A Hierarchical Approach for Resource Management in Heterogeneous Systems

Michele Zanella*,1, Federico Reghenzani*,1,
Giuseppe Massari*,1, William Fornaciari*,1

* *Dipartimento di Elettronica, Informatica e Bioingegneria, Politecnico di Milano,
Via Ponzio 34/5 20133 Milan, Italy*

ABSTRACT

Heterogeneous architectures are emerging as a dominant trend for HPC, mainly thanks to their high performance-per-watt ratio. Dealing with heterogeneity and task-based applications requires to consider different aspects at both infrastructures level and single node in order to meet power, thermal and performance requirements. Thus, in order to provide an effective and fine-grained management of the available resources, as well as balancing the load by dispatching applications among the different computing nodes, we proposed a hierarchical approach in which different resource managers, running in the nodes, collaborate to reach a multi-objectives optimization.

KEYWORDS: Resource Management, Heterogeneous Architecture, HPC

1 Introduction

In the latest years, High-Performance Computing (HPC) is experiencing a radical push towards Exascale, quickly evolving at the hardware, software and application levels. From the hardware point of view, the energy budget and sustainability may constrain the size and power of HPC centres. Thus, the evolution of HPC hardware and software architectures needs to embrace technologies with high performances and low power consumptions. At this regard, heterogeneous architectures are emerging as a dominant trend for pure performance and, even more, for performance-per-watt, besides they are much more complex to program and manage. From the software side, the spreading of HPC resources around the world combined with innovative delivery modes (such as cloud HPC) is increasing the number of users that can afford HPC computational resources. This leads novel classes of applications to emerge. In this context, applications that are time-critical and Quality of Service (QoS) sensitive, such as financial analytics, online video transcoding and medical imaging require predictable performance, which are at odds with the need to maximize resource usage while minimizing power consumption.

In order to achieve an effective and efficient management of heterogeneous resources we need to tackle different challenges: (a) dealing with task-based applications in which

[1]E-mail: {name.surname}@polimi.it

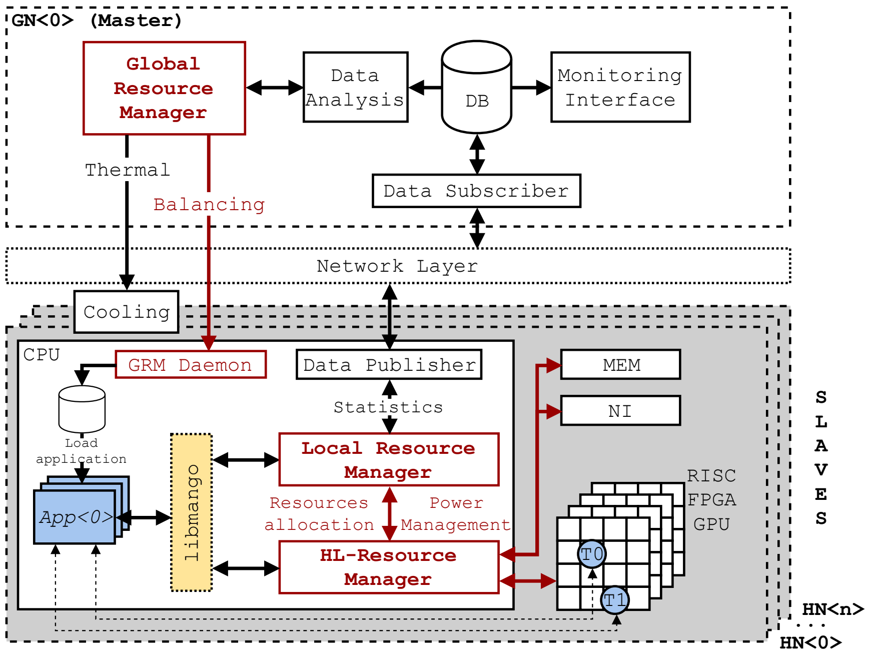

Figure 1: The hierarchical architecture for the distributed resource management. The "master" GN-node hosts the instance of the GRM and specific software to manage and monitor the global infrastructure. The node is interconnected through the Network Layer to different "slave" HN-nodes, where a LRM is in charge to allocate the different tasks of applications (e.g., T0, T1...) to the available resources. Finally, the HRM provides direct access to low-level resources and it is in charge of verifying their availability.

