RESEARCH CENTRE

Paris

2020
ACTIVITY REPORT

Team

KOPERNIC

Keeping wOrst case reasoning aPpropriatE foR differeNt critICALITIES

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)

DOMAIN

Algorithmics, Programming, Software and Architecture

THEME

Embedded and Real-time Systems

Contents

Team KOPERNIC

Creation of the Team: 2018 July 03

Keywords

Computer sciences and digital sciences

A1.1.1. – Multicore, Manycore

A1.5. – Complex systems

A1.5.1. – Systems of systems

A1.5.2. – Communicating systems

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

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 Team members, visitors, external collaborators

Research Scientists

- Liliana Cucu [Team leader, Inria, Researcher, HDR]
- Yves Sorel [Inria, Senior Researcher]

Faculty Members

- Yasmina Abdeddaïm [ESIEE, Associate Professor, until Jul 2020]
- Avner Bar-Hen [CNAM, Professor, from Sep 2020, HDR]

Post-Doctoral Fellow

- Roberto Medina Bonilla [Inria, until May 2020]

PhD Students

- Slim Ben Amor [Inria, until Dec 2020]
- Evariste Ntaryamira [Univ de Cergy Pontoise]
- Walid Talaboulma [Inria, until September 2020]
- Marwan Wehaiba El Khazen [StatInf, CIFRE, from Oct 2020]
- Kevin Zagalo [Inria]

Interns and Apprentices

- Houssain Boukadida [Inria, from Apr 2020 until Sep 2020]

Administrative Assistants

- Christine Anocq [Inria]
- Nelly Maloisel [Inria]

External Collaborators

- Yasmina Abdeddaïm [ESIEE, from Jul 2020]
- Avner Bar-Hen [CNAM, until Aug 2020]
- Rihab Bennour [StatInf]
- Adriana Gogonel [StatInf]
- Roberto Medina Bonilla [Huawei, from Jun 2020]

2 Overall objectives

The Kopernic members are focusing their research on studying **time for embedded communicating systems**, also known as cyber-physical systems. More precisely, the team proposes a **system-oriented solution** to the problem of studying time properties of the cyber components of a CPS. The solution is expected to be obtained by composing probabilistic and non-probabilistic approaches for CPSs. Moreover, statistical approaches are expected to validate existing hypotheses or propose new ones for the models considered by probabilistic analyses.

The term cyber-physical systems refers to a new generation of systems with integrated computational and physical capabilities that can interact with humans through many new modalities [16]. A defibrillator, a mobile phone, an autonomous car or an aircraft, they all are CPSs. Beside constraints like power consumption, security, size and weight, CPSs may have cyber components required to fulfill their functions within a limited time interval (a.k.a. time dependability), often imposed by the environment, e.g., a physical process controlled by some cyber components. The appearance of communication channels between cyber-physical components, easing the CPS utilization within larger systems, forces cyber components with high criticality to interact with lower criticality cyber components. This interaction is completed by external events from the environnement that has a time impact on the CPS. Moreover, some programs of the cyber components may be executed on time predictable processors and other programs on less time predictable processors.

Different research communities study separately the three design phases of these systems: the modeling, the design and the analysis of CPSs [27]. These phases are repeated iteratively until an appropriate solution is found. During the first phase, the behavior of a system is often described using model-based methods. Other methods exist, but model-driven approaches are widely used by both the research and the industry communities. A solution described by a model is proved (functionally) correct usually by a formal verification method used during the analysis phase (third phase described below).

During the second phase of the design, the physical components (e.g., sensors and actuators) and the cyber components (e.g., programs, messages and embedded processors) are chosen often among those available on the market. However, due to the ever increasing pressure of smartphone market, the microprocessor industry provides general purpose processors based on multicore and, in a near future, based on manycore processors. These processors have complex architectures that are not time predictable due to features like multiple levels of caches and pipelines, speculative branching, communicating through shared memory or/and through a network on chip, internet, etc. Due to the time unpredictability of some processors, nowadays the CPS industry is facing the great challenge of estimating worst case execution times (WCETs) of programs executed on these processors. Indeed, the current complexity of both processors and programs does not allow to propose reasonable worst case bounds. Then, the phase of design ends with the implementation of the cyber components on such processors, where the models are transformed in programs (or messages for the communication channels) manually or by code generation techniques [19].

During the third phase of analysis, the correctness of the cyber components is verified at program level where the functions of the cyber component are implemented. The execution times of programs are estimated either by static analysis, by measurements or by a combination of both approaches [38].

These WCETs are then used as inputs to scheduling problems [29], the highest level of formalization for verifying the time properties of a CPS. The programs are provided a start time within the schedule together with an assignment of resources (processor, memory, communication, etc.). Verifying that a schedule and an associated assignment are a solution for a scheduling problem is known as a schedulability analysis.

The current CPS design, exploiting formal description of the models and their transformation into physical and cyber parts of the CPS, ensures that the functional behavior of the CPS is correct. Unfortunately, there is no formal description guaranteeing today that the execution times of the generated programs is smaller than a given bound. Clearly all communities working on CPS design are aware that **computing takes time** [26], but there is **no CPS solution guaranteeing time predictability** of these systems as **the processors appear late within the design phase** (see Figure 1). Indeed, the choice of the processor is made at the end of the CPS design process, after writing or generating the programs.

