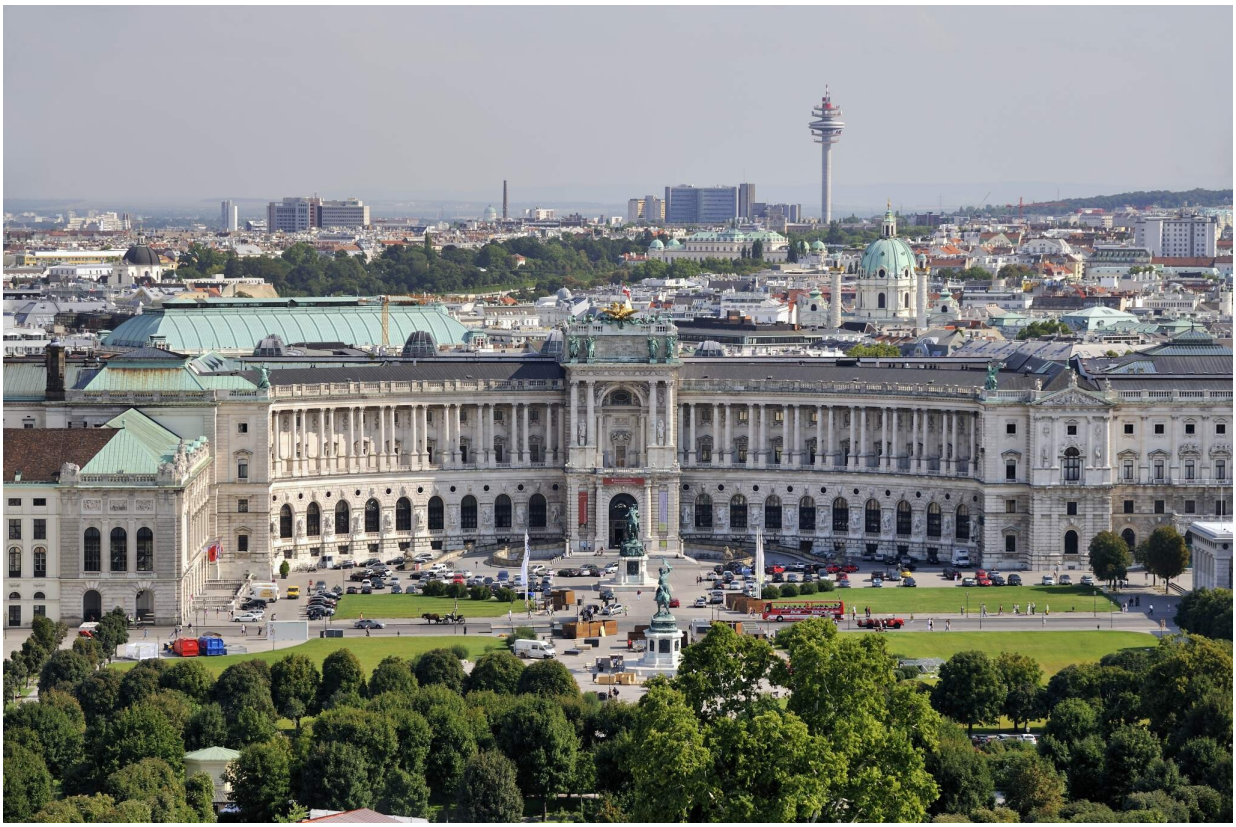
22nd International Conference on Reliable Software Technologies

Ada-Europe 2017



12-16 June 2017, Vienna, Austria

Abstracts Regular Papers

http://www.ada-europe.org/conference2017

In cooperation with

acm In-Cooperation SIGAda SIGPLAN

Ada Resource Association

Runtimes

Evaluating MSRP and MrsP with the Multiprocessor Ravenscar Profile

Jorge Garrido, Juan Zamorano, Alejandro Alonso, and Juan Antonio de La Puente

Abstract. One of the main challenges of developing real-time systems with Ada on multiprocessor platforms is finding an appropriate scheduling policy and locking policy for shared objects. Some modifications of the standard `Ceiling_Locking` policy have been proposed for multiprocessor architectures, among which MSRP and MrsP have raised most interest. In this paper the possible uses of both policies in full Ada and Ravenscar programs are explored. To this purpose, classical response time analysis is extended in the paper to deal with heterogeneous access costs in multiprocessor systems. A case study has been used to validate the approach, and an extensive test bench for comparing MSRP and MrsP has been run in order to compare the schedulability properties of both methods. The conclusion is that, although MrsP provides a better overall performance, in many practical situations the simpler MSRP protocol provides comparable results when considering heterogeneous access costs, while being compatible with the Ravenscar restrictions.

