Implicitly-Parallel Functional Dataflow for Productive Cloud Programming on Chameleon

Scott Krieder, Ioan Raicu
Illinois Institute of Technology
skrieder@hawk.iit.edu, iraicu@cs.iit.edu

Justin Wozniak, Michael Wilde
Argonne National Laboratory
{wozniak, wilde}@mcs.anl.gov,

1. INTRODUCTION

This project explores a programming-model and runtime-environment that addresses the urgent yet vexing problem of simplifying the programming of distributed parallel systems.

One solution that makes parallel programming implicit rather than explicit is the dataflow model. Conceived ~35 years ago, it has only recently been made practical through systems such as Dryad and Swift [1]. We believe that we have successfully created a base for an implicitly-parallel functional dataflow programming model, as exemplified by Swift, a workflow language for executing scientific applications. This model has been characterized as a perfect fit for the many-task computing (MTC) paradigm. Some broad application classes that fit the MTC paradigm are workflows, MapReduce, high-throughput computing, and a subset of high-performance computing. MTC emphasizes using many computing resources over short periods of time to accomplish many smaller computational tasks (both dependent and independent), where the primary metrics are measured in seconds. MTC has proven successful in grid computing and supercomputing, but the distributed nature of today's cloud resources pose many challenges in the efficient support of MTC workloads. This work aims to address the programmability gap between MTC and cloud computing, through an innovative parallel scripting language, Swift, which will enable MTC workloads to efficiently leverage cloud resources. This work will enable a broader class of MTC applications to leverage cloud systems.

This project addresses the following research problems:

- Supporting diverse cloud instances: (general-purpose & memory-intensive)
- Scaling downwards (e.g. increasing the spectrum of applications by decreasing leaf-function granularity)
- Scaling upwards (increasing scalability to extreme-scale clouds)
- Language interoperability (integrate with many languages and programming-models for performing leaf tasks: C/C++, Fortran; MPI, OpenMP)
- Runtime facilities for tracing and debugging large distributed parallel workflows
- Evaluation of the programming model on applications in: global crop modeling, cancer detection, glass-state materials, and biophysical dynamics.

This work represents a non-traditional community for cloud-systems in general and Chameleon in particular, one that focuses on MTC. There are many advantages to MTC, such as improved programmability, implicit parallelism, and improved fault tolerance, all reasons why applications and researchers in the community have adopted MTC as their programming-model for large-scale applications. Given the popularity of cloud infrastructures, being able to leverage the use of MTC on cloud architectures such as Chameleon, at large scale, is a critical activity towards the acceptance of MTC as a viable programming model for future cloud computing.

2. Systems Software

The elasticity of the cloud is well suited to the Swift model: Swift can release and re-obtain nodes as the workflow's demand varies. The ability to provide a "computing commons" to support cross-institution collaborations without the complexities of local authentication/authorization will pave the way for future collaborations. And the ability to extend campus resources on-demand for critical deadlines is invaluable.

2.1 Swift implicitly parallel functional dataflow language

We will integrate the Swift parallel programming system with the Chameleon platform. Swift has been successfully used in many large-scale computing applications to increase productivity in running complex applications. Its dataflow-driven programming model, allows implicit, pervasive parallelism to be harnessed through automated dependency management.

2.2 GeMTC

GeMTC [2] is a CUDA based framework for supporting many-task computing workloads on NVIDIA based GPGPU devices. As shown in Figure 1, a NVIDIA GPU is comprised of many Streaming Multiprocessors (SMXs). A SMX contains many warps, and each warp provides 32 concurrent threads of execution. All threads within a warp run in a Single Instruction Multiple Thread (SIMT) fashion. GeMTC schedules independent computations on the GPU at the warp level, a level of independent task concurrency not provided by any mainstream GPU programming model.