Kopernic research hypothesis is that mastering time for CPSs is not possible as long as the processor appears so late within the CPS design process with the consequence that the CPS designer has no

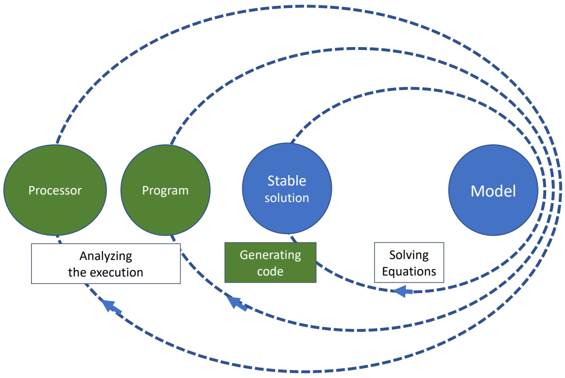


Figure 1: The CPSs design: from models towards analyzing the time properties of the cyber components

influence on the worst case execution time of a program, nor on a schedulability analysis.

Placing the processor central within the CPS design is our major purpose, that we consider achievable by identifying equivalence classes defined on the set of programs with respect to processor features. These classes are integrated within the CPS design at modeling level, as described in Figure 2.

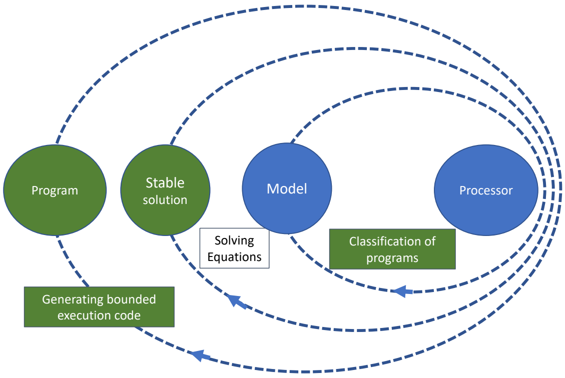


Figure 2: The Kopernic hypothesis puts the processor as the central concern of the CPS design

Before enumerating our scientific objectives, we introduce the concept of variability factors. More precisely, the time properties of a cyber component are subject to variability factors. We understand by variability the distance between the smallest value and the largest value of a time property. With respect to the time properties of a CPS, the factors may be classified in three main classes:

- program structure: for instance, the execution time of a program that has two main branches is obtained, if appropriate composition principles apply, as the maximum between the largest execution time of each branch. In this case the branch is a variability factor on the execution time of the program;
- processor structure: for instance, the execution time of a program on a less predictable processor (e.g., one core, two levels of cache memory and one main memory) will have a larger variability than the execution time of the same program executed on a more predictable processor (e.g., one core, one main memory). In this case the cache memory is a variability factor on the execution time of the program;
- execution environment: for instance, the appearance of a pedestrian in front of a car triggers the execution of the program corresponding to the brakes in an autonomous car. In this case the pedestrian is a variability factor for the triggering of the program.

We identify four main scientific objectives to validate our research hypothesis. The first three objectives are presented from program level, where we use statistical approaches, to the level of communicating programs, where we use probabilistic and non-probabilistic approaches. The fourth one is transversal to the first three objectives and its inclusion is motivated by its capacity to quantify the gain of statistical approaches included within the CPSs design.

The Kopernic scientific objectives are:

- **[O1] worst case execution time estimation of a program** - modern processors induce an increased variability of the execution time of programs, making difficult (or even impossible) a complete static analysis to estimate such worst case. Our objective is to propose a solution composing probabilistic and non-probabilistic approaches based both on static and on statistical analyses by answering the following **scientific challenges**:

    1. **a classification of the variability of execution times** of a program with respect to the processor features. The difficulty of this challenge is related to the definition of an element belonging to the set of variability factors and its mapping to the execution time of the program.
    2. **a compositional rule** of statistical models associated to each variability factor. The difficulty of this challenge comes from the fact that a global maximum of a multicore processor cannot be obtained by upper bounding the local maxima on each core.

- **[O2] deciding the schedulability of all programs running within the same cyber component** - in this case the programs may have different time criticalities, but they share the same processor, possibly multicore[1]. Our objective is to propose a solution composing probabilistic and non-probabilistic approaches based on answers to the following **scientific challenges**:

    1. **scheduling algorithms taking into account the interaction between different variability factors**. The existence of time parameters described by probability distributions imposes to answer to the challenge of revisiting scheduling algorithms that lose their optimality even in the case of an unicore processor [30]. Moreover, the multicore partionning problem is recognized difficult for the non-probabilistic case [37];
    2. **schedulability analyses** based on the algorithms proposed previously. In the case of predictable processors, the schedulability analyses accounting for operating systems costs increase the time dependability of CPSs [33]. Moreover, in presence of variability factors, the composition property of non-probabilistic approaches is lost and new principles are required.

- **[O3] deciding the schedulability of all programs communicating through predictable and non-predictable networks** - in this case the programs of the same cyber component execute on the same processor and they may communicate with the programs of other cyber components through networks that may be predictable (network on chip) or non-predictable (internet, telecommunications). Our objective is to propose a solution to this challenge by analysing schedulability of programs, for which existing (worst case) probabilistic solutions exist [31], communicating through networks, for which probabilistic worst-case solutions [20] and average solutions exist [28].

- **[O4] minimizing the energy consumption** - intuitively the statistical approaches may optimize the CPSs design and the energy consumption is one possible way to quantify the expected gain. The difficulty in achieving such objective comes from the fact that all current models are including CPU frequency variation, when the largest energy consumption feature is the memory access [32].

3 Research program

The research program for reaching these four objectives is organized according three main research axes

- Worst case execution time estimation of a program, detailed in Section 3.1;
- Building measurement-based benchmarks, detailed in Section 3.2;
- Scheduling of graph tasks on different resources, detailed in Section 3.3.

[1] This case is referred as a mixed criticality approach.

3.1 Worst case execution time estimation of a program

The temporal study of real-time systems is based on the estimation of the bounds for their temporal parameters and more precisely the WCET of a program executed on a given processor. The main analyses for estimating WCETs are static analyses [38], dynamic analyses [21], also called measurement-based analyses, and finally hybrid analyses that combine the two previous ones [38].

