Activity Report 2017

Team AOSTE2

Models and methods of analysis and optimization for systems with real-time and embedded contraints

Inria teams are typically groups of researchers working on the definition of a common project, and objectives, with the goal to arrive at the creation of a project-team. Such project-teams may include other partners (universities or research institutions).

RESEARCH CENTER
Paris

THEME
Embedded and Real-time Systems

Table of contents

Team AOSTE2

Creation of the Team: 2017 January 01

Keywords:

Computer Science and Digital Science:

A1.3. - Distributed Systems
A1.5.2. - Communicating systems
A2.1.1. - Semantics of programming languages
A2.1.8. - Synchronous languages
A2.1.10. - Domain-specific languages
A2.2.3. - Run-time systems
A2.2.4. - Parallel architectures
A2.3. - Embedded and cyber-physical systems
A2.3.1. - Embedded systems
A2.3.2. - Cyber-physical systems
A2.3.3. - Real-time systems
A2.4.1. - Analysis
A2.4.3. - Proofs
A8.2. - Optimization

Other Research Topics and Application Domains:

B5.2. - Design and manufacturing
B5.2.1. - Road vehicles
B5.2.2. - Railway
B5.2.3. - Aviation
B5.2.4. - Aerospace
B6.6. - Embedded systems

1. Personnel

Research Scientists

Liliana Cucu [Inria, Researcher, HDR]
Robert Davis [University of York UK, Chair]
Dumitru Potop-Butucaru [Inria, Researcher, HDR]
Yves Sorel [Inria, Senior Researcher]

PhD Students

Slim Ben-Amor [Inria]
Keryan Didier [Inria]
Cristian Maxim [Airbus]
Salah-Eddine Saidi [IFPEN]
Evariste Ntaryamira [Inria, Embassy of France at Burundi]
Walid Talaboulma [Inria]

Technical staff

Irina-Mariuca Asavoae [Inria, until Aug 2017]
Mihail Asavoae [Inria, until Apr 2017]

Antoine Bertout [Inria, until Jul 2017]
Adriana Gogonel [Inria]
Fatma Jebali [Inria]
Tomasz Kloda [Inria, until Jul 2017]
Mehdi Mezouak [Inria]

Interns

Michail Papadimitriou [Huawei, from Feb 2017 until Jul 2017]
Anselme Revuz [Inria, from May 2017 until Aug 2017]

Administrative Assistant

Christine Anocq [Inria]

Visiting Scientist

George Lima [Inria, from May 2017 until Jun 2017]

External Collaborator

Laurent George [Univ Paris-Val de Marne, HDR]

2. Overall Objectives

2.1. Overall Objectives

The recent advances in merging different technologies and engineering domains has led to the emergence of Cyber-Physical Systems (CPS). In such systems, embedded computers interact with, and control physical processes. These embedded computers (cyber) may communicate from a tightly coupled way, for example through a serial CAN bus in the automotive domain or through an AFDX bus in the avionics domain to control engine(s) or brakes (physics), to a loosely coupled way for example through the internet network to offer multimedia services or data-base accesses. Because of the heterogeneity of the involved components (multi-physics, sensors, actuators, embedded computers), CPS may feature very complex design and implementation phases as well as complex computer platforms (multi/manycore, multiprocessor, distributed and parallel computers), ever raising the need for effective approaches in order to build reliable systems.

Most of these CPS are time sensitive, i.e. time is a crucial issue which must be carefully mastered, that yet increases their complexity. Mastering time in such CPS is the major objective of the team. Due to their heterogeneous nature, the different components may have different levels of criticality, e.g. engine and brakes have a higher criticality level than multimedia services, which increase the difficulty in the design and implementation phases since lower criticality parts must not interfere with higher criticality parts. In the team we mainly address mixed-criticality issues in term of software safety. However, we started to take into account, in addition, security issues (cyber attacks).

The members of the team beeing involved for a long time in *synchronous languages*, we address the design of CPS with models compliant with the semantics of these languages. Theses models are basicaly graphs and more specifically “clocked graphs” that model data dependences beetween the functions of the functional specification as well as “logical clocks” that are attached to every function. These logical clocks may be related to physical clocks which correspond to periods of functions. These periods are defined by automatic control engineers and are not dependent of the platform. Such approach allows verifications on the functional specification, guaranteeing that the output events of the control system obtained “in reaction” to some input events, are consistent with the input events that triggered them. Verifying functional specifications very early in the design phase, prevents a lot of classic errors found usually later on during the implementation phase. This approach is an important step for providing “correct by construction” implementations. However, non functional specifications must also be taken into consideration. Indeed, to perform real-time schedulability analyses used to guarantee that the implementation is correct in terms of time, we need for every function its worst case execution times (WCET) and for every dependence its worst case communication times (WCCT). Both worst case execution and communication times are dependent of the platform. Using these

worst case times, schedulability analyses are able to compute worst case response times and end-to-end worst case execution times in order to verify if real-time constraints, e.g. deadline, imposed by automatic control engineers, are met. Note that, unfortunately, automatic control engineers define these constraints whereas they usually do not know the platform that will be used later on in the developpement process.

This is the reason why, in the non functionnal specifications we need precise models that encompass important features found at different levels of the platform architecture, e.g. at a high level the number of cores, their means of communication, at a low level the structure of the caches, pipelines, etc. Depending on the complexity of the platform the problem of estimating these worst case times may be more or less difficult. In the case of simple predictable processors and buses, both used presently in the industry for critical railways and avionics applications, the estimation of worst case times is relatively easy. For this purpose we use static analyses or techniques based on measurements for the WCETs for example. However, due to the ever increasing smartphone market, the microprocessor industry provides more and more general purpose platforms based on multicore and, in a near future, based on manycore. These platform have complex architectures that are not predictable due to, e.g. multiple levels of cache and pipeline, speculative branching, communicating through shared memory or/and through a network on chip, etc. Therefore, nowadays the CPS industry has to face the great challenge of using such off the shelf platforms and consequently to estimate the corresponding worst case times of the programs (tasks) that they will execute.

From functional and non functional specifications of the design phase we intend to synthesize, as automatically as possible, based on the real-time schedulability theory, an implementation that is correct by construction. This synthesizing process is close to the process used in language compilation but, in addition, it must take into account more complex non functional specifications. On the other hand, when platforms are not predictable an alternative to the classic estimation of worst case times mentioned previously, consists in reformulating the different problems in a probabilistic framework.

The overall objectives given above lead to three main research programs that are detailed below.

3. Research Program

3.1. The Algorithm-Architecture Adequation methodology and Real-Time Scheduling

Participants: Liliana Cucu, Dumitru Potop-Butucaru, Yves Sorel.

The Algorithm-Architecture Adequation (AAA) methodology relies on distributed real-time schedulability and optimization theories to map efficiently an algorithm model to an architecture model.

The algorithm model which describes the functional specifications of the applications, is an extension of the well known data-flow model from Dennis [16]. It is a directed acyclic hyper-graph (DAG) that we call "conditioned factorized data dependence graph", whose vertices are functions and hyper-edges are directed "data or control dependences" between functions. The data dependences define a partial order on the functions execution. The basic data-flow model was extended in three directions: first infinite (resp. finite) repetition of a sub-graph pattern in order to specify the reactive aspect of real-time systems (resp. in order to specify the finite repetition of a sub-graph consuming different data similar to a loop in imperative languages), second "state" when data dependences are necessary between different infinite repetitions of the sub-graph pattern introducing cycles which must be avoided by introducing specific vertices called "delays" (similar to z -n in automatic control), third "conditioning" of a function by a control dependence similar to conditional control structure in imperative languages, allowing the execution of alternative subgraphs. Delays combined with conditioning allow the programmer to specify automata necessary for describing "mode changes".