each task may have data and timing dependencies with other ones; (b) the execution of task on different heterogeneous computing units could lead to obtain different QoS or power consumption; (c) applications' requirements must be satisfied, while addressing system's constraints (power/thermal/energy); d) the infrastructure is composed by various computational nodes equipped with heterogeneous resources. Our work aims at addressing the aforementioned challenges inside the context of the MANGO project [FAA+16] [FAA+17] by providing managed applications with a resource-agnostic view of the available resources. Developers writes the applications following a specific programming model, then a hierarchical resource management approach, depicted in Figure 1, addresses the allocation of tasks to the available resources, performing the required optimizations at both infrastructure and node levels. In particular, we propose a 3-layers hierarchy, that comprises: a *Global Resource Manager* (GRM) that acts at infrastructure-wide level; a *Local Resource Manager* (LRM) that is in charge of managing the resources of one specific node; and a *Heterogeneous-Level Resource Manager* (HRM) which performs low-level management operations on the specific node's hardware.

2 Local node management

In the MANGO infrastructure, the applications must be written with the dedicated programming model [AFM+18], similar to OpenCL. Each host-side executable is linked with the run-time library called *libmango*. The library is in charge of exchanging the run-time information with the LRM BarbequeRTRM [BMF15]. The application provides to the library the description of the workload to be deployed on heterogeneous node as a *task graph*. The task graph is a direct, possibly cyclic, graph, with the arcs representing the mapping between the kernel, the buffers and the events. The nodes can be buffers or kernels and two adjacent nodes cannot be of the same type. The resource manager exploits the task graph to pick the best resource allocations among the available ones.

Due to the hardware heterogeneity and complexity, a monolithic decision algorithm would be unfeasible for the huge decision space. In fact, the resource assignment is a complex decision that has to consider: (1) the available processor types and the target architectures of the application kernels; (2) the number of threads of the kernels and the number of cores of the processors; (3) the NoC resources, in terms of bandwidth and availability of virtual networks; (4) the memory allocation, especially considering the internal fragmentation; (5) the temperature and power requirements at different layers, from the single core to the entire infrastructure; (6) the potential cache conflicts and optimizations; (7) the application QoS and its variability. Taking in account all of these factors entails a multi-dimensional and large decision space, especially in the presence of multiple applications. For this reason, the decision is splitted among different actors: the HRM is in charge of verifying the availability of low-level resources, such as the NoC bandwidth. Given the feasible allocations, the local resource manager explores the still large decision space helped by a dedicated memory manager. Current ongoing works are trying to solve this problem using genetic algorithms to find a quasi-optimal allocation.

3 Infrastructure-level management

At the top of the hierarchy we need to consider the infrastructure-wide requirements and constraints. In particular, nodes are organized as follow: a "master" General-purpose Node (GN) hosts the instance of the *Global Resource Manager* (GRM) SLURM [YAG03], as well as specific software to manage and monitor the global infrastructure; "slave" Heterogeneous Nodes (HN) perform the effective computation and are further locally managed as explained in the previous Section. An interposed *Network layer* provides inter-node communication capabilities through LAN Ethernet connections.

At infrastructure-level we need to achieve two objectives: (1) performing load balancing among the slave nodes; (2) performing an effective thermal management.
Load balancing is important to maintain a good QoS, preserving an acceptable power and thermal conditions. In order to perform such balancing, the GRM is in charge of dispatching the execution of an application on a specific node, basing on the global status of the architecture. The latter comprises the information collected from the LRMs, enriched by thermal and cooling information. In fact, thermal management needs to be addressed jointly with cooling control to ensure reliability and maximize energy efficiency. Thus, allocation strategies can leverage on those information to exploit the new architectures and the heterogeneity of the entire platform for the particular target applications.