The Kopernic approach for solving the WCET estimation problem is based on (i) the identification of the impact of variability factors on the execution of a program on a processor and (ii) the proposition of compositional rules allowing to integrate the impact of each factor within a WCET estimation. More precisely, we propose to identify classes of programs and classes of processors features for which we may provide compositional rules. By such identification, we restrain our WCET estimation problem to instances of programs and processors for which compositional rules may be proposed. We say that a rule $\circ$ is compositional for any two sets of measured execution times $\mathscr{C}_1$ and $\mathscr{C}_2$ of a program A, and a WCET statistical estimator p, if we obtain a safe WCET estimation for A from $p(\mathscr{C}_1 \circ \mathscr{C}_2)$. For instance, $\mathscr{C}_1$ may be the set of measured execution times of the program A while all processor features except the local cache L1 are deactivated, while $\mathscr{C}_2$ is obtained, similarly, with a shared L2 cache activated. We consider that the variation of all input variables of the program A follows the same sequence of values, when measuring the execution time of the program p. For instance, our preliminary results indicate that the branch predictors do not allow such compositional rule $\circ$ to exist for the case of single core processors. We identify the following open research problems related to the first research axis:

1. the generalization of statistical modes analysis to multi-dimensional, each dimension representing a program when several programs cooperate;
2. the proposition of a rules set for building programs that are time predictable for the internal architecture of a given unicore and, then, of a multicore processor;
3. modeling the impact of processor features on the energy consumption to better consider both worst case execution time and schedulability analyses.

3.2 Building measurement-based benchmarks

The real-time community is facing the lack of benchmarks adapted to measurement-based analyses. Existing benchmarks for the estimation of WCET [34, 25, 22] have been used to estimate WCETs mainly for static analyses. They contain very simple programs and are not accompanied by a measurement protocol. They do not take into account functional dependencies between programs, mainly due to shared global variables which, of course, influence their execution times. Furthermore, current benchmarks do not take into account interferences due to the competition for resources, e.g., the memory shared by the different cores in a multicore. On the other hand, measurement-based analyses require execution times measured while executing programs on embedded processors, similar to those used in the embedded systems industry. For example, the mobile phone industry uses multicore based on non predictable cores with complex internal architecture, such as those of the ARM Cortex-A family. In a near future, these multicore will be found in critical embedded systems found in application domains such as avionics, autonomous cars, railway, etc., in which the team is deeply involved. This increases dramatically the complexity of measurement-based analyses compared to analyses performed on general purpose personal computers as they are currently performed.

Proposing reproducibility and representativity properties that measurement-based benchmarks should follow is the strength of this research axis. We understand by measurement-based benchmarks a 3-uple composed by a program, a processor and a measurement protocol. The associated measurement protocols should detail the variation of the input variables (associated to sensors) of these benchmarks and their impact on the output variables (associated to actuators), as well as the variation of the processor states. We understand by the reproducibility, the property of a measurement protocol to provide the same ordered set of execution times for a fixed pair (program, processor). We understand by the representativity, the existence of a (sufficiently small) number of values for the input variables allowing a measurement protocol to provide an ordered set of execution times that ensure a convergence for the Extreme Value Index estimators.

Within this research axis we identify the following open problems:

1. proving reproducibility and representativity properties while extending current benchmarks from predictable unicore processors (e.g., ARM Cortex-M4) to non predictable ones (e.g., ARM Cortex-A53 or Cortex-A7);

2. proving reproducibility and representativity properties while extending unicore benchmarks to multicore processors. In this context, we face the supplementary difficulty of defining the principles that an operating system should satisfy in order to ensure a real-time behaviour.

3.3 Scheduling of graph tasks on different resources

As stressed in the previous sections, the utilisation of multicore processors is the current trend of the CPS industry. On the other hand, following the model-driven approach, the functional description of the cyber part of the CPS, is performed as a graph of dependent functions, e.g., a block diagram of functions in Simulink, the most widely used modeling/simulation tool in industry. Of course, a program is associated to every function. Since the graph of dependent programs becomes a set of dependent tasks when real-time constraints must be taken into account, we are facing the problem of verifying the schedulability of such dependent task sets when it is executed on a multicore processor.

Directed Acyclic Graphs (DAG) are widely used to model different types of dependent task sets. The typical model consists of a set of independent tasks where every task is described by a DAG of dependent sub-tasks with the same period inherited from the period of each task [17]. In such DAG, the sub-tasks are vertices and edges are dependencies between sub-tasks. This model is well suited to represent, for example, the engine controller of a car described with Simulink. The multicore schedulability analysis may be of two types, global or partitionned. To reduce interference and interactions between sub-tasks, we focus on partitioned scheduling where each sub-task is assigned to a given core [23].

In order to propose a general DAG task model, we identify the following open research problems:

1. a general schedulability problem where tasks are executed on predictable and non predictable processors, and such that some tasks communicate through predictable networks, e.g., inside a multicore or a manycore processor, and non-predictable networks, e.g., between these processors through internet. Within this general schedulability problem we mix probabilistic and non-probabilistic approaches;

2. the validation of the proposed framework on a multicore drone case study. By combining both multicore systems and distributed systems we have the challenging objective to propose time predictable platforms for drones, inspired by avionics design.

4 Application domains

4.1 Avionics

The time critical solutions in this context are based on temporal and spatial isolation of the programs and the understanding of multicore interferences is crucial. Our contributions belong mainly to the solutions space for the objective [O1] identified previously.

4.2 Railway

The time critical solutions in this context concern both the proposition of an appropriate scheduler and associated schedulability analyses. Our contributions belong to the solutions space of problems dealt within objectives [O1] and [O2] identified previously.