The architecture model which describes the non functional specifications is, in the simplest case, a directed graph whose vertices are of two types: “processor” (one sequencer of functions, several sequencers of communications and distributed or shared memories) and “medium” (multiplexers and demultiplexers), and whose edges are directed connections. With such model it is possible to describe classic heterogeneous distributed, parallel and multiprocessor platforms as well as the most recent multi/manycore platforms. The worst case times mentioned previously are estimated according to this model.

The implementation model is a graph obtained by applying an external composition law such that an architecture graph operates on an algorithm graph to give an algorithm graph while taking advantage of timing characteristics, basically periods, deadlines and WCETs. This resulting algorithm graph is built by performing spatial and timing allocations (distribution and scheduling) of algorithm graph functions on architecture graph resources, and of dependences between functions on communication media. In that context "Adequation" means to search, in the solution space of implementation graphs, one implementation graph which verifies real-time constraints and, in addition, minimizes some criteria. These criteria consists in the total execution time of the algorithm executed on the architecture, the number of computing or communication resources, etc. Below, we describe distributed real-time schedulability analyses and optimization techniques suited for that purposes.

We address two main issues: uniprocessor and multiprocessor real-time scheduling for which some real-time constraints are of high criticality, i.e. they must be satisfied otherwise dramatic consequences occur.

In the case of uniprocessor real-time scheduling, besides the usual deadline constraint, often equal to the period of each task, i.e. a function with timing characteristics, we take into consideration dependences beetween tasks, and possibly several latencies. The latter are “end-to-end” constraints that may have complex relationships. Dealing with multiple real-time constraints raises the complexity of the scheduling problems. Moreover, costs of the Real-Time Operating System (RTOS) and of preemptions lead to, at least, a waste of resources due to their approximation in the WCET (Worst Execution Time) of each task, as proposed by Liu and Layland in their seminal article [18]. This is the reason why we first studied non-preemptive real-time scheduling with dependences, periodicities, and latencies constraints. Although a bad approximation of costs of the RTOS and of preemptions, may have dramatic consequences on real-time scheduling, there are only few researches on this topic. Thus, we investigated preemptive real-time scheduling while taking into account its cost which is very difficult to determine because it varies according to the instance (job) of each task. This latter is integrated in the schedulability conditions, and in the corresponding scheduling algorithms we propose. More generally, we integrate in schedulability analyses costs of the RTOS and of preemptions.

In the case of multiprocessor real-time scheduling, we chose to study first the “partitioned approach”, rather than the “global approach”, since the latter uses task migrations whose cost is prohibitive for current commercial processors, even for the more recent many/multicore. The partitioned approach enables us to reuse the results obtained in the uniprocessor case in order to derive solutions for the multiprocessor case. We consider also the semi-partitioned approach which allows only some migrations in order to minimize their costs. In addition, to satisfy the multiple real-time constraints mentioned in the uniprocessor case, we have to minimize the total execution time (makespan) since we deal with automatic control applications involving feedback loops. The complexity of such minimization problem increases because the cost of interprocessor communications (through buses in a multi-processor or routers in a manycore) must be taken into account. Furthermore, the domain of embedded systems leads to solving minimization resources problems. Since both optimization problems are NP-hard we develop exact algorithms (ILP, B & B, B & C) which are optimal for simple problems, and heuristics which are sub-optimal for realistic problems corresponding to industrial needs. Long time ago we proposed a very fast “greedy” heuristics whose results were regularly improved, and extended with local neighborhood heuristics, or used as initial solutions for metaheuristics.

Besides the spatial dimension (distributed) of the real-time scheduling problem, other important dimensions are the type of communication mechanisms (shared memory vs. message passing), or the source of control and synchronization (event-driven vs. time-triggered). We explore real-time scheduling on architectures corresponding to all combinations of the above dimensions. This is of particular impact in application domains such as railways and avionics.

3.2. Probabilistic Worst Case Reasoning for Real-Time Systems

Participants: Liliana Cucu, Robert Davis, Yves Sorel.

The arrival of modern hardware responding to the increasing demand for new functionalities exacerbates the limitations of the current worst-case real-time reasoning, mainly to the rarity of worst-case scenarios. Several solutions exist to overcome this important pessimism and our solution takes into account the extremely low probability of appearance of a worst-case scenario within one hour of functioning (10^{-45}), compared to the certification requirements for instance (10^{-9} for the highest level of certification in avionics). Thus we model and analyze real-time systems with time parameters described by using probabilistic models. Our results for such models address both schedulability analyses as well as timing analyses. Both such analyses are impacted by existing misunderstanding. The independence between tasks is a property of real-time systems that is often used for its basic results. Any complex model takes into account different dependences caused by sharing resources other than the processor. On another hand, the probabilistic operations require, generally, the (probabilistic) independence between the random variables describing some parameters of a probabilistic real-time system. The main (original) criticism to probabilistic is based on this hypothesis of independence judged too restrictive to model real-time systems. In reality the two notions of independence are different. Providing arguments to underline this confusion is at the center of our dissemination effort in the last years.

We provide below the bases driving our current research as follows:

- *Optimality of scheduling algorithms* stays an important aspect of the probabilistic real-time systems, especially that the introduction of probabilistic time parameters has a direct impact on the optimality of the existing scheduling algorithms. For instance Rate Monotonic scheduling policy is no longer optimal in the case of one processor when a preemptive fixed-priority solution exists. We expect other classes of algorithms to lose their optimality and we concentrate our efforts to propose new scheduling solutions in this context [10].
- *Increased complexity of schedulability analysis* due to the introduction of probabilistic parameters requires appropriate complexity reasoning, especially with the emergence of probabilistic schedulability analyses for mixed-criticality real-time systems [4]. Moreover the real-time applications are rarely independent and precedence constraint using graph-based models are appropriate in this context. Precedence constraints do decrease the number of possible schedulers, but they also imposes an "heritage" of probabilistic description from execution times to release times for instance.
- *Proving feasibility intervals* is crucial for these approaches that are often used in industry on top of simulation. As worst-case situations are rare events, then observing them or at least observe those events that do provoke later the appearance of worst-case situations is difficult. By proposing an iterative process of composition between different statistical models [13], we provide the basis to build a solution to this essential problem to prove any probabilistic real-time reasoning based on measurements.
- *Providing representativeness* of a measurement-based estimator is the final proof that a probabilistic worst-case reasoning may receive. Our first negative results [3] indicate that the measurement protocol is tighly connected to the statistical estimator and that both must verified properties of reproducibility in order to contribute to a convergence proof.

3.3. Real-Time Systems Compilation

Participant: Dumitru Potop-Butucaru.

In the early days of embedded computing, most software development activities were manual. This is no longer true at the low level, where manual assembly coding has been almost completely replaced with the combined use of so-called “high-level” languages (C, Ada, *etc.*) and the use of compilers. This was made possible by the early adoption of standard interfaces that allowed the definition of economically-viable compilation tools with a large-enough user base. These interfaces include not only the programming languages (C, Ada, *etc.*), but also relatively stable microprocessor instruction set architectures (ISAs) or executable code formats like ELF.

The paradigm shift towards fully automated code generation is far from being completed at the system level, mainly due to the slower introduction of standard interfaces. This also explains why real-time scheduling has historically dedicated much of its research effort to verifying the correctness of very abstract and relatively standard implementation models (the task models). The actual construction of the implementations and the abstraction of these implementations as task models drew comparatively less interest, because they were application-dependent and non-portable.