Ravenscar-EDF: Comparative Benchmarking of an EDF Variant of a Ravenscar Run-time

Paolo Carletto and Tullio Vardanega

Abstract. Subsequent to the seminal work of Liu and Layland in 1973, researchers and practitioners alike discussed which online scheduling algorithm was to be preferred between FPS and EDF. Results published in 2005 sustained the superiority of EDF, already proven in theory, also from an implementation perspective. In this work, we aim at digging deeper into the roots of those results. To this end, we took the Ravenscar run-time, which includes a lean, fast and efficient implementation of an FPS scheduler, combined with its IPCP locking policy companion, and developed a variant of it that implements EDF scheduling coupled with DFP locking. In this manner, we were able to transparently attach those two run-time variants to a vast suite of application benchmarks, which we used to perform an extensive quantitative comparison between those two run-times, achieving the goal of digging deeper into where one prevails on the other.

Safety and Security

Sanitizing Sensitive Data: How to Get it Right (or at Least Less Wrong...)

Roderick Chapman

Abstract. Coding standards and guidance for secure programming call for sensitive data to be sanitized before being de-allocated. This paper considers what this really means in technical terms, why it is actually rather difficult to achieve, and how such a requirement can be realistically implemented and verified, concentrating of the facilities offered by Ada and SPARK. The paper closes with a proposed policy and coding standard that can be applied and adapted to other projects.

Enforcing Timeliness and Safety in Mission-Critical Systems

António Casimiro, Inês Gouveia, and José Rufino

Abstract. Advances in sensor, microprocessor and communication technologies have been fostering new applications of cyber-physical systems, often involving complex interactions between distributed autonomous components and the operation in harsh or uncertain contexts. This has led to new concerns regarding performance, safety and security, while ensuring timeliness requirements are met. For instance, in the automotive domain, the introduction of vehicular functions in intelligent autonomous vehicles has been strongly limited by the difficulty in satisfying these non-functional requirements altogether, for arbitrary operational contexts. To conciliate uncertainty with the required predictability, hybrid system architectures have been proposed, which separate the system in two parts: one that behaves in a best-effort way, depending on the context, and another that behaves as predictably as needed, providing critical services for a safe and secure operation. In this paper we address the problem of how to verify in run-time the correct provisioning of critical functions in such hybrid architectures. We consider, in particular, the KARYON hybrid architecture and its safety kernel. We also consider a hardware-based non-intrusive run-time verification approach, describing how it is applied to verify safety kernel functions. We finally illustrate how the combination would be deployed in a vehicular application context.

Timing Verification

Supporting Nested Resources in MrsP

Jorge Garrido, Shuai Zhao, Alan Burns, and Andy Wellings

Abstract. The original MrsP proposal presented a new multiprocessor resource sharing protocol based on the properties and behaviour of the Priority Ceiling Protocol, supported by a novel helping mechanism. While this approach proved to be as simple and elegant as the single processor protocol, the implications with regard to nested resources was identified as requiring further clarification. In this work we present a complete approach to nested resources behaviour and analysis for the MrsP protocol.

Predicting Worst-Case Execution Time Trends in Long-Lived Real-Time Systems

Xiaotian Dai and Alan Burns

Abstract. In some long-lived real-time systems, it is not uncommon to see that execution times of some tasks may exhibit trends. For hard and firm real-time systems, it is important to ensure these trends will not jeopardise the system. In this paper, we first introduce the notion of dynamic worst-case execution time, which is a new perspective that could help the system to predict potential timing failures and optimise resource allocation. We then had a comprehensive review of trend prediction methods. In the evaluation, we made a comparative study of dWCET trend prediction. Four prediction methods, combined with three data selection processes, were applied in an evaluation framework. The result shows the importance of applying data pre-processing and suggests that non-parametric estimators perform better than parametric methods.

