedersen and A. Wasowski. 'Compositional design methodology with constraint Markov chains'. In: *QEST 2010*. Williamsburg, Virginia, United States, Sept. 2010. DOI: 10.1109/QEST.2010.23. URL: http://hal.inria.fr/inria-00591578/en.

[44] B. Caillaud, B. Delahaye, K. G. Larsen, A. Legay, M. L. Pedersen and A. Wasowski. 'Constraint Markov Chains'. In: *Theoretical Computer Science* 412.34 (May 2011), pp. 4373–4404. DOI: 10.1016/j.tcs.2011.05.010. URL: http://hal.inria.fr/hal-00654003/en.

[45] S. L. Campbell and C. W. Gear. 'The index of general nonlinear DAEs'. In: *Numerische Mathematik* 72.2 (Dec. 1995), pp. 173–196. DOI: 10.1007/s002110050165. URL: http://dx.doi.org/10.1007/s002110050165.

[46] F. Casella and A. Guironnet. 'ScalableTestGrids - An Open-Source and Flexible Benchmark Suite to Assess Modelica Tool Performance on Large-Scale Power System Test Cases'. In: *Proceedings of the 14th International Modelica Conference*. Linköping Electronic Conference Proceedings 181. Linköping, Sweden: Modelica Association and Linköping University Electronic Press, Sept. 2021, pp. 351–358. DOI: 10.3384/ecp21181351.

[47] A. Chakrabarti. 'A Framework for Compositional Design and Analysis of Systems'. PhD thesis. EECS Department, University of California, Berkeley, Dec. 2007. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2007/EECS-2007-174.html.

[48] A. Chakrabarti, L. de Alfaro, T. A. Henzinger and M. Stoelinga. 'Resource Interfaces'. In: *Embedded Software, Third International Conference, EMSOFT 2003, Philadelphia, PA, USA, October 13-15, 2003, Proceedings*. Vol. 2855. Lecture Notes in Computer Science. Springer, 2003, pp. 117–133. DOI: 10.1007/978-3-540-45212-6_9.

[49] E. Y. Chang, Z. Manna and A. Pnueli. 'Characterization of Temporal Property Classes'. In: *ICALP*. Vol. 623. Lecture Notes in Computer Science. Springer, 1992, pp. 474–486. DOI: 10.1007/3-540-55719-9_97.

[50] D. Cifuentes and P. A. Parrilo. 'Chordal Networks of Polynomial Ideals'. In: *SIAM J. Appl. Algebra Geom.* 1.1 (2017), pp. 73–110. DOI: 10.1137/16M106995X. URL: https://doi.org/10.1137/16M106995X.

[51] E. Clarke, O. Grumberg and D. Peled. *Model Checking*. MIT Press, 1999. URL: https://mitpress.mit.edu/9780262038836/model-checking/.

[52] G. E. Collins and H. Hong. 'Partial Cylindrical Algebraic Decomposition for Quantifier Elimination'. In: *J. Symb. Comput.* 12.3 (1991), pp. 299–328. DOI: 10.1016/S0747-7171(08)80152-6.

[53] N. J. Cutland, ed. *Nonstandard analysis and its applications*. Cambridge Univ. Press, 1988. DOI: 10.1017/CBO9781139172110.

[54] Dassault Systèmes. *Dymola official webpage*. Accessed: 2023-06-12. 2023. URL: https://www.3ds.com/products-services/catia/products/dymola/.

[55] A. David, K. G. Larsen, A. Legay, U. Nyman and A. Wasowski. 'ECDAR: An Environment for Compositional Design and Analysis of Real Time Systems'. In: *Automated Technology for Verification and Analysis - 8th International Symposium, ATVA 2010, Singapore, September 21-24, 2010. Proceedings*. 2010, pp. 365–370. DOI: 10.1007/978-3-642-15643-4_29.

[56] A. David, K. G. Larsen, A. Legay, U. Nyman and A. Wasowski. 'Timed I/O automata: a complete specification theory for real-time systems'. In: *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2010, Stockholm, Sweden, April 12-15, 2010*. 2010, pp. 91–100. DOI: 10.1145/1755952.1755967.