4.3 Autonomous cars

Autonomous cars - the time critical solutions in this context concern the interaction between programs executed on multicore processors and messages transmitted through wireless communication channels. Our contributions belong to the solutions space of all three classes of problems dealt within all three Kopernic objectives identified previously.

4.4 Drones

As it is the case of autonomous cars, there is an interaction between programs and messages, suggesting that our contributions in this context belong to the solutions space of all three classes of problems dealt within the objectives identified previously.

5 Social and environmental responsibility

5.1 Impact of research results

The Kopernic members provide theoretical means to improve the processor utilization. Such gain is estimated within 30% to 60% utilization gain for existing architectures or energy consumption for new architectures by decreasing the number of necessary cores.

6 Highlights of the year

The Kopernic team has hosted the first virtual edition of RTNS2020. Preparing such virtual edition under a very short notice has required an important synchronisation effort with our administrative colleagues that we thank here.

The EVT_Kopernic has been transferred under an exclusive licence from Inria to StatInf, as well as two Inria patents on the validation of statistical WCET estimation.

7 New software and platforms

7.1 New software

7.1.1 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.

URL: http://www.syndex.org

Contact: Yves Sorel

Participant: Yves Sorel

7.1.2 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.

URL: http://www.statinf.fr

Authors: Liliana Cucu, Adriana Gogonel

Contacts: Liliana Cucu, Adriana Gogonel

Participants: Adriana Gogonel, Liliana Cucu

8 New results

8.1 Worst case execution time estimation of a program

We consider WCET statistical estimators that are based on the utilization of the Extreme Value Theory [24]. Compared to existing methods [38], our results require the execution of the program under study on the targeted processor or at least a cycle-accurate simulator of this processor. The originality of considering such WCET statistical estimators consists in the proposition of a black box solution with respect to the program structure. Such solution is obtained by (i) comparing different Generalized Extreme Values estimators [12] and (ii) separating the impact of the processor features from those of the program structure and of the execution environment, as variability factors for the CPS time properties.

Understanding the impact of the processor features and of the program structure is widely studied in the static analysis literature. In our context, we take into account also the execution environment and one may start by identifying this environment with respect to its impact on the paths of a program, which are, in our case, exposed by clusters of execution times. In Figure 3, the horizontal axis represents the order of execution, while the vertical axis the execution time values. Once the paths identified, identically distributed tests indicate if each path is sufficiently observed, while independence tests are used to identify the input variables that do have an impact on the execution time.

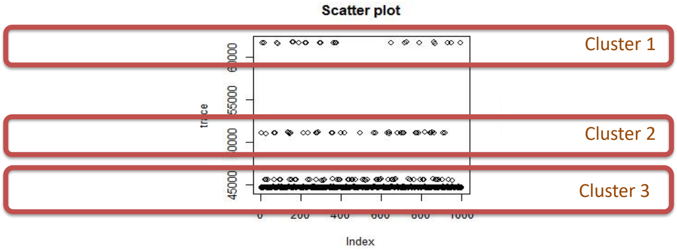


Figure 3: Example of clusters of execution times of a program

Last, but not least, execution scenarios for the identified input variables are fixed and they allow to observe the weight of each variability factor. In Figure 4, the horizontal axis represents the order of execution, while the vertical axis the execution time values. The lowest curve (blue) represents the values of an input variable, while the other curves are obtained by activating a new variability factor for exactly the same execution scenario, i.e., variation of the input variables and of the processor features. For instance, the lowest (yellow style) are obtained when the program is executed alone on one core and the highest curves are obtained for executions with all cores active, two activated levels of caches, etc.

These preliminary results [10] confirm that the input variables and the cache memories are the most important variability factors and, most importantly, that **the impact of some variability factors are composable**. As dependencies are related to the order of execution between consecutive instances of a program, we exploit them to understand the inter-execution relations. To illustrate our preliminary results, we use execution time measurements made on programs performing the PX4-RT autopilot. To our best knowledge there are no equivalent results because the authors of existing work had never access to real programs of an industrial application.

Our first results confirm the existence, for each program, of execution modes [13]. For example in the case of one program of the drone autopilot, we can identify three modes as three concentrations of response times, shown in the left part of Figure 5. The response time of a program corresponds to its execution time when the operating system is taken into account. The modes are estimated with statistical methods by the red curve shown in the right part of Figure 5.

In order to restrain our WCET estimation problem with respect to the program structure, we propose a new modeling semantic allowing the characterization of programs that are "time predictable". Indeed, the problem of time unpredictability [15] increases the complexity of the WCET estimation. We consider programs executed on general purpose processors that are not predictable. The time unpredictability of a program is underlined by an important execution time variability.

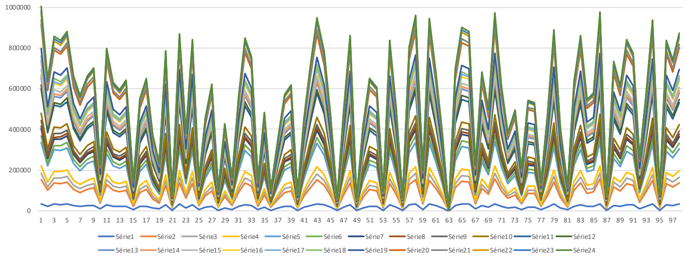
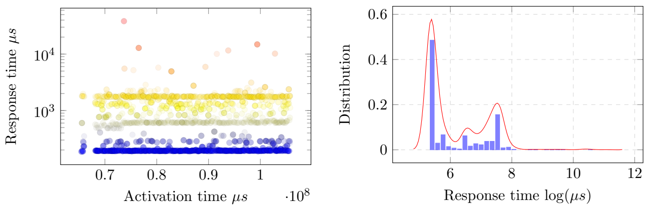