MC2: Multicore and Cache Analysis via Deterministic and Probabilistic Jitter Bounding

Enrique Díaz Roque, Mikel Fernández, Leonidas Kosmidis, Enrico Mezzetti, Carles Hernandez, Jaume Abella, and Francisco J Cazorla

Abstract. In critical domains, reliable software execution increasingly covers aspects related to timing. This is due to the advent of high-performance (complex) hardware used to provide the rising levels of guaranteed performance needed in those domains. Caches and multicores are two of the hardware features that can significantly reduce WCET estimates, yet they pose new challenges on current-practice measurement-based timing analysis (MBTA) approaches. In this paper we propose MC2, a technique for multilevel-cache multicores that reliably handles both the jitter (variability in execution time) generated by caches and the contention in the access to shared resources by combining deterministic and probabilistic jittery-bounding approaches. We evaluate MC2 on a COTS quad-core LEON-based board and our initial results show how it effectively captures cache and multicore contention in pWCET estimates with respect to actual observed vaules.

Programming Models

Lock Elision for Protected Objects Using Intel Transactional Synchronization Extensions

Seongho Jeong, Shinhyung Yang, and Bernd Burgstaller

Abstract. Lock elision is a technique to replace coarse-grained locks by optimistic concurrent execution. In this paper, we introduce lock elision for protected objects (POs) in Ada. We employ Intel Transactional Synchronization Extensions (TSX) as the underlying hardware transactional memory facility. With TSX, a processor can detect dynamically whether tasks need to serialize through critical sections protected by locks. We adapt the GNU Ada run-time library (GNARL) to elide locks transparently from protected functions and procedures. We critically evaluate opportunities and difficulties of lock elision with protected entries. We demonstrate that lock elision can achieve significant performance improvements for a selection of three synthetic and one real-world benchmark. We show the scalability of our approach for up to 44 cores of a two-CPU, 44-core Intel E5-2699 v4 system.

An Executable Semantics for Synchronous Task Graphs: From SDRT to Ada

Morteza Mohaqeqi, Syed Md Jakaria Abdullah, and Wang Yi

Abstract. We study a graph-based real-time task model in which inter-task synchronization can be specified through a rendezvous mechanism. Previously, efficient methods have been proposed for timing analysis of the corresponding task sets. In this paper, we first formally specify an operational semantics for the model. Next, we describe a method for Ada code generation for a set of such task graphs. We also specify extensions of the approach to cover a notion of broadcasting, as well as global inter-release separation time of real-time jobs. We have implemented the proposed method in a graphical tool which facilitates a model-based design and implementation of real-time software.

RxAda: An Ada implementation of the ReactiveX API

Alejandro R. Mosteo

Abstract. The ReactiveX API, also known as the Reactive Extensions in the .NET world, is a recently popularized reactive programming framework for asynchronous, event-based, multi-threaded programming. Presented by its proponents as a solid tool for applications requiring a simple yet powerful approach to event-driven systems, it has seen favorable adoption in many popular languages. Although Ada has been long-favored by powerful tasking capabilities that reduce the need for additional multi-threading support, the reactive approach has properties that are well-suited to the safety and maintainability culture predominant in the Ada world, such as complexity reduction, race-condition and deadlock avoidance and enhanced maintainability by means of concise and well-defined information flows. This work presents the design for a ReactiveX Ada implementation that aims to balance desirable library properties such as compile-time checks, reasonable user-required generic instantiations, and a shallow learning curve for both library clients and maintainers. The Ada programmer can henceforth benefit from the abundant documentation existing for the language-agnostic ReactiveX approach without stepping out of the Ada tool chain.