But today the situation is bound to change. First, automation can no longer be avoided, as the complexity of systems steadily increases in both specification size (number of tasks, processors, etc.) and complexity of the objects involved (parallelized dependent tasks, multiple modes and criticalities, many-cores, *etc.*). Second, fully automated implementation is attainable for industrially significant classes of systems, due to significant advances in the standardization of both specification languages (Simulink, Scade, etc.) and of implementation platforms (ARINC, AUTOSAR, *etc.*).

To allow the automatic implementation of complex embedded systems, we advocate for a *real-time systems compilation* approach that combines aspects of both real-time scheduling – including the AAA methodology – and (classical) compilation. Like a classical compiler such as GCC, a real-time systems compiler should use fast and efficient scheduling and code generation heuristics, to ensure scalability. Similarly, it should provide traceability support under the form of informative error messages enabling an incremental trial-and-error design style, much like that of classical application software. This is more difficult than in a classical compiler, given the complexity of the transformation flow (creation of tasks, allocation, scheduling, synthesis of communication and synchronization code, *etc.*), and requires a full formal integration along the whole flow, including the crucial issue of correct hardware/platform abstraction.

A real-time systems compiler should perform precise, conservative timing accounting along the whole scheduling and code generation flow, allowing it to produce safe and tight real-time guarantees. In particular, resource allocation, timing analysis, and code generation must be tightly integrated to ensure that generated code (including communication and synchronization primitive calls) satisfies the timing hypotheses used for scheduling. More generally, and unlike in classical compilers, the allocation and scheduling algorithms must take into account a variety of non-functional requirements, such as real-time constraints, criticality/partitioning, preemptability, allocation constraints, *etc.* As the accent is put on the respect of requirements (as opposed to optimization of a metric, like in classical compilation), resulting scheduling problems are quite different from those of classical compilation.

We have designed and built a prototype real-time systems compiler, called LoPhT, for statically scheduled real-time systems. Results on industrial case studies are encouraging, hinting not only at the engineering potential of the approach, but also at the scientific research directions it opens.

One key issue here is sound hardware/platform abstraction. To prove that it is possible to reconcile performance with predictability in a fully automatic way, we started in the best possible configuration with industrial relevance: statically-scheduled software running on very predictable (yet realistic) platforms. Already, in this case, platform modeling is more complex than the one of classical compilation [1] or real-time scheduling. [2] The objective is now to move beyond this application class to more dynamic classes of specifications implementations, but without losing too much of the predictability and/or effciency.

Efficiency is also a critical issue in practical systems design, and we must invest more in the design of optimizations that improve the *worst-case* behavior of applications and take into account non-functional requirements in a *multi-objective optimization* perspective, but while remaining in the class of low-complexity heuristics to ensure scalability. Optimizations of classical compilation, such as loop unrolling, retiming, and inlining, can serve as inspiration.

Ensuring the safety and efficiency of the generated code cannot be done by a single team. Collaborations on the subject will have to cover at least the following subjects: the interaction between real-time scheduling

[1] Because safe timing accounting is needed.
[2] The compiler must perform safe timing accounting, and not rely on experience-derived margins.

and WCET analysis, the design of predictable hardware and software architectures, programming language support for efficient compilation, and formally proving the correctness of the compiler.

4. Application Domains

4.1. Avionics

Participants: Liliana Cucu, Keryan Didier, Adriana Gogonel, Cristian Maxim, Dumitru Potop-Butucaru, Yves Sorel.

A large number of our activities, in analysis, modelling, design and implementation of real-time embedded systems addresses specific applications mainly in the avionics field (with partners such as Airbus, Thales, Safran, etc.) (in the CAPACITES and ASSUME projects 9.1.2.1, 9.2.1.1).

4.2. Many-Core Embedded Architectures

Participants: Liliana Cucu, Keryan Didier, Dumitru Potop-Butucaru, Anselme Revuz, Yves Sorel.

The AAA approach (fitting embedded applications onto embedded architectures) requires a sufficiently precise description of (a model of) the architecture (description platform). Such platforms become increasingly heterogeneous, and we had to consider a number of emerging ones with that goal in mind, such as Kalray MPPA (in the CAPACITES and ASSUME projects 9.1.2.1, 9.2.1.1).

4.3. Railways

Participants: Liliana Cucu, Adriana Gogonel, Walid Talaboulma.

The statistical estimation of bounds on the execution time of a program on a processor is applied in the context of railroad crossing in the context of the collaborative project DEPARTS 9.1.2.2.

5. Highlights of the Year

5.1. Highlights of the Year

Our team has hosted for the first time in France the 38th Real-Time Systems Symposium (RTSS'17) which is the flag conference of our research domain. All the members of team jointly participated to the big effort of ensuring an excellent edition.

6. New Software and Platforms

6.1. SynDEx

KEYWORDS: Distributed - Optimization - Real time - Embedded systems - Scheduling analyses
SCIENTIFIC DESCRIPTION: SynDEx is a system level CAD software implementing the AAA methodology for rapid prototyping and for optimizing distributed real-time embedded applications. It is developed in OCaML.

Architectures are represented as graphical block diagrams composed of programmable (processors) and non-programmable (ASIC, FPGA) computing components, interconnected by communication media (shared memories, links and busses for message passing). In order to deal with heterogeneous architectures it may feature several components of the same kind but with different characteristics.

Two types of non-functional properties can be specified for each task of the algorithm graph. First, a period that does not depend on the hardware architecture. Second, real-time features that depend on the different types of hardware components, ranging amongst execution and data transfer time, memory, etc.. Requirements are generally constraints on deadline equal to period, latency between any pair of tasks in the algorithm graph, dependence between tasks, etc.

Exploration of alternative allocations of the algorithm onto the architecture may be performed manually and/or automatically. The latter is achieved by performing real-time multiprocessor schedulability analyses and optimization heuristics based on the minimization of temporal or resource criteria. For example while satisfying deadline and latency constraints they can minimize the total execution time (makespan) of the application onto the given architecture, as well as the amount of memory. The results of each exploration is visualized as timing diagrams simulating the distributed real-time implementation.

Finally, real-time distributed embedded code can be automatically generated for dedicated distributed real-time executives, possibly calling services of resident real-time operating systems such as Linux/RTAI or Osek for instance. These executives are deadlock-free, based on off-line scheduling policies. Dedicated executives induce minimal overhead, and are built from processor-dependent executive kernels. To this date, executives kernels are provided for: TMS320C40, PIC18F2680, i80386, MC68332, MPC555, i80C196 and Unix/Linux workstations. Executive kernels for other processors can be achieved at reasonable cost following these examples as patterns.
FUNCTIONAL DESCRIPTION: Software for optimising the implementation of embedded distributed real-time applications and generating efficient and correct by construction code
NEWS OF THE YEAR: We improved the distribution and scheduling heuristics to take into account the needs of co-simulation.

- Participant: Yves Sorel
- Contact: Yves Sorel
- URL: http://www.syndex.org

6.2. EVT Kopernic

KEYWORDS: Embedded systems - Worst Case Execution Time - Real-time application - Statistics
SCIENTIFIC DESCRIPTION: The EVT-Kopernic tool is an implementation of the Extreme Value Theory (EVT) for the problem of the statistical estimation of worst-case bounds for the execution time of a program on a processor. Our implementation uses the two versions of EVT - GEV and GPD - to propose two independent methods of estimation. Their results are compared and only results that are sufficiently close allow to validate an estimation. Our tool is proved predictable by its unique choice of block (GEV) and threshold (GPD) while proposant reproducible estimations.
FUNCTIONAL DESCRIPTION: EVT-Kopernic is tool proposing a statistical estimation for bounds on worst-case execution time of a program on a processor. The estimator takes into account dependences between execution times by learning from the history of execution, while dealing also with cases of small variability of the execution times.
NEWS OF THE YEAR: Any statistical estimator should come with an representative measurement protocole based on the processus of composition, proved correct. We propose the first such principle of composition while using a Bayesien modeling taking into account iteratively different measurement models. The composition model has been described in a patent submitted this year with a scientific publication under preparation.