Figure 4: Impact of AURIX Tricore features on the execution time of a program containing a loop



(a) log-Response times against activation times

(b) Distribution of log-response times

Figure 5: Evolution over time of response times for one program of the PX4-RT drone autopilot

Compared to static analysis-based WCET estimation methods, that mainly estimate the WCET of a program on a processor, our goal is to propose rules for the design of time predictable programs with respect to a multicore processor, while reducing the design time. The first step consists in identifying the different time unpredictability factors of a program (internal and external architecture, the structure of the program, the scheduling algorithm used by the operating system executing the program) and then propose a formalization of the relations between time unpredictability factors and the execution time of a program. This formalization could strengthen the statistical analysis proposed by the measurement-based analyses presented above. The second step consists in deducing from the first step some programming rules reducing and controlling the time unpredictability. Our preliminary work consists in measuring the impact of the internal architecture on execution times of programs executed on a multicore processor based on ARM Cortex-A53 widely used in embedded devices. Presently, these programs are executed without using an operating system (bare-metal) to focus on the impact of the internal architecture. The internal architecture of the ARM Cortex-A53 processor has been studied in depth and we proposed simple programs allowing to deduce first relations between the execution times of these programs and the internal architecture of the processor.

8.2 Multicore processor graph tasks scheduling

Due to widespread of multicore processors on embedded and real-time systems, we concentrate our work on the study of the schedulability of real-time tasks with precedence constraints on such processors.

As stressed in the previous sections, the utilisation of multicore processors is the current trend of the CPS industry, while model-driven approaches are considered. Within such models, the functional description of the cyber part of the CPS, is performed, often, as a graph of dependent functions, e.g., a block diagram of functions in Simulink, the most widely used modeling/simulation tool in industry. Of course, a program is associated to every function. Since the graph of dependent programs becomes a set of dependent tasks when real-time constraints must be taken into account, we are facing the problem of verifying the schedulability of such dependent task sets when it is executed on a multicore processor.

Directed Acyclic Graphs (DAG) are widely used to model different types of dependent task sets. The typical model consists of a set of independent tasks where every task is described by a DAG of dependent sub-tasks with the same period inherited from the period of each task [17]. In such DAG, the sub-tasks are vertices and edges are dependencies between sub-tasks. This model is well suited to represent, for example, the engine controller of a car described with Simulink. The multicore schedulability analysis may be of two types, global or partitionned. To reduce interference and interactions between sub-tasks, we focus on partitioned scheduling where each sub-task is assigned to a given core [23].

Our preliminary results [8, 9, 14] make use of the potential parallelism between the sub-tasks to decrease the response time of each DAG. The schedulability analysis performs a partition of the sub-tasks on the cores and for every core performs a schedulability analysis. For instance, in Figure 6, each of the two tasks is described by a DAG, where identical colour indicates that these sub-tasks are executed on the same core.

Furthermore, in order to reduce the pessimism and its associated over-dimensioning of the hardware architectures, the proposed schedulability analysis is probabilistic considering the variability of execution times of tasks, that is, several execution times are associated to each task with a probability to occur [18]. This multicore probabilistic schedulability analysis is based on fixed-point equations instead of mixed integer linear programming (MILP) formulation used by existing results. Indeed, the MILP formulation

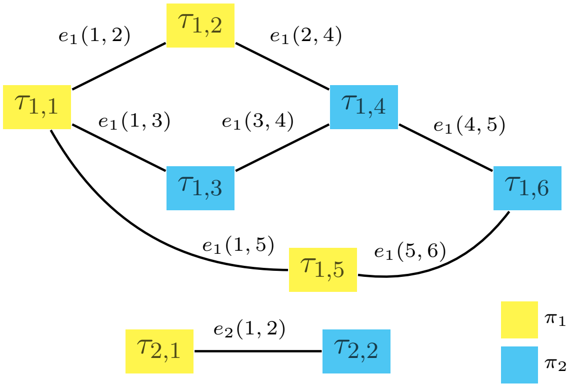


Figure 6: Example of DAG task

has an important complexity that probabilistic analyses cannot afford. We assign priorities at the sub-task level to define the execution order between different vertices from the same graph, which reduces the response time of the entire DAG task. Our priority assignment is performing well for both non-probabilistic and for probabilistic bounds on the execution times of sub-tasks. It performs significantly better than existing work, when compared to the case of non-probabilistic WCET. This is due to a better utilization of the possible parallelism in a DAG structure that decreases the response time. We extend the previous partitioning algorithm by considering the communication times corresponding to dependencies between sub-tasks when they are assigned to different cores. Our solution improves significantly the schedulability ratio compared to existing work. The current limitation of our first results is the unique period for all sub-tasks of a task and, to overcome it, we need the generalize the DAG model to the tasks. This generalization leads to a unique DAG where vertices are tasks that may have different periods and edges are describing dependencies between tasks. Such general model is more suited to new model-driven models used in industry for designing complex CPS, e.g., an autonomous vehicle system, merging inside each vehicle numerous control loops, some of them including artificial intelligence algorithms, and navigation planning algorithms involving communication between vehicles, and between vehicles and infrastructures.