The Future of Safety-Minded Languages

A New Ravenscar-Based profile

Patrick Rogers, Jose Ruiz, Tristan Gingold, and Patrick Bernardi

Abstract. We describe a new Ada language profile based directly upon the Ravenscar profile, intended to add expressive power for applications in the real-time and embedded systems domains. The new profile enhancements result primarily from the removal of certain Ravenscar restrictions but new capabilities are added. We provide the motivation and requirements for such a profile, the corresponding language changes, and performance analyses.

OpenMP Tasking Model for Ada: Safety and Correctness

Sara Royuela, Xavier Martorell, Eduardo Quiñones, and Luis Miguel Pinho

Abstract. The safety-critical real-time embedded domain increasingly demands the use of parallel architectures to fulfill performance requirements. Such architectures require the use of parallel programming models to exploit the underlying parallelism. In this paper, we evaluate the applicability of OpenMP, a widespread parallel programming model, into Ada, a language widely used in the safety-critical domain. OpenMP, born in the 90's out of the need for standardizing the different vendor specific directives related to parallelism, has successfully emerged as the de facto standard for shared-memory parallel programming, with advance support for distributed memory as well.

Concretely, this paper demonstrates that applying the OpenMP tasking model to exploit fine-grained parallelism within Ada tasks does not impact on programs safeness and correctness, which is vital in the environments where Ada is mostly used. Moreover, we compare the OpenMP tasking model with the proposal of Ada extensions to define parallel blocks, parallel loops and reductions. Overall, we conclude that OpenMP can be safely used in such environments, being a very promising approach to exploit fine-grain parallelism in Ada tasks.

Mixed Criticality

Migrating Mixed Criticality Tasks Within a Cyclic Executive Framework

Alan Burns and Sanjoy Baruah

Abstract. In a cyclic executive, a series of frames are executed in sequence; once the series is complete the sequence is repeated. Within each frame, units of computation are executed, again in sequence. In implementing cyclic executives upon multi-core platforms, there is advantage in coordinating the execution of the cores so that frames are released at the same time across all cores. For mixed criticality systems, the requirement for separation would additionally require that, at any time, code of the same criticality should be executing on all cores. In this paper we derive algorithms for constructing such multiprocessor cyclic executives for systems of periodic tasks, when inter-processor migration is permitted.

Directed Acyclic Graph Scheduling for Mixed-Criticality Systems

Roberto Medina, Etienne Borde, and Laurent Pautet

Abstract. Deploying safety-critical systems into constrained embedded platforms is a challenge for developers who must arbitrate between two conflicting objectives: software has to be safe and resources need to be used efficiently. Mixed-criticality (MC) has been proposed to meet a trade-off between these two aspects. Nonetheless, most task models considered in the literature of MC scheduling, do not take into account precedence constraint among tasks. In this paper, we propose a scheduling approach for a model presenting MC tasks and their dependencies as a Directed Acyclic Graph (DAG). We also introduce an evaluation framework for this model, released as an open source software. Evaluation of our scheduling algorithm provides evidence of the difficulty to find correct scheduling for DAGs of MC tasks. Besides, experimentation results provided in this paper show that our scheduling algorithm outperforms existing algorithms for scheduling DAGs of MC tasks.

Software Time Reliability in the Presence of Cache Memories

Suzana Milutinovic, Jaume Abella, Irune Agirre, Mikel Azkarate-Askasua, Enrico Mezzetti, Tullio Vardanega, and Francisco J Cazorla

Abstract. The use of caches challenges measurement-based timing analysis (MBTA) in critical embedded systems. In the presence of caches, the worst-case timing behavior of a system heavily depends on how code and data are laid out in cache. Guaranteeing that test runs capture, and hence MBTA results are representative of, the worst-case conflictive cache layouts, is generally unaffordable for end users. The probabilistic variant of MBTA, MBPTA, exploits randomized caches and relieves the user from the burden of concocting layouts. In exchange, MBPTA requires the user to control the number of runs so that a solid probabilistic argument can be made about having captured the effect of worst-case cache conflicts during analysis. We present a computationally tractable Time-aware Address Conflict (TAC) mechanism that determines whether the impact of conflictive memory layouts is indeed captured in the MBPTA runs and prompts the user for more runs in case it is not.