[57] B. Delahaye, J.-P. Katoen, K. G. Larsen, A. Legay, M. L. Pedersen, F. Sher and A. Wasowski. 'Abstract Probabilistic Automata'. In: *Verification, Model Checking, and Abstract Interpretation - 12th International Conference, VMCAI 2011, Austin, TX, USA, January 23-25, 2011. Proceedings*. Vol. 6538. Lecture Notes in Computer Science. 2011, pp. 324–339. DOI: 10.1007/978-3-642-18275-4_23.

[58] F. Diener and G. Reeb. *Analyse non standard*. Hermann, 1989. URL: https://www.editions-hermann.fr/livre/analyse-non-standard-francine-diener.

[59] D. L. Dill. *Trace Theory for Automatic Hierarchical Verification of Speed-Independent Circuits*. ACM Distinguished Dissertations. MIT Press, 1989. DOI: 10.7551/mitpress/6874.001.0001.

[60] J. Edmonds and R. M. Karp. 'Theoretical improvements in algorithmic efficiency for network flow problems'. In: *Journal of the ACM* 19.2 (1972), pp. 248–264. DOI: 10.1145/321694.321699. URL: http://dx.doi.org/10.1145/321694.321699.

[61] H. Elmqvist, S. E. Mattsson and M. Otter. 'Modelica extensions for Multi-Mode DAE Systems'. In: *Proceedings of the 10th International Modelica Conference, March 10-12, 2014, Lund, Sweden*. Linköping University Electronic Press, Mar. 2014. DOI: 10.3384/ecp14096183.

[62] H. Elmqvist. 'A structured model language for large continuous systems'. PhD thesis. Universiteit i Lund, 1978.

[63] H. Elmqvist, A. Neumayr and M. Otter. 'Modia-dynamic modeling and simulation with julia'. In: *Juliacon'18*. University College London, UK, Aug. 2018. URL: https://elib.dlr.de/124133/.

[64] H. J. Ferreau, S. Almér, H. Peyrl, J. L. Jerez and A. Domahidi. 'Survey of industrial applications of embedded model predictive control'. In: *2016 European Control Conference (ECC)*. 2016, pp. 601–601. DOI: 10.1109/ECC.2016.7810351.

[65] E. Frisk, A. Bregon, J. Aslund, M. Krysander, B. Pulido and G. Biswas. 'Diagnosability analysis considering causal interpretations for differential constraints'. In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 42.5 (2012), pp. 1216–1229.

[66] P. Fritzson, A. Pop, K. Abdelhak, A. Ashgar, B. Bachmann, W. Braun, D. Bouskela, R. Braun, L. Buffoni, F. Casella, R. Castro, R. Franke, D. Fritzson, M. Gebremedhin, A. Heuermann, B. Lie, A. Mengist, L. Mikelsons, K. Moudgalya, L. Ochel, A. Palanisamy, V. Ruge, W. Schamai, M. Sjölund, B. Thiele, J. Tinnerholm and P. Östlund. 'The OpenModelica Integrated Environment for Modeling, Simulation, and Model-Based Development'. In: *Modeling, Identification and Control* 41.4 (2020), pp. 241–295. DOI: 10.4173/mic.2020.4.1.

[67] A. V. Goldberg and R. E. Tarjan. 'A new approach to the maximum flow problem'. In: *Proceedings of the eighteenth annual ACM symposium on Theory of computing (STOC'86)*. 1986. DOI: 10.1145/12130.12144. URL: http://dx.doi.org/10.1145/12130.12144.

[68] F. Hashemniya, E. Frisk and M. Krysander. 'Hierarchical Diagnosis Algorithm for Component-Based Multi-Mode Systems'. In: *22nd IFAC World Congress*. IFAC. 2023.

[69] C. Höger. 'Compiling Modelica: about the separate translation of models from Modelica to OCaml and its impact on variable-structure modeling'. PhD thesis. TU Berlin, 2019. DOI: 0.14279/depositonce-8354.