8.3 Data-oriented scheduling approaches

Following our existing results confirmed by a journal publication [7], we consider the scheduling problem of tasks using an inter-task communication model based on a circular buffer, easing the data consistency between tasks [36, 35]. The tasks are scheduled on one processor by a fixed priority preemptive scheduling algorithm and they have implicit deadlines. Previously, we have provided a formal method calculating the optimal size for each of the buffers while ensuring data consistency and an analytical characterization of the temporal validity and reachability properties of the data flowing in between communicating tasks. These two properties are characterized by considering both tasks execution and data propagation orders. Moreover, we propose protocols to ensure such properties highly depend on the considered communication model (shared registers or large buffers) and the data access policy (directly or via local copies). In order to overcome this limitation, we provide in [11] the means for managing the FIFO buffers to guarantee these data properties in a way that communication dependencies do not impact the tasks system scheduling order. We do so while considering the communication model presented in [36]. We provide, also, an algorithm implementing the *last reader tags mechanism* together with the corresponding data temporal matching.

9 Bilateral contracts and grants with industry

9.1 CIFRE Grant funded by StatInf

A CIFRE agreement between the Kopernic team and the start-up StatInf has started on October 1st, 2020. Its funding is related to study of WCET models taking into account the energy consumption according to the fourth research objective of our team.

10 Partnerships and cooperations

10.1 International initiatives

10.1.1 Inria associate team not involved in an IIL

KEPLER

Title: Probabilistic foundations for time, a **KE**y concept for cyber-**P**hysica**L** systems c**ER**tification

Duration: 2020 - 20223

Coordinator: Liliana Cucu

Partners:

- STER, Universidade Federal da Bahia (Brazil)

Inria contact: Liliana Cucu

Summary: Today the term of cyber-physical systems (CPSs) refers to a new generation of systems integrating computational and physical capabilities to interact with humans. A defibrillator, a mobile phone, a car or an aircraft, they all are CPSs. Beside constraints like power consumption, security, size and weight, CPSs may have cyber components required to fulfill their functions within a limited time interval, property a.k.a safety. This expectation arrives simultaneously with the need of implementing new classes of algorithms, e.g. deep learning techniques, requiring the utilization of important computing and memory resources. These ressources may be found on multicore processors, known for increasing the execution time variability of programs. Therefore, ensuring time predictability on multicore processors is our identified challenge. The Kepler project faces this challenge by developing new mechanisms and techniques for supporting CPS applications on multicore processors, focusing on scheduling and timing analysis, for which probabilistic guarantees should be provided.

10.2 European initiatives

10.2.1 Collaborations in European programs, except FP7 and H2020

The Kopernic members have joind the COST initiative Cerciras (see https://www.cost.eu/actions/CA19135/#tabs|Name:parties).

10.2.2 Collaborations with major European organizations

University of York The Kopernic members have tight collaborations with the members of the Real-Time System Group (UK) on more rigorous principles that are needed for a good understanding of statistics and probability theory on the design of real-time systems.

10.3 National initiatives

10.3.1 FUI

CEOS The CEOS project has started on May 2017 and it is expected to finish by February 2021. Partners of the project are: ADCIS, ALERION, Aeroport de Caen, 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.

11 Dissemination

11.1 Promoting scientific activities

11.1.1 Scientific events: organisation

General chair, scientific chair

- Liliana Cucu-Grosjean and Roberto Medina have been the general chairs of the International Conference on Real-Time and Networked Systems (RTNS2020).

Chair of conference program committees

- Liliana Cucu-Grosjean has been the Track co-chair of DATE2020 (the real-time systems track) and PC co-chair of IEEE RTAS 2020.

Member of the conference program committees

- Liliana Cucu-Grosjean has been a PC member for RTAS2021, ETFA2020 and WFCS2020.
- Yves Sorel has been a PC member for DASIP 2020.
- Yasmina Abdeddaïm has been a PC member of RTNS2020.

Reviewer

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

11.1.2 Scientific expertise

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

11.1.3 Invited talks

- Liliana Cucu-Grosjean has been invited to give a talk at IRT System X on February 2020. The talk is available at https://www.youtube.com/watch?v=YLmdNFpehsQ.

11.1.4 Leadership within the scientific community

- Liliana Cucu-Grosjean is a IEEE TCRTS member (2016-2021) and a member of the RTNS, RTSOPS and WMC steering committes.

11.1.5 Research administration

- Liliana Cucu-Grosjean is the co-chair of the Equal Opportunities and Gender Equality committee of Inria.
- Liliana Cucu-Grosjean is an elected member of Inria scientific board (CS) and Inria CRCN disciplinary commision (CAP).
- Yves Sorel is chair of the CUMI Paris center commission.
- Yves Sorel is member of the CDT Paris center commission.

11.2 Teaching - Supervision - Juries

11.2.1 Teaching

- Master: Yves Sorel, Optimization of distributed real-time embedded systems, 38H, M2, University of Paris Sud, France.
- Master: Yves Sorel, Safe design of reactive systems, 18H, M2, ESIEE Engineering School, Noisy-Le-Grand, France.
- Master: Liliana Cucu-Grosjean, Software Engineering, 30H, ESIEE, Noisy-le-Grand, France.

11.2.2 Supervision

- PhD in progress: Marwan WEHAIBA EL KHAZEN, Statistical models for optimizing the energy consumption of cyber-physical systems, UMPC, started on October 2020, supervised by Liliana Cucu-Grosjean and Adriana Gogonel (StatInf).
- PhD in progress: Kevin Zagalo, Statistical predictability of cyber-physical systems, UMPC, started on October 2019, supervised by Liliana Cucu and Prof. Avner Bar-Hen (CNAM).
- PhD in progress: Evariste Ntaryamira, Analysis of embedded systems with time and security constraints, UPMC, started on May 2017, supervised by Liliana Cucu and Cristian Maxim (IRT SystemX). His PhD defense is planned for April 14th, 2021.
- defended PhD: Slim Ben-Amor, Schedulability analysis of probabilistic real-time tasks under end to end constraints, UPMC, started on November 2016, supervised by Liliana Cucu. This thesis has been defended on December 14th, 2020.