- Participants: Adriana Gogonel and Liliana Cucu
- Contact: Adriana Gogonel
- URL: http://inria-rscript.serveftp.com/

6.3. LoPhT-manycore

Logical to Physical Time compiler for many cores

KEYWORDS: Real time - Compilation - Task scheduling - Automatic parallelization
SCIENTIFIC DESCRIPTION: Lopht is a system-level compiler for embedded systems, whose objective is to fully automate the implementation process for certain classes of embedded systems. Like in a classical compiler (e.g. gcc), its input is formed of two objects. The first is a program providing a platform-indepedent description of the functionality to implement and of the non-functional requirements it must satisfy (e.g. real-time, partitioning). This is provided under the form of a data-flow synchronous program annotated with non-functional requirements. The second is a description of the implementation platform, defining the topology of the platform, the capacity of its elements, and possibly platform-dependent requirements (e.g. allocation).

From these inputs, Lopht produces all the C code and configuration information needed to allow compilation and execution on the physical target platform. Implementations are correct by construction Resulting implementations are functionally correct and satisfy the non-functional requirements. Lopht-manycore is a version of Lopht targeting shared-memory many-core architectures.

The algorithmic core of Lopht-manycore is formed of timing analysis, allocation, scheduling, and code generation heuristics which rely on four fundamental choices. 1) A static (off-line) real-time scheduling approach where allocation and scheduling are represented using time tables (also known as scheduling or reservation tables). 2) Scalability, attained through the use of low-complexity heuristics for all synthesis and associated analysis steps. 3) Efficiency (of generated implementations) is attained through the use of precise representations of both functionality and the platform, which allow for fine-grain allocation of resources such as CPU, memory, and communication devices such as network-on-chip multiplexers. 4) Full automation, including that of the timing analysis phase.

The last point is characteristic to Lopht-manycore. Existing methods for schedulability analysis and real-time software synthesis assume the existence of a high-level timing characterization that hides much of the hardware complexity. For instance, a common hypothesis is that synchronization and interference costs are accounted for in the duration of computations. However, the high-level timing characterization is seldom (if ever) soundly derived from the properties of the platform and the program. In practice, large margins (e.g. 100%) with little formal justification are added to computation durations to account for hidden hardware complexity. Lopht-manycore overcomes this limitation. Starting from the worst-case execution time (WCET) estimations of computation operations and from a precise and safe timing model of the platform, it maintains a precise timing accounting throughout the mapping process. To do this, timing accounting must take into account all details of allocation, scheduling, and code generation, which in turn must satisfy specific hypotheses.
FUNCTIONAL DESCRIPTION: Accepted input languages for functional specifications include dialects of Lustre such as Heptagon and Scade v4. To ensure the respect of real-time requirements, Lopht-manycore pilots the use of the worst-case execution time (WCET) analysis tool (ait from AbsInt). By doing this, and by using a precise timing model for the platform, Lopht-manycore eliminates the need to adjust the WCET values through the addition of margins to the WCET values that are usually both large and without formal safety guarantees. The output of Lopht-manycore is formed of all the multi-threaded C code and configuration information needed to allow compilation, linking/loading, and real-time execution on the target platform.
NEWS OF THE YEAR: In the framework of the ITEA3 ASSUME project we have extended the Lopht-manycore to allow multiple cores to access the same memory bank at the same time. To do this, the timing accounting of Lopht has been extended to take into account memory access interferences during the allocation and scheduling process. Lopht now also pilots the aiT static WCET analysis tool from AbsInt by generating the analysis scripts, thus ensuring the consistency between the hypotheses made by Lopht and the way timing analysis is performed by aiT. As a result, we are now able to synthesize code for the computing clusters of the Kalray MPPA256 platform. Lopht-manycore is evaluated on avionics case studies in the perspective of increasing its technology readiness level for this application class.

- Participants: Dumitru Potop-Butucaru and Keryan Didier
- Contact: Dumitru Potop-Butucaru

7. New Results

7.1. Uniprocessor Mixed-Criticality Real-Time Scheduling

Participants: Slim Ben-Amor, Liliana Cucu, Robert Davis, Mehdi Mezouak, Yves Sorel.

In the framework of the FUI CEOS project 9.1.1.1 we mainly investigated the PX4 autopilot free software program that was chosen by the partners to be implemented on the Pixhawk electronic board. This board will be installed in the multirotor drone that the project is intended to built. The board is based on a microcontroller which contains an ARM Cortex M4 microprocessor, timers, several sensors, accelerometer, gyroscope, magnetometer, barometer, and actuators, mainly four to eight electric motors depending on the level of redundancy.

We studied the existing source code of PX4 which consists of two main layers: the flight stack, which is an estimation and flight control system, and the middleware, which is a general robotics layer providing internal/external communications and hardware integration. This study allowed us to understand the general architecture of PX4. The flight stack is split into a set of threads communicating asynchronously through a micro object request broker messaging. In the CEOS project our team is in charge to guarantee that the drone will satisfy multiple real-time criticality levels. In order to be able to perform a real-time schedulability analysis on the PX4 autopilot, first we transformed this set of communicating threads into a task dependency graph. Second, we sought the period of each task starting from input tasks which read from sensors, to output tasks which write into actuators. The partners of the project chose to run PX4 on the NuttX OS which is open source, light-weight, efficient and very stable. It provides POSIX API and some form of real-time scheduling. Thus, we had to deeply understand the scheduler and the management of interruptions and time of NuttX. We plan to modify NuttX in order to support mixed-criticality applications using to start, online real-time scheduling, and then offline real-time scheduling.

Finally, always to perform the real-time schedulability analysis of PX4, we must estimate the worst execution time (WCET) of each task. This problem is very complex due to the multiple possible paths in a task as well as the different data it consumes. Moreover, the processor and/or the microcontroller itself may have some features like memory contentions, bus accesses, caches, pipelines, speculative branchings that increase the difficulty to determine WCETs. All these variabilities lead us to introduce probabilistic reasoning in characterizing the timing behavior (WCET, schedulability analyses) of mixed-criticality real-time applications [4].

7.2. Multiprocessor Real-Time Scheduling

Participants: Salah-Eddine Saidi, Yves Sorel.