[70] C. Höger. 'Faster Structural Analysis of Differential-Algebraic Equations by Graph Compression'. In: *IFAC-PapersOnLine* 48.1 (2015). 8th Vienna International Conference on Mathematical Modelling, pp. 135–140. DOI: 10.1016/j.ifacol.2015.05.100. URL: https://www.sciencedirect.com/science/article/pii/S2405896315001019.

[71] *IEEE Standard VHDL Analog and Mixed-Signal Extensions, Std 1076.1-1999*. 1999. DOI: 10.1109/IEEESTD.1999.90578. URL: http://dx.doi.org/10.1109/IEEESTD.1999.90578.

[72] R. Isermann. *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Springer Berlin Heidelberg, 2006.

[73] Y. Iwasaki, A. Farquhar, V. Saraswat, D. Bobrow and V. Gupta. 'Modeling Time in Hybrid Systems: How Fast Is "Instantaneous"?' In: *IJCAI*. 1995, pp. 1773–1781. URL: https://www.ijcai.org/Proceedings/95-2/Papers/097.pdf.

[74] J.-B. Jeannin, K. Ghorbal, Y. Kouskoulas, R. Gardner, A. Schmidt, E. Zawadzki and A. Platzer. 'Formal verification of ACAS X, an industrial airborne collision avoidance system'. In: *2015 International Conference on Embedded Software, EMSOFT 2015, Amsterdam, Netherlands, October 4-9, 2015*. Ed. by A. Girault and N. Guan. Amsterdam, Netherlands: IEEE, 2015, pp. 127–136. DOI: 10.1109/EMSOFT.2015.7318268. URL: https://hal.science/hal-01660902.

[75] H. Khorasgani and G. Biswas. 'Structural fault detection and isolation in hybrid systems'. In: *IEEE Transactions on Automation Science and Engineering* 15.4 (2017), pp. 1585–1599.

[76] A. Lamercerie. ‘Principe de transduction sémantique pour l’application de théories d’interfaces sur des documents de spécification’. Theses. Université Rennes 1 ; Rennes 1, Apr. 2021. URL: https://theses.hal.science/tel-03366457.

[77] L. Lamport. ‘Proving the Correctness of Multiprocess Programs’. In: *IEEE Trans. Software Eng.* 3.2 (1977), pp. 125–143. DOI: 10.1109/TSE.1977.229904.

[78] K. G. Larsen, U. Nyman and A. Wasowski. ‘On Modal Refinement and Consistency’. In: *Proc. of the 18th International Conference on Concurrency Theory (CONCUR’07)*. Springer, 2007, pp. 105–119. DOI: 10.1007/978-3-540-74407-8_8.

[79] K. G. Larsen and B. Thomsen. ‘A Modal Process Logic’. In: *Proceedings of the Third Annual Symposium on Logic in Computer Science (LICS’88)*. IEEE, 1988, pp. 203–210. DOI: 10.1109/LICS.1988.5119.

[80] T. Lindstrøm. ‘An Invitation to Nonstandard Analysis’. In: *Nonstandard Analysis and its Applications*. Ed. by N. J. Cutland. Cambridge Univ. Press, 1988, pp. 1–105. DOI: 10.1017/CBO9781139172110.002.

[81] H.-A. Loeliger. ‘An introduction to factor graphs’. In: *IEEE Signal Processing Magazine* 21.1 (2004), pp. 28–41. DOI: 10.1109/MSP.2004.1267047.

[82] N. A. Lynch. ‘Input/Output Automata: Basic, Timed, Hybrid, Probabilistic and Dynamic’. In: *CONCUR 2003 - Concurrency Theory, 14th International Conference, Marseille, France, September 3-5, 2003, Proceedings*. Vol. 2761. Lecture Notes in Computer Science. Springer, 2003, pp. 187–188. DOI: 10.1007/978-3-540-45187-7_12.

[83] N. A. Lynch and E. W. Stark. ‘A Proof of the Kahn Principle for Input/Output Automata’. In: *Inf. Comput.* 82.1 (1989), pp. 81–92. DOI: 10.1016/0890-5401(89)90066-7.