11.2.3 Juries

- Liliana Cucu-Grosjean was a reviewer for the PhD theses of Matteo Bertolino (Telecom Paris), Marco Pagani (University of Lille and Scuola Superiore Sant'Anna) and Stephan Plassart (University of Grenoble).

11.3 Popularization

- Liliana Cucu-Grosjean and Adriana Gogonel have participated to the Agoranov action on promoting start-ups and female investissement in schools, the video is available at https://www.youtube.com/watch?v=FlUvnXSQqJs.

12 Scientific production

12.1 Major publications

[1] L. Cucu-Grosjean and A. Gogonel. 'Simulation Device'. FR2016/050504 (France). Mar. 2016. URL: https://hal.archives-ouvertes.fr/hal-01666599.

[2] L. Cucu-Grosjean, L. Santinelli, M. Houston, C. Lo, T. Vardanega, L. Kosmidis, J. Abella, E. Mezzetti, E. Quiñones and F. J. Cazorla. 'Measurement-Based Probabilistic Timing Analysis for Multi-path Programs'. In: *the 24th Euromicro Conference on Real-Time Systems, ECRTS*. 2012, pp. 91–101.

[3] A. Gogonel and L. Cucu-Grosjean. 'Dispositif de caractérisation et/ou de modélisation de temps d'exécution pire-cas'. 1000408053 (France). June 2017. URL: https://hal.archives-ouvertes.fr/hal-01666535.

[4] T. Kloda, A. Bertout and Y. Sorel. 'Latency analysis for data chains of real-time periodic tasks'. In: *the 23rd IEEE International Conference on Emerging Technologies and Factory Automation, ETFA'18*. Sept. 2018.

[5] C. Maxim, A. Gogonel, I. M. Asavoae, M. Asavoae and L. Cucu-Grosjean. 'Reproducibility and representativity: mandatory properties for the compositionality of measurement-based WCET estimation approaches'. In: *SIGBED Review* 14.3 (2017), pp. 24–31.

[6] S. E. Saidi, N. Pernet and Y. Sorel. 'Scheduling Real-time HiL Co-simulation of Cyber-Physical Systems on Multi-core Architectures'. In: *the 24th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*. Aug. 2018.

12.2 Publications of the year

International journals

[7] T. Kloda, A. Bertout and Y. Sorel. 'Latency upper bound for data chains of real-time periodic tasks'. In: *Journal of Systems Architecture* 109 (Oct. 2020). DOI: 10.1016/j.sysarc.2020.101824. URL: https://hal.inria.fr/hal-03111542.

International peer-reviewed conferences

[8] S. Ben-Amor, L. Cucu-Grosjean, M. Mezouak and Y. Sorel. 'Probabilistic Schedulability Analysis for Precedence Constrained Tasks on Partitioned Multi-core'. In: 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). Vienna, Austria, 8th Sept. 2020, pp. 345–352. DOI: 10.1109/ETFA46521.2020.9211973. URL: https://hal.inria.fr/hal-03119185.

[9] S. Ben-Amor, L. Cucu-Grosjean, M. Mezouak and Y. Sorel. 'Probabilistic Schedulability Analysis for Real-time Tasks with Precedence Constraints on Partitioned Multi-core'. In: ISORC 2020 - IEEE 23rd International Symposium on Real-Time Distributed Computing. Nashville / Virtual, United States, 19th May 2020, pp. 142–143. DOI: 10.1109/ISORC49007.2020.00029. URL: https://hal.inria.fr/hal-03119186.

[10] K. Kougblenou, R. Bennour, A. Gogonel and L. Cucu-Grosjean. 'Work in Progress : Towards representative measurement protocols'. In: RTSS 2020 - 41st IEEE Real Time Systems Symposium. Dallas / Virtual, United States, 1st Dec. 2020. URL: https://hal.inria.fr/hal-03124586.

[11] E. Ntaryamira, C. Maxim, T. Niyonsaba and L. Cucu-Grosjean. 'An efficient FIFO buffer management to ensure task level and effect-chain level data properties'. In: ICESS 2020 - IEEE International Conference on Embedded Software and Systems. Shanghai / Virtual, China, 10th Dec. 2020. DOI: 10.1109/ICESS49830.2020.9301518. URL: https://hal.archives-ouvertes.fr/hal-03097920.

[12] M. Wehaiba El Khazen, A. Gogonel and L. Cucu-Grosjean. 'Work in Progress Lessons learnt from creating Extreme Value Libraries in Python'. In: RTSS 2020 - 41st IEEE Real Time Systems Symposium. Dallas / Virtual, United States, 1st Dec. 2020. URL: https://hal.inria.fr/hal-03124588.

[13] K. Zagalo, L. Cucu-Grosjean and A. Bar-Hen. 'Identification of execution modes for real-time systems using cluster analysis'. In: 25th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA. Vienne, Austria, 9th Sept. 2020. DOI: 10.1109/ETFA46521.2020.9211983. URL: https://hal.inria.fr/hal-02938202.

Doctoral dissertations and habilitation theses

[14] S. Ben-Amor. 'Scheduling of Dependent Tasks with Probabilistic Execution Times on Multi-core Processors.' Sorbonne Universite, 14th Dec. 2020. URL: https://hal.inria.fr/tel-03119187.

12.3 Cited publications

[15] P. Axer, R. Ernst, H. Falk, A. Girault, D. Grund, N. Guan, B. Jonsson, P. Marwedel, J. Reineke, C. Rochange, M. Sebastian, R. von Hanxleden, R. Wilhelm and W. Yi. 'Building timing predictable embedded systems'. In: *ACM Trans. Embedded Comput. Syst.* 13.4 (2014), 82:1–82:37.

[16] R. Baheti and H. Gill. *Cyber-physical systems.* IEEE, 2011.

