Topic 9: Parallel and Distributed Programming (Introduction)

José Cunha, Michael Philippsen, Domenico Talia, and Ana Lucia Varbanescu

Topic Committee

This topic provides a forum for presentation of new results and practical experience in the development of parallel and distributed programs. The development of high-performance, correct, portable, and scalable parallel programs is a hard task, requiring advanced algorithms, realistic modeling, adequate programming abstractions and models, efficient design tools, high performance languages and libraries, and experimental evaluation. Current challenges in this topic are concerned with improved solutions for conciliating the transparency and expressiveness of the programming abstractions and models, with new issues arising in modern applications with increasing problem size and complexity, and in heterogeneous computing infrastructures with varying performance, scalability, failure and dynamic behaviors. This motivates for example, abstractions for handling concurrency, parallelism and distribution, and support for predictable performance, self-adaptation, fault-tolerance, and large-scale deployment.

This year, a diversity of papers was submitted to this topic, proposing relevant and valuable research contributions. As a result of the reviewing process, 4 papers were accepted for publication. Globally, the accepted papers discuss the gaps between high-level programming abstractions and the domain experts and application developers, and present reports of their implementation and evaluation via concrete applications and performance benchmarks.

One of the papers (by Nanz, West, and Silveira) presents a comparative study of four approaches with distinct parallel programming abstractions and communication paradigms, which are then evaluated via a suite of benchmark programs. Original and revised versions by experts are compared with respect to source code size, coding time, execution time, and speedup. Two of the papers discuss structured models of parallelism as ways of easing the parallel programming tasks. One paper (by Legaux, Hu, Loulergue, Matsuzaki, and Tesson) explores algorithmic skeletons in conjunction with a notion of list homomorphism in the context of the bulk synchronous parallelism (BSP) model and reports on its use for parallel algorithm development in two applications, with an implementation of BSP homomorphism in a data-parallel algorithmic skeletons library. The other paper (by Tasci and Demirbas) also explores the BSP model for parallel processing of large-scale graph applications. The paper discusses modifications to the BSP model and how these were supported with improved performance, on top of Giraph, an open-source clone of Pregel, Google's scalable infrastructure for graph-based mining. Another accepted paper (by Sharp and Morgan) discusses a critical concern in transactional memory systems, regarding the occurrence of conflicts that lead to transactions aborting and retrying their execution. The

F. Wolf, B. Mohr, and D. an Mey (Eds.): Euro-Par 2013, LNCS 8097, pp. 432–433, 2013.

paper describes an implementation for determining the scheduling of aborted transactions, and discusses the effectiveness of the proposal via performance benchmarks.

We would like to thank all the authors who submitted papers to this topic, and the external reviewers, for their contribution to the success of the conference. We also thank the overall coordination and valuable support that was provided by the conference chairs.