During the third year of the PhD thesis of Salah Eddine Saidi, we focused on two aspects. First, we finalized our work on the parallelization on multi-core processors of FMI-based co-simulation of numerical models in order to accelerate its execution. Our approach, based on the transformation of FMU graphs into operation graphs which reveal more parallelism, comprises the following two steps: first acyclic orientation necessary for avoiding that some operations of a same model are executed in parallel and second multi-core offline scheduling of operations [5]. We proposed exact algorithms based on ILP (Integer Linear Programming) and heuristics for performing the acyclic orientation and the multi-core scheduling. Also, we proposed a random generator of synthetic co-simulations. Using these generated co-simulations, we compared the performances of the heuristics and the ILP-based exact algorithm for both the acyclic orientation and the scheduling in terms of execution time and quality of the obtained solution. Tests have been carried out for different sizes of co-simulation and different numbers of cores. Moreover, we compared the performance of our offline approach with an online scheduling approach based on the Intel TBB runtime library. This comparison was acheived by applying both approaches on an industrial use case which consists in a co-simulation of a four cylinder spark ignition engine. The various tests that we performed showed the efficieny of our proposed heuristics. Second, we focused on the parallelization of FMI-based co-simulation under real-time constraints. In particular, we

were interested in HiL (Hardware-in-the-Loop) co-simulation where a part of the co-simulation is replaced by its real counterpart that is physically available. The real and simulated parts have to exchange data during the execution of the co-simulation under real-time constraints. In other words, the inputs and ouputs of the real part are sampled periodically, sending and receving data to and from the simulated part. This periodic data exchange defines a set of real-time constraints to be satisfied by the simulated part. We proposed a method for defining these real-time constraints and propagating them to all the operations of the co-simulation (simulated part). In our ongoing work, we are focusing on multi-core scheduling of FMI-based co-simulation under real-time constraints. More precisely, we are working on a heuristic and an ILP-based algorithm that will enable the execution of the co-simulation on a multi-core processor while ensuring the defined real-time constraints are respected.

7.3. Principles of Probabilistic Composition

Participants: Slim Ben-Amor, Liliana Cucu, Adriana Gogonel, Cristian Maxim.

The statistical estimation of time parameters for real-time systems is proposed at two levels:

1. at program level and in this case we are dealing with timing analysis of programs that requires later appropriate probabilistic composition principles like reproducibility and representativity [3], [1]. For instance we have underlined in [14] the difficulties to ensure such properties for many-cores architectures.

   While we are proposing static analyses using worst-case bounds on the execution at instruction level for specialized architectures [2], we are interested also in proposing composition principles allowing to combine the timing impact of execution time variation factors, identified as a key open problem in the context of the timing analysis of programs while using the Extreme Value Theory [1]. Our composition solution is based on a Bayesian modeling that considers iteratively the inclusion of new factors while a representative measurement protocole is built [13] with respect to the reproducible Extreme Value Theory-based estimator that we have proposed.

2. at system level and in this case we are dealing with schedulability analysis of set of programs, a.k..a tasks, that requires appropriate composition principles like probabilistic independence while the dependence between tasks is taken into account. After proposing a first solution to the schedulability analysis of real-time probabilistic tasks in presence of precedence constraints on uniprocessor system [6], we explore the state of art of real-time scheduling on multiprocessor system and probabilistic real-time existing analysis. Our choice goes to partitioned multiprocessor scheduling to ensure the applicability of our previous results in the case of one processor. We have proposed a first optimal partitioning strategy based individual task utilization and we compare different tasks combinations that fit on a single processor following an utilization task ratio principle as partitioning choice. When assessing our method, a counter example of a possible optimality has appeared. Moreover this method has not an important improvement compared to existing partitioning strategies like best fit. Therefore we prepare the application of an existing solution to the bin packing problem [17] proposed in mathematics domain to partition real-time tasks on multiprocessor system in order to propose an appropriate probabilistic analysis.

   The exact schedulability analyses are often competing with statistical estimation of response time based on simulation and we propose such result in [9]. Such results allow to advance on the understanding of the notion of representativeness in the context of our problem that becomes today central in our community. The explosion of probabilistic schedulability analyses published in the last years have convinced us to join the book proposal of a Handbook on Real-Time Computing in order to integrate a comprehensive description of these analyses [4].

7.4. pWCET Estimation: a System Concern

Participants: Irina-Mariuca Asavoae, Mihail Asavoae, Slim Ben-Amor, Antoine Bertout, Liliana Cucu, Adriana Gogonel, Tomasz Kloda, Cristian Maxim, Walid Talaboulma.

From modelling to time validation, the design of an embedded system may benefit from a better utilisation of probabilities while providing means to prove their results. The arrival of new complex processors has made the time analysis of the programs more difficult while there is a growing need to integrate uncertainties from all levels of the embedded systems design. Probabilistic and statistical approaches are one possible solution and they require appropriate proofs in order to be accepted by both scientific community and industry. Such proofs cannot be limited at processor or program level and we plead for a system approach in order to take into account the possible interactions between different design levels by using the probabilistic formulation as compositional principle.

Our first arguments are provided by a valid statistical estimation of bounds on the execution time of a program on a processor. More precisely, the probabilistic worst-case execution time (pWCET) $\mathcal{C}$ of a program is an upper bound on all possible probabilistic execution times $\mathcal{C}_i$ for all possible execution scenarios $S_i, \forall i \geq 1$. According to EVT if the maximum of execution times of a program converges, then this maximum of the execution times $\mathcal{C}_i, \forall i \geq 1$ converges to one of the three possible Generalized Extreme Value (GEV) laws: Fréchet, Weibull and Gumbel corresponding to a shape parameter $\xi > 0$, $\xi < 0$, and $\xi = 0$, respectively. EVT has two different formulations: Generalized Extreme Value (GEV) and Generalized Pareto Distribution (GPD) and the difference between them is the way the extreme values are selected. GEV is based on the block maxima reasoning, grouping execution times by chronological groups (called blocks) and only the largest value of each group is considered as an extreme value. GPD is a method based on the threshold approach that considers only the values larger than the chosen threshold as extreme values. The voting procedure is based on the utilization of the both formulations of the EVT.

- **Block size estimation :** The GEV models obtained for different block sizes (BS), BS from 10 to $\frac{n}{10}$ are compared, where n is the cardinal of the trace of execution times. We compare the models fitting the extreme values corresponding to each choice of BS and the evolution of the shape parameter function of BS. We keep the BS that assures the best compromise between fitting the data and having a shape parameter within a stability interval of a range of shape parameters estimates. The way GEV models fit the data is analyzed within the tool by using a graphical method including the qqplot and the return level plot. We keep the GEV model corresponding to the shape parameter as the result of the aforementioned compromise and we compute the pWCET as the $1 - CDF$ (inverse of the cumulative distribution function) of the GEV.
- **Threshold level estimation :** The procedure is similar to the GEV procedure. All GPD models obtained for different threshold levels from 80% to 99%, are compared. In the same way as for GEV, we compare the models fitting the extreme values corresponding to each threshold and the evolution of the shape parameter function of threshold. At the end we keep the threshold level assuring the best compromise between fitting the data (graphical method) and having the shape parameter within a stability interval of a range of shape parameters estimates. We also consider the mean residual life plot (mean of excess) that may be consulted in case of a doubt between two different thresholds, we will prefer the threshold level such that the curve of mean of excess experiences linearity. We keep the GPD model corresponding to the shape parameter resulting from the aforementioned compromise and we compute the pWCET as the $1 - CDF$ of the GPD.
- **Comparing GEV and GPD pWCET estimates :** The comparison of the pWCET obtained with both methods, GEV and GPD is done graphically. Superposing the two curves allows to compare the distance between the two distributions. If an important difference is noticed, other GEV/GPD models are tested. In such cases calculating the pWCET estimate as a combination of GEV and GPD results is also recommended. A joint pWCET estimate is obtained by choosing for each probability the largest value between GEV and GPD . The tool implementing this method is available on line at `inria-rscript.serveftp.com` (requires a secured connection to be provided under request) [8].
- **Conditions of use :** The application of EVT requires to verify that the analyzed data are identically distributed, i.e., the execution times are following the same (unknown) probability distribution. That condition is tested before the analysis is started, and data is treated according to the test results. Another EVT applicability condition is the independence of the data. That condition is not mandatory in the sense that non-independent data can be analyzed. The case of dependent data can be split in

two sub cases. The first one is where there are dependencies within the data, still the picked extremes values are independent. In that case the analysis will be done in the same way as for the independent data. The second case is the one where there are dependencies also between the extreme values. In that case one more step is added in the procedure. This step is the de-clustering process before applying GPD and the use of the index while GEV is applied.