In order to provide the aforementioned management, a *Data Layer* is introduced to collect run-time statistics coming from the LRMs and to provide them to different actors. *Data analytics softwares* can produce knowledge on the infrastructure behaviour related to applications dispatch and execution with respect to thermal, power and performance measurements. This knowledge is further used to feed the GRM strategies. A *Monitoring Interface* provides a run-time and complete view of the actual status of the system. In order to be scalable and easily extensible, the *Data Layer* is based on a publish-subscribe mechanism. The master node exploits a *Data Communication Interface* implementing a *Data Subscriber* in order to subscribe to different information related to specific slave nodes. In turn, each slave node's LRM has been extended with a *Data Publisher* that collects subscriptions possibly coming from different subscribers and push to them the most updated required information.

4 Conclusions

Dealing with heterogeneous architectures and task-based applications requires to consider different aspects at both infrastructures level and single node in order to meet power/thermal and performance requirements. Thus, in order to provide an effective and fine-grained management of the available resources, as well as dispatching applications among the different computing nodes, we proposed a hierarchical approach in which different resource managers, located in the nodes, collaborate to reach a multi-objectives optimization. We plan to prove the benefits of our approach inside the MANGO project context and to apply it also in emerging computing scenarios, such as Distributed Mobile Computing and Fog paradigms, where heterogeneous resources are spread across different interconnected devices.

References

[AFM$^+$18] G. Agosta, W. Fornaciari, G. Massari, A. Pupykina, F. Reghenzani, and M. Zanella. Managing heterogeneous resources in hpc systems. In *Proceedings of the 9th Workshop and 7th Workshop on Parallel Programming and RunTime Management Techniques for Manycore Architectures and Design Tools and Architectures for Multicore Embedded Computing Platforms*, PARMA-DITAM '18, pages 7–12, New York, NY, USA, 2018. ACM.

[BMF15] P. Bellasi, G. Massari, and W. Fornaciari. Effective runtime resource management using linux control groups with the barbequertrm framework. *ACM Trans. Embed. Comput. Syst.*, 14(2):39:1–39:17, March 2015.

[FAA$^+$16] J. Flich, G. Agosta, P. Ampletzer, D. A. Alonso, C. Brandolese, A. Cilardo, W. Fornaciari, Y. Hoornenborg, M. Kovac, B. Maitre, G. Massari, H. Mlinaric, E. Papastefanakis, F. Roudet, R. Tornero, and D. Zoni. Enabling hpc for qos-sensitive applications: The mango approach. In *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 702–707, March 2016.

[FAA$^+$17] J. Flich, G. Agosta, P. Ampletzer, D. A. Alonso, C. Brandolese, E. Cappe, A. Cilardo, L. Dragic, A. Dray, A. Duspara, W. Fornaciari, G. Guillaume, Y. Hoornenborg, A. Iranfar, M. Kovac, S. Libutti, B. Maitre, J. M. MartÃnez, G. Massari, H. Mlinaric, E. Papastefanakis, T. Picornell, I. Piljic, A. Pupykina, F. Reghenzani, I. Staub, R. Tornero, M. Zapater, and D. Zoni. Mango: Exploring manycore architectures for next-generation hpc systems. In *2017 Euromicro Conference on Digital System Design (DSD)*, pages 478–485, Aug 2017.

[YAG03] A. B. Yoo, M. A.Jette, and M. Grondona. Slurm: Simple linux utility for resource management. In Dror Feitelson, Larry Rudolph, and Uwe Schwiegelshohn, editors, *Job Scheduling Strategies for Parallel Processing*, pages 44–60, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.