First Annual Workshop on Programming Model Alternatives to Message Passing (PMAMP)-2017

Overview:

Message-passing programming models have dominated high-performance computing (HPC) for the past quarter century. With the continued breakdown of the (uniform) communicating sequential processors abstract machine model, many researchers have questioned the continued viability of message passing as a model for direct, application-level interaction. Recent years have seen an explosion of new paradigms in programming models for distributed-memory computing, ranging from evolutionary to revolutionary. With this paradigm shift, it has become clear that raw performance is not the only concern in the design of a programming model; rather, for many applications the overall time-to-solution is just as important. The move to the next generation of HPC platforms presents a wider variety of challenges than ever before. Potential challenges to address include the increased need for asynchrony, increased heterogeneity and performance non-uniformity, decreased hardware reliability and increased failure rates, increased hardware diversity, and the maintainability of increasingly complex scientific code bases. Alternative programming models that address some or all of these challenges include task-based programming models (and a variety of higher-level programming models that generate task-based execution), distributed-memory actor models, extensions and augmentations to the existing message-passing paradigm (such as active messages), and partitioned global address space models (implemented at a broad range of abstraction levels).

This workshop proposes to promote a dialogue about the spectrum from evolutionary to revolutionary programming models and the relative importance of the concerns these new approaches address. Dialogues we propose to encourage include: the right level of abstraction for application developers to interact with; the importance of performance vs. productivity when developing both rapid prototypes and production-scale applications; the relative importance of fault-tolerance and the potential for non-uniformity in future hardware; the feasibility of obtaining performance without machine-specific implementation (and the importance of doing so); and the viability of evolving or replacing current programming models in production-scale scientific applications. Another goal of this workshop is also to expose researchers who work on extensions or augmentations to message-passing models (the "evolutionary" crowd) to alternative programming models ***and*** to expose those who develop fundamentally different programming models (the "revolutionary" crowd) to the recent advances in message-passing programming models. The workshop seeks to garner a better understanding of key motivations for alternative programming models, key insights from advances in message passing (and how they can be applied to alternative models), and key opportunities for collaboration between researchers on both sides.

Topics of Interest:

We solicit preliminary work on programming models that offer extensions or alternatives to message passing, with emphasis on addressing emerging concerns in high performance computing (HPC), including topics relating to (but are not restricted to) the following:

- Asynchrony
- Heterogeneity
- Performance non-uniformity
- Fault-tolerance and resilience
- Performance portability:
    - *a posteriori* adaptability ("one code for many machines")
    - *in situ* adaptability (reactive and dynamic runtime system solutions)
- Maintainability (particularly in the presence of these other challenges)
- Testability and debuggability (particularly in the presence of these other challenges)
- Experimental comparative results (particularly with emphasis on overall time-to-solution)
- Relevance of alternate programming models to particular scientific kernels
- Provision for power/energy/temperature management

Also, we solicit position papers on analyzing the importance (or lack thereof) of the above topics for the future of HPC, again with emphasis on the need (or lack thereof) to address these concerns *at the programming model level* (thus, for instance, arguments for why the challenges can be addressed at other levels of the software stack without programming model intervention will also be considered).

Submission Guidelines and Important Dates:

Interested authors are encouraged to submit full papers (8-10 pages) or short/position papers (4 pages) "sigconf" style in the ACM 2017 template. This page limit includes figures, tables and appendices but not references. Authors should submit their work at this Easychair website (https://easychair.org/conferences/?conf=pmamp1).

Important Dates:

Paper submission: June 15, 2017
Author Notification: July 10, 2017
Camera-ready papers due: July 30, 2017
Workshop: September 25, 2017