During the second year of PhD thesis of Talaboulma Walid, we continued exploring solutions to WCET (Worst Case Execution Time) estimation and Real Time Scheduling on multiprocessors. WCET analysis done on a monoprocessor system (in isolation) can no longer be trusted to be accurate when we run our tasks on a multiprocessor (two processors), the problem of Co-runner interference arises and this is due to contention in shared hardware, two processors share the same memory and contention will occur when a simultaneous access is done, thus delaying one of the request, and this can counter-intuitively make programs run longer in a multiprocessor than what the analysis predicted on a monoprocessor, leading to deadline misses. In [20] authors evaluate explicit reservation of cache memory to reduce the cache-related preemption delay observed when tasks share a cache in a preemptive multitasking hard real-time system. Another solution is presented in [19] by management of tasks shared resources access using performance counter to stop tasks when they exceed their allocated budget (for instance cache misses) and thus providing guarantees on global memory bandwidths, moreover in [15] some offline analysis is done using heuristics to find optimal time triggered schedules for shared memory access.

We propose in our work to generate programs memory access profile, that we obtain by running tasks on a cycle accurate System Simulator, with a precise cycle accurate model of DDRAM memory controller and a full model of memory hierarchy including caches and main memory devices, and we log every memory event that occurs inside the simulation, our approach doesn't necessitate modifications of software layer, or recompilation of task code First we focus on simple tasks with few branches and simple memory access patterns as a proof of concept, and we choose a COTS (component of the shelf) platform with two complex processor cores. We intend to loosen those constraints when our analysis is matured. We use those profiles to account for co runners interference and add it to WCET value obtained in isolation, and then update our schedule, we can also insert idle times at correct scheduling events to decrease this interference, and in the future use a modified memory management system to pre-load specific memory areas into the cache and thus slide those access back in time to eliminate simultaneous memory access and converge toward an isolation WCET value.

7.5. Safe Parallelization of Hard Real-Time Avionics Software

Participants: Keryan Didier, Dumitru Potop-Butucaru.

This work took place in the framework of the ITEA3 ASSUME project, which funds the PhD thesis of Keryan Didier, and in close collaboration with Inria PARKAS, Airbus, and Kalray.

Concurrent programming is notoriously difficult, especially in constrained embedded contexts. Threads, in particular, are wildly nondeterministic as a model of computation, and difficult to analyze in the general case. Fortunately, it is often the case that multi-threaded, semaphore-synchronized embedded software implements high-level functional specifications written in a deterministic data-flow language such as Scade or (safe subsets of) Simulink.

In many cases, the multi-threaded implementation of such specifications preserves a fundamentally dataflow structure, with specific rules on the way platform resources (shared memory, semaphores) are used. When this happens, the implementation is best represented as a dataflow synchronous program whose elements are mapped on the platform resources. Ensuring the correctness of such an implementation consists in ensuring that:

1. The dataflow program (without the mapping) implements the semantics of the functional specification. This analysis can be performed inside the dataflow model.

2. Once the mapping of program elements onto the platform resources [3] is performed, the execution of the platform (under platform semantics) implements the behavior of the dataflow program.

Together, the dataflow program and the mapping information form an *implementation model*. This model is strictly richer than the multi-threaded C code, which can be obtained through a pretty-printing of model parts. Exposing the internal data-flow structure of the implementation facilitates defining and establishing correctness, *e.g.* the correctness of the synchronization or memory coherence protocols synthesized during the implementation process. All analyses can be realized using efficient tools specific to the synchronous model. Finally, if manual inspection of the C multi-threaded code is required, such a representation can be used to enforce strict code structuring rules which facilitate understanding.

We proposed a language for describing such implementation models that expose the data-flow behavior hiding under the form of a multi-threaded program. The language allows the representation of efficient implementations featuring pipelined scheduling and optimized memory allocation and synchronization [12].

We also proposed a design and tool flow taking as input industrial specifications based on Lustre/Scade and automatically producing fully mapped parallel implementation models and implementations with hard real-time guarantees. The front-end of the flow implements properties facilitating the mapping, e.g., exposing the state of all nodes to memory optimization. To strictly enforce realtime guarantees, the offline mapping algorithms of the back-end consider all sources of interference, including concurrent memory accesses, coherence protocols and event-driven synchronization. Our flow scales to an avionics application comprising more than 5000 unique nodes, targeting the Kalray MPPA 256 many-core platform, selected for its timing predictability.

7.6. Real-time Platform Modeling

Participants: Fatma Jebali, Dumitru Potop-Butucaru.

One key difficulty in embedded systems design is related to the existence of multiple models of the same system, at different abstraction levels, and used in various phases of the design flow. Usual models include *cycle-accurate, bit-accurate (CABA)* system models used to perform exact simulation for precision tuning, microarchitectural models used during WCET (*Worst-Case Execution Time*) analysis of sequential tasks, and high-level models used during WCRT (*Worst-Case Response Time*) analysis of the whole system. In current practice, these models are developed separately, and it is difficult to ensure (by extensive simulation) that they are consistent.

We explore the possibility of obtaining both a CABA and a WCET microarchitectural simulator from a single source, along with a formal consistency guarantee. This year we considered the timing abstraction issue: Both CABA and WCET simulators use a cycle-based execution model, but the cycle corresponds in one case to hardware clock cycles, and in the other to PC (program counter) advancement. We showed that for architectures satisfying a scheduling-independence property (known as in-order architectures) it is possible to produce from a single source both types of simulations (clock-driven and PC-driven), with a formal correctness guarantee. Preliminary results have been presented at the Synchron'07 workshop.

8. Bilateral Contracts and Grants with Industry

8.1. Bilateral Grants with Industry

The Airbus CIFRE grant which started on March 2014, provides full support for the PhD thesis of Cristian Maxim. The thesis concerns the statistical timing analysis while different variability factors are taken into account. The proposed methods are built on top of existing statistical approaches while proving appropriate programs for training these methods and thus learning from the history of the execution.

[3] Sequencing of blocks into threads executed by processors; code, stack and data variables to memory locations; synchronizations to semaphores, *etc.*

8.2. Bilateral Grants with Industry

The IFPEN grant which started on December 2014, provides full support for the PhD thesis of Salah-Eddine Saidi. The thesis concerns the automatic parallelization and scheduling approaches for co-simulation of numerical models on multi-core processors. The goal of the first research topic is to propose multi-core scheduling solutions for the co-simulation in order to accelerate its execution. The second research topic aims at proposing multi-core scheduling solutions in order to enable the execution of co-simulation under real-time constraints in the context of Hardware-in-the-Loop validation.

9. Partnerships and Cooperations

9.1. National Initiatives

9.1.1. FUI

9.1.1.1. CEOS

Participants: Slim Ben-Amor, Liliana Cucu, Mehdi Mezouak, Yves Sorel, Walid Talaboulma.

This project was started on May 2017. Partners of the project are: ADCIS, ALERION, Aéroports de Lyon, EDF, ENEDIS, RTaW, EDF, Thales Communications and Security, ESIEE engineering school and Lorraine University. The CEOS project delivers a reliable and secure system of inspections of pieces of works using professional mini-drone for Operators of Vital Importance coupled with their Geographical Information System. These inspections are carried out automatically at a lower cost than current solutions employing helicopters or off-road vehicles. Several software applications proposed by the industrial partners, are developed and integrated in the drone, within an innovative mixed-criticality approach using multi-core platforms.