[17] S. Baruah, V. Bonifaci, A. Marchetti-Spaccamela, L. Stougie and A. Wiese. 'A Generalized Parallel Task Model for Recurrent Real-time Processes'. In: *2012 IEEE 33rd Real-Time Systems Symposium (RTSS).* 2012, pp. 63–72.

[18] S. Ben-Amor, D. Maxim and L. Cucu-Grosjean. 'Schedulability analysis of dependent probabilistic real-time tasks'. In: *the 24th International Conference on Real-Time Networks and Systems (RTNS).* 2016.

[19] T. Bourke, J.-L. Colaço, B. Pagano, C. Pasteur and M. Pouzet. 'A Synchronous-Based Code Generator for Explicit Hybrid Systems Languages'. In: *Compiler Construction - 24th International Conference, CC, Joint with ETAPS.* 2015, pp. 69–88.

[20] L. Cucu. 'Preliminary results for introducing dependent random variables in stochastic feasiblity analysis on CAN'. In: *the WIP session of the 7th IEEE International Workshop on Factory Communication Systems (WFCS).* 2008.

[21] R. I. Davis and L. Cucu-Grosjean. 'A Survey of Probabilistic Timing Analysis Techniques for Real-Time Systems'. In: *LITES* 6.1 (2019), 03:1–03:60.

[22] H. Falk, S. Altmeyer, P. Hellinckx, B. Lisper, W. Puffitsch, C. Rochange, M. Schoeberl, R. B. Sorensen, P. Wägemann and S. Wegener. 'TACLeBench: A Benchmark Collection to Support Worst-Case Execution Time Research'. In: *16th International Workshop on Worst-Case Execution Time Analysis (WCET).* Vol. 55. OASICS. 2016, 2:1–2:10.

[23] J. Fonseca, G. Nelissen, V. Nelis and L. Pinho. 'Response time analysis of sporadic DAG tasks under partitioned scheduling'. In: *11th IEEE Symposium on Industrial Embedded Systems (SIES).* May 2016, pp. 1–10.

[24] S. J. Gil, I. Bate, G. Lima, L. Santinelli, A. Gogonel and L. Cucu-Grosjean. 'Open Challenges for Probabilistic Measurement-Based Worst-Case Execution Time'. In: *Embedded Systems Letters* 9.3 (2017), pp. 69–72.

[25] J. Gustafsson, A. Betts, A. Ermedahl and B. Lisper. 'The Mälardalen WCET Benchmarks: Past, Present And Future'. In: *10th International Workshop on Worst-Case Execution Time Analysis (WCET).* Vol. 15. OASICS. 2010, pp. 136–146.

[26] E. Lee. 'Computing Needs Time'. In: *Communications of ACM* 52.5 (2009).

[27] E. Lee and S. Seshia. *Introduction to embedded systems - a cyber-physical systems approach.* MIT Press, 2017.

[28] J. Lehoczky. 'Real-Time Queueing Theory'. In: *the 10th IEEE Real-Time Systems Symposium (RTSS).* 1996.

[29] S. B. M. Bertogna and G. Buttazzo. *Multiprocessor Scheduling for Real-Time Systems.* Springer, 2015.

[30] D. Maxim, O. Buffet, L. Santinelli, L. Cucu-Grosjean and R. I. Davis. 'Optimal Priority Assignment Algorithms for Probabilistic Real-Time Systems'. In: *the 19th International Conference on Real-Time and Network Systems (RTNS)*. 2011.

[31] D. Maxim and L. Cucu-Grosjean. 'Response Time Analysis for Fixed-Priority Tasks with Multiple Probabilistic Parameters'. In: *the IEEE Real-Time Systems Symposium (RTSS)*. 2013.

[32] R. Medina and L. Cucu-Grosjean. 'Work-in-Progress: Probabilistic System-Wide DVFS for Real-Time Embedded Systems'. In: *IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2019, pp. 508–511.

[33] F. Ndoye and Y. Sorel. 'Monoprocessor Real-Time Scheduling of Data Dependent Tasks with Exact Preemption Cost for Embedded Systems'. In: *the 16th IEEE International Conference on Computational Science and Engieering (CSE)*. 2013.

[34] F. Nemer, H. Cassé, P. Sainrat, J. P. Bahsoun and M. D. Michiel. 'PapaBench: a Free Real-Time Benchmark'. In: *6th Intl. Workshop on Worst-Case Execution Time (WCET) Analysis*. Vol. 4. OASICS. 2006.

[35] E. Ntaryamira, C. Maxim and L. Cucu-Grosjean. 'Data consistency and temporal validity under the circular buffer communication paradigm'. In: *RACS '19 - Conference on Research in Adaptive and Convergent Systems*. Chongqing, China: ACM Press, 2019, pp. 51–56. URL: https://hal.inria.fr/hal-02409672.

[36] E. Ntaryamira, C. Maxim and L. Cucu-Grosjean. 'The temporal correlation of data in a multirate system'. In: *RTNS'2019 - 27th International Conference on Real-Time Networks and Systems*. Toulouse, France, 2019. URL: https://hal.archives-ouvertes.fr/hal-02362858.

[37] S. E. Saidi, N. Pernet and Y. Sorel. 'Automatic Parallelization of Multi-Rate FMI-based Co-Simulation On Multi-core'. In: *the Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium*. 2017.

[38] R. Wilhelm, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley, G.Bernat, C. Ferdinand, R.Heckmann, T. Mitra, F. Mueller, I. Puaut, P. Puschner, G. Staschulat and P. Stenströem. 'The worst-case execution time problem: overview of methods and survey of tools'. In: *Trans. on Embedded Computing Systems* 7.3 (2008), pp. 1–53.