[84] Z. Manna and A. Pnueli. *Temporal verification of reactive systems: Safety*. Springer, 1995. DOI: 10.1007/978-1-4612-4222-2.

[85] MathWorks, Inc. *Simulink official webpage*. Accessed: 2023-06-12. 2023. URL: https://www.mathworks.com/products/simulink.html.

[86] B. Meyer. ‘Applying “Design by Contract”’. In: *Computer* 25.10 (Oct. 1992), pp. 40–51. DOI: 10.1109/2.161279. URL: http://dx.doi.org/10.1109/2.161279.

[87] Modelica Association. *Modelica, A Unified Object-Oriented Language for Systems Modeling. Language Specification, Version 3.6*. Accessed: 2023-06-12. 2023. URL: https://modelica.org/documents/MLS.pdf.

[88] P. Nuzzo, A. L. Sangiovanni-Vincentelli, X. Sun and A. Puggelli. ‘Methodology for the Design of Analog Integrated Interfaces Using Contracts’. In: *IEEE Sensors Journal* 12.12 (Dec. 2012), pp. 3329–3345. DOI: 10.1109/JSEN.2012.2211098.

[89] C. Pantelides. ‘The consistent initialization of differential-algebraic systems’. In: *SIAM J. Sci. Stat. Comput.* 9.2 (1988), pp. 213–231. DOI: 10.1137/0909014.

[90] J. D. Pryce. ‘A Simple Structural Analysis Method for DAEs’. In: *BIT Numerical Mathematics* 41.2 (Mar. 2001), pp. 364–394. DOI: 10.1023/a:1021998624799. URL: http://dx.doi.org/10.1023/a:1021998624799.

[91] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legay and R. Passerone. ‘A Modal Interface Theory for Component-based Design’. In: *Fundamenta Informaticae* 108.1-2 (2011), pp. 119–149. DOI: 10.3233/FI-2011-416. URL: http://hal.inria.fr/inria-00554283/en.

[92] R. Rashad, F. Califano, A. van der Schaft and S. Stramigioli. ‘Twenty years of distributed port-Hamiltonian systems: a literature review’. In: *IMA J. Math. Control. Inf.* 37.4 (2020), pp. 1400–1422. DOI: 10.1093/imamci/dnaa018.

[93] A. Robinson. *Non-Standard Analysis*. Princeton Landmarks in Mathematics, 1996. URL: https://press.princeton.edu/books/paperback/9780691044903/non-standard-analysis.

[94] E. Sikora, B. Tenbergen and K. Pohl. ‘Industry needs and research directions in requirements engineering for embedded systems’. In: *Requirements Engineering* 17 (2012), pp. 57–78. DOI: 10.1007/s00766-011-0144-x. URL: http://link.springer.com/article/10.1007/s00766-011-0144-x.

[95] W. Y. Sit. ‘The Ritt–Kolchin theory for differential polynomials’. In: *Differential Algebra and Related Topics*. 2002, pp. 1–70. DOI: 10.1142/9789812778437_0001.

[96] H. Tamaki. ‘Positive-instance driven dynamic programming for treewidth’. In: *J. Comb. Optim.* 37.4 (2019), pp. 1283–1311. DOI: 10.1007/s10878-018-0353-z.

[97] J. Thibault and K. Ghorbal. ‘Leveraging Structural Analysis for Quantified Boolean Formulae’. In: *Summer School on Modelling and Verification of Parallel Processes, Grenoble, France* 6 (2020). http://khalilghorbal.info/assets/pdf/papers/RBTF_movep.pdf.

[98] S. Trenn. ‘Switched Differential Algebraic Equations’. In: *Dynamics and Control of Switched Electronic Systems: Advanced Perspectives for Modeling, Simulation and Control of Power Converters*. Ed. by F. Vasca and L. Iannelli. London: Springer London, 2012, pp. 189–216. DOI: 10.1007/978-1-4471-2885-4.