9.1.1.2. WARUNA

Participants: Antoine Bertout, Liliana Cucu, Adriana Gogonel, Tomasz Kloda, Yves Sorel, Walid Talaboulma.

This project was started on September 2015. It targets the creation of a framework allowing to connect different existing methods while enriching the description with Waruna results. This framework allows timing analyses for different application domains like avionics, railways, medical, aerospace, automotive, etc.

9.1.2. PIA

9.1.2.1. CAPACITES

Participants: Liliana Cucu, Cristian Maxim, Dumitru Potop-Butucaru, Yves Sorel, Walid Talaboulma.

This project is funded by the LEOC Call (Logiciel Embarqué et Objets Connectés) of the national support programme Investissements d'Avenir. It was started on November 1st, 2014 with the kick-off meeting held on November, 12th 2014. The project cordinator is Kalray, and the objective of the project is to study the relevance of Kalray-style MPPA processor array for real-time computation in the avionic domain (with partners such as Airbus for instance). The PhD of Walid Talaboulma is funded on this contract.

9.1.2.2. DEPARTS

Participants: Liliana Cucu, Adriana Gogonel, Walid Talaboulma.

This project is funded by the BGLE Call (Briques Logicielles pour le Logiciel Embarqué) of the national support programme Investissements d'Avenir. Formally started on October 1st, 2012 with the kick-off meeting held on April, 2013 for administrative reasons. Research will target solutions for probabilistic component-based models, and a Ph.D. thesis should start at latest on September 2015. The goal is to unify in a common framework probabilistic scheduling techniques with compositional assume/guarantee contracts that have different levels of criticality.

9.2. European Initiatives

9.2.1. Collaborations in European Programs, Except FP7 & H2020

9.2.1.1. ASSUME

Participants: Keryan Didier, Fatma Jebali, Dumitru Potop-Butucaru.

Program: ITEA

Project acronym: ASSUME

Project title: Affordable Safe and Secure Mobility Evolution

Duration: September 2015 - August 2018

Coordinator: Daimler

Other partners: among 38 partners Absint, Ansys, Airbus, Kalray, Safran, Thales, ENS, KTH, FZI, etc.

Abstract: Future mobility solutions will increasingly rely on smart components that continuously monitor the environment and assume more and more responsibility for a convenient, safe and reliable operation. Currently the single most important roadblock for this market is the ability to come up with an affordable, safe multi-core development methodology that allows industry to deliver trustworthy new functions at competitive prices. ASSUME will provide a seamless engineering methodology, which addresses this roadblock on the constructive and analytic side.

9.2.2. Collaborations with Major European Organizations

University of York: Real-Time System Group (UK)

Uncertainties in real-time systems: the utilization of extreme value theory has received increased efforts from our community and more rigorous principles are needed for its full understanding. Our two research teams have gathered these principles in a joint publication.

9.3. International Research Visitors

9.3.1. Visits of International Scientists

Professor George Lima (University of Baha, Brazil) visited us between May and June. His stay was dedicated the study of the utilization of extreme value theory on the problem of probabilistic estimation of worst case execution time bounds for a program on a processor.

10. Dissemination

10.1. Promoting Scientific Activities

10.1.1. Scientific Events Organisation

10.1.1.1. Member of the Steering Committees

- Liliana Cucu-Grosjen is member of the steering committees of the following conferences and workshops: RTSS, RTAS, RTNS, WMC, RTSOPS.
- Rob Davis is member of the steering committees of the following conferences and workshops: RTSS, RTAS, RTNS, WMC, RTSOPS.

10.1.1.2. Member of the Organizing Committees

- Liliana Cucu is Local Arrangement Chair of the 38th IEEE Real-time Systems Symposium (RTSS'17).

10.1.2. Scientific Events Selection

10.1.2.1. Member of the Conference Program Committees

- Liliana Cucu: RTAS, RTNS, WFCS
- Robert Davis: RTSS, RTAS, RTNS
- Adriana Gogonel: ACM RACS, WMC
- Dumitru Potop-Butucaru: ACSD, EMSOFT
- Yves Sorel: DASIP

10.1.2.2. Reviewer

All members of the team are regularly serving as reviewers for the main scientific events of our domain: RTSS, RTAS, RTCSA, RTNS, DATE, ETFA, EMSOFT, DASIP, etc.

10.1.3. Journal

10.1.3.1. Reviewer - Reviewing Activities

All members of the team are regularly serving as reviewers for the main journals of our domain: Journal of Real-Time Systems, Journal of Systems Architecture, Leibniz Transactions on Embedded Systems, IEEE Transactions on Industrial Informatics, etc.

10.1.4. Invited Talks

- Liliana Cucu is keynote speaker at the 11th edition of CRTS, invited speaker at MMR'17 and MEFOSYLOMA seminar.

10.1.5. Leadership within the Scientific Community

Liliana Cucu and Rob Davis are the scientific organizers of the 2nd Dagstuhl seminar on mixed-criticality systems.

10.1.6. Scientific Expertise

- Yves Sorel: Steering Committee of System Design and Development Tools Group of Systematic Paris-Region Cluster.
- Yves Sorel: Steering Committee of Technologies and Tools Program of SystemX Institute for Technological Research (IRT).

10.1.7. Research Administration

- Liliana Cucu-Grosjean is member of Inria Evaluation Commission, co-chair of Inria Committes on gender equality and equal oportunities, and member of the CLHCST.
- Dumitru Potop-Butucaru is member of mobility grant commission for postdocs and invited professors.

10.2. Teaching - Supervision - Juries

10.2.1. Teaching

Master: Liliana Cucu, Distributed Databases and Statistics in Computer Science, 64h, U. Dunarea de Jos, Romania (Invited Professor).

Master: Dumitru Potop Butucaru, A synchronous approach to the design of embedded real-time systems, 30h, M1, EPITA Engineering School, Paris France.

Master: Yves Sorel, Optimization of distributed real-time embedded systems, 38H, M2, University of Paris Sud, France.

Master: Yves Sorel, Synchronous languages and real-time scheduling, 9H, M2, University of Paris-Est Créteil, France.

Master: Yves Sorel, Correct by construction design of reactive systems, 18H, M2, ESIEE Engineering School, Noisy-Le-Grand, France.

10.2.2. Supervision

PhD: Cristian Maxim, End to end constraints using probabilistic approaches, UPMC, defended December 2017, supervised by Liliana Cucu.

PhD in progress: Slim Ben-Amor, Schedulability analysis of probabilistic real-time tasks under end to end constraints, UPMC, started on September 2016, supervised by Liliana Cucu.

PhD in progress: Keryan Didier, Formal certification of real-time implementations, Université Pierre et Marie Curie/EDITE, started November 2015, supervised by Dumitru Potop Butucaru.

PhD in progress: Cristian Maxim, End to end constraints using probabilistic approaches, UPMC, started March 2014, supervised by Liliana Cucu.

PhD in progress: Evariste Ntaryamira, Analysis of embedded systems with time and security constraints, UPMC, started on January 2017, supervised by Liliana Cucu and Rachel Akimana.

PhD in progress: Walid Talaboulma, Probabilistic timing analysis in presence of dependences, UPMC, started November 2015, co-supervised by Liliana Cucu and Adriana Gogonel.

PhD in progress: Salah-Edinne Saidi, Distributed real-time scheduling for the co-simulation of multiple control models, University of UMPC-Paris-Sorbonne, started December 2014, co-supervised by Nicolas Pernet (IFPEN) and Yves Sorel.

