Ph.D. in Information Technology:

Durelli, Gibilisco and Krstic Final Dissertations

DEIB Conference Room

February 5th, 2016

2.00 pm

First Ph.D. presentation and discussion:

Gianluca DURELLI – XXVIII Cycle

"On the Design of Autonomic Techniques for Runtime Resource Management in Heterogeneous Systems"

Supervisor: Prof. **Marco Santambrogio**

Abstract:

Nowadays, high performance systems are designed to serve rather static workloads with high-performance requirements, although we are moving towards a highly flexible, on-demand computing scenario that is characterized by varying workloads, constituted by diverse applications with different performance requirements and criticality. A promising approach to address the challenges posed by this scenario is to better exploit specialized computing resources integrated in a heterogeneous system architecture such as asymmetric multicore CPUs, specialized graphic co-processors, GPUs, or reconfigurable HW such as FPGAs. Furthermore we can expect that, in the forthcoming years, more and more companies will drive their businesses on the basis of complex data analysis as is testified by the Big Data phenomenon. Under this assumption datacenters will have to face even more unpredictable workloads coming from a variety of contexts and users each one with his own service level objectives. With this unpredictability the datacenter management systems must be able to react to situations unknown and not predictable at design time. To face this challenge the research community started to devise autonomic resource management techniques that improve the standard static resource mechanisms that have usually left the target systems under utilized. Unfortunately, only few solutions have been proposed in the field of heterogeneous systems. This dissertation analyzes this context and proposes a general frameworks for the development and implementation of runtime resource management techniques in the context of heterogeneous systems.

The outcomes of this thesis are the definition of a backbone infrastructure for runtime resource management, accompanied by an high level simulator for fast prototyping of autonomic policies; finally we also propose new controlling techniques that complete and enhance the ones currently in the state of the art.

Second Ph.D. presentation and discussion:

Giovanni Paolo GIBILISCO – XXVIII Cycle

"A Methodology and a Tool for QoS-Oriented Design of Multi-Cloud Applications"

Supervisor: Prof. **Danilo Ardagna**

Abstract:

This work focuses on the support of the development of multi-cloud enabled applications with Quality of Service (QoS) guarantees. It embraces the model driven engineering principles and aims at providing development teams with methodologies and tools to assess the expected QoS of their applications early in the design stages. To do so we adopt and enrich different component based and UML-like modeling technologies like the Palladio Component Model and MODACloudML extending them in order to determine the optimal deployment in multi-cloud environments by introducing a new cloud specific meta-model. The integration of the new meta-model into state of the art modeling tools like Palladio Bench or Modelio allows software architects to use well known modeling approaches and specify a cloud specific deployment for their applications. In order to ease the portability of both the model and the application the meta-model uses three abstraction levels. The Cloud enabled Computation Independent Model (CCIM) allows to describe the application without any reference to specific cloud technologies or providers; the Cloud Provider Independent Model (CPIM) adds the specificity of some cloud technologies introducing concepts like Infrastructure and Platform as a Service (IaaS/PaaS) but still abstracts away the specificity of each particular provider; the Cloud Provider Specific Model (CPSM) adds all the details related to a particular cloud provider and the services offered allowing to automatize the deployment of the application and generate performance models that can be analyzed to assess the expected QoS of the application. High level architectural models of the application are then transformed into a Layered Queuing Network performance model that is analyzed with state of the art solvers like LQNS or LINE in order to derive performance metrics. The result of the evaluation can be used by software architects to refine their design choices. Furthermore, the approach automates the exploration of deployment configurations in order to minimize operational costs of the cloud infrastructure and guarantee application QoS, in terms of availability and response time. In the IaaS context, as an example, the deployment choices analyzed by the tool are the size of instances (e.g. Amazon EC2 m3.xlarge) used to host each application tier and the number of replicas for each hour of the day. The problem of finding the optimal deployment configuration has been analyzed from a mathematical point of view and has been shown to be NP-hard. For this reason a heuristic approach has been proposed to effectively explore the space of possible deployment configurations. The heuristic approach uses a relaxed formulation based on M/G/1 queuing models to derive a promising initial solution that is then refined by means of a two level hybrid heuristic and validated against the LQN performance model. The proposed methodology has been validated by two

industrial case study in the context of the MODAClouds project. A scalability and robustness analysis has also been performed and shows that our heuristic approach allows reductions in the cost of the solution ranging from 39% to 78% with respect to current best practice policies implemented by cloud vendors. The scalability analysis shows that the approach is applicable also to complex scenarios, with the optimized solution of the most complex instance analyzed being found in 36 minutes for a single cloud deployment and in 46 minutes for a multicloud scenario.

Third Ph.D. presentation and discussion:

Srdan KRSTIC– XXVIII Cycle

"Trace Checking of Quantitative Properties"

Supervisor: Prof. **Carlo Ghezzi**

Abstract:

Software engineering has dramatically changed over the past decade and many of the changes have challenged our most basic assumptions about the nature of the software products that we develop. The most important realization is that modern software has a very complex interaction with the environment in which it executes and it is often not safe to assume that the behavior of the environment is stable. Designing software that anticipates changes in the environment makes the software itself exhibit dynamic behavior that can only be observed at run time. This asks for verification techniques that complement design-time approaches and puts forward trace checking as a viable complementary choice for verifying modern software. Trace checking is an automatic procedure for evaluating a formal specification over a trace of recorded events produced by a system after execution. The output of the procedure states whether the system behaves according to its specification.

The goal of this thesis is to develop general and efficient trace checking procedures that support a broad class of quantitative properties. Quantitative properties can be seen as constraints on quantifiable values observed in an execution of a system.

Quantitative properties typically express non-functional requirements, like constraints on resource utilization (e.g., number of computation resources, power consumption, costs), constraints on the runtime characteristics of the environment (e.g., arrival rates, response time), or constraints on the runtime behavior of the system (e.g., timing constraints, QoS, availability, fault tolerance).

The first part of the thesis discusses two algorithms that implement the satisfiability procedure for SOLOIST – a specification language based on metric temporal logic (MTL) used to express quantitative properties. We show how a satisfiability procedure can be used to perform trace checking and apply the proposed approach to an extensive case study in the domain of cloud-based elastic systems.

The second part of the thesis focuses on the problem of distributed trace checking and provides algorithms that rely on existing distributed computation frameworks (like MapReduce and Spark) to efficiently check SOLOIST specifications over very large traces. The thesis also contributes to the state of the art in MTL trace checking by proposing a novel decomposition technique for MTL formulae. This decomposition provides a scalable way of trace checking formulae with large time intervals. Due to known restrictions of the standard point-based MTL semantics we facilitate the decomposition by proposing an alternative semantics for MTL, called lazy semantics. The new semantics is more powerful than point-based semantics and possesses certain properties that allow us to decompose any MTL formula into an equivalent MTL formula with smaller time intervals.