10.2.3. Juries

- Liliana Cucu is Phd reviewer for the thesis of Fabrice Guet, ONERA and ISAE, defended December 2017.
- Liliana Cucu is Phd reviewer for the thesis of Bader Alahmad, University of British Columbia, defended December 2017.
- Liliana Cucu is Phd jury member for the thesis of Romain Gratia, Telecom Paritech, defended January 2017.

10.3. Popularization

Popularization video of the probabilistic notions for mixed-criticality systems https://www.youtube.com/watch?v=sSJT4eGhS_A

11. Bibliography

Publications of the year

Articles in International Peer-Reviewed Journals

[1] S. J. GIL, I. BATE, G. LIMA, L. SANTINELLI, A. GOGONEL, L. CUCU-GROSJEAN. *Open Challenges for Probabilistic Measurement-Based Worst-Case Execution Time*, in "IEEE Embedded Systems Letters", June 2017, vol. 9, n^{o} 3, pp. 69 - 72 [*DOI :* 10.1109/LES.2017.2712858], https://hal.archives-ouvertes.fr/hal-01633802

[2] B. LESAGE, S. ALTMEYER, D. GRIFFIN, L. CUCU-GROSJEAN, R. DAVIS. *On the analysis of random replacement caches using static probabilistic timing methods for multi-path programs*, in "Real-Time Systems / Real Time Systems; The Journal of Real-Time Systems", 2017, pp. 1-82, forthcoming [*DOI :* 10.1007/s11241-017-9295-2], https://hal.archives-ouvertes.fr/hal-01666091

[3] C. MAXIM, A. GOGONEL, I. ASAVOAE, M. ASAVOAE, L. CUCU-GROSJEAN. *Reproducibility and representativity: mandatory properties for the compositionality of measurement-based WCET estimation approaches*, in "ACM SIGBED Review", November 2017, vol. 14, n^{o} 3, pp. 24 - 31 [*DOI :* 10.1145/3166227.3166230], https://hal.archives-ouvertes.fr/hal-01666084

International Conferences with Proceedings

[4] D. I. MAXIM, R. DAVIS, L. I. CUCU-GROSJEAN, A. EASWARAN. *Probabilistic Analysis for Mixed Criticality Systems using Fixed Priority Preemptive Scheduling*, in "RTNS 2017 - International Conference on Real-Time Networks and Systems", Grenoble, France, October 2017, 10 p. [*DOI :* 10.1145/3139258.3139276], https://hal.inria.fr/hal-01614684

[5] S. E. SAIDI, N. PERNET, Y. SOREL. *Automatic parallelization of multi-rate fmi-based co-simulation on multi-core*, in "TMS/DEVS 2017 - Symposium on Theory of Modeling and Simulation", Virginia Beach, United States, ACM, April 2017, Article No. 5, https://hal.inria.fr/hal-01610268

Conferences without Proceedings

[6] S. BEN-AMOR, D. MAXIM, L. CUCU. *Schedulability analysis of dependent probabilistic real-time tasks* , in "MAPSP 2017 - 13th Workshop on Models and Algorithms for Planning and Scheduling Problems", Seeon-Seebruck, Germany, RTNS '16 Proceedings of the 24th International Conference on Real-Time Networks and Systems, ACM, June 2017, pp. 99-107 [*DOI :* 10.1145/2997465.2997499], https://hal.archives-ouvertes.fr/hal-01666138

[7] L. CUCU-GROSJEAN, A. GOGONEL. *Probabilistic foundations for the time predictions of cyber-physical systems*, in "MMR 2017 - 10th International Conference on Mathematical Methods in Reliability", Grenoble, France, July 2017, https://hal.archives-ouvertes.fr/hal-01666293

[8] A. GOGONEL, C. MAXIM, L. CUCU-GROSJEAN. *pWCET estimator for real-time systems*, in "RTSS 2017 - IEEE Real-Time Systems Symposium", Paris, France, December 2017, https://hal.archives-ouvertes.fr/hal-01666342

[9] D. MAXIM, A. BERTOUT. *Analysis and Simulation Tools for Probabilistic Real-Time Systems*, in "8th International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS)", Dubrovnik, Croatia, June 2017, https://hal.archives-ouvertes.fr/hal-01552798

Scientific Books (or Scientific Book chapters)

[10] D. MAXIM, L. CUCU-GROSJEAN, R. DAVIS. *Probabilistic schedulability analysis* , in "Handbook on Real-Time Computing", A. EASWARAN (editor), Handbook on Real-Time Computing, Springer, 2017, forthcoming, https://hal.archives-ouvertes.fr/hal-01666110

Books or Proceedings Editing

[11] L. CUCU-GROSJEAN, R. DAVIS, S. K. BARUAH, Z. STEPHENSON (editors). *Mixed Criticality on Multicore / Manycore Platforms (Dagstuhl Seminar 17131)*, Schloss Dagstuhl, 2017, pp. 70-98 [*DOI :* 10.4230/DAGREP.7.3.70], https://hal.archives-ouvertes.fr/hal-01666118

Research Reports

[12] K. DIDIER, A. COHEN, A. GAUFFRIAU, A. GRAILLAT, D. POTOP-BUTUCARU. *Sheep in wolf's clothing: Implementation models for data-flow multi-threaded software*, Inria Paris, April 2017, n° RR-9057, 31 p. , https://hal.inria.fr/hal-01509314

Patents and standards

[13] A. GOGONEL, L. CUCU-GROSJEAN. *Dispositif de caractérisation et/ou de modélisation de temps d'exécution pire-cas*, June 2017, n° 1000408053, https://hal.archives-ouvertes.fr/hal-01666535

Other Publications

[14] A. REVUZ, L. CUCU-GROSJEAN. *Towards statistical estimation of worst case inter-core communications*, October 2017, JRWRTC 2017 - 11th Junior Researcher Workshop on Real-Time Computing, Poster, https://hal.inria.fr/hal-01666243

References in notes

[15] M. BECKER, D. DASARI, B. NIKOLIC, B. AKESSON, V. NÉLIS, T. NOLTE. *Contention-Free Execution of Automotive Applications on a Clustered Many-Core Platform*, in "28th Euromicro Conference on Real-Time Systems, ECRTS 2016, Toulouse, France, July 5-8, 2016", 2016, pp. 14–24, https://doi.org/10.1109/ECRTS.2016.14

[16] J. DENNIS. *First Version of a Dataflow Procedure Language*, in "Lecture Notes in Computer Sci.", Springer-Verlag, 1975, vol. 19, pp. 362-376

[17] R. E. KORF. *A New Algorithm for Optimal Bin Packing*, in "Eighteenth National Conference on Artificial Intelligence", 2002, pp. 731–736

[18] C. LIU, J. LAYLAND. *Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment*, in "Journal of the ACM", January 1973, vol. 20, n° 1, pp. 46-61

[19] R. PELLIZZONI, E. BETTI, S. BAK, G. YAO, J. CRISWELL, M. CACCAMO, R. KEGLEY. *A Predictable Execution Model for COTS-Based Embedded Systems*, in "17th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS 2011, Chicago, Illinois, USA, 11-14 April 2011", 2011, pp. 269–279, https://doi.org/10.1109/RTAS.2011.33

[20] J. WHITHAM, N. C. AUDSLEY, R. I. DAVIS. *Explicit Reservation of Cache Memory in a Predictable, Preemptive Multitasking Real-time System*, in "ACM Trans. Embed. Comput. Syst.", April 2014, vol. 13, n° 4s, pp. 120:1–120:25, http://doi.acm.org/10.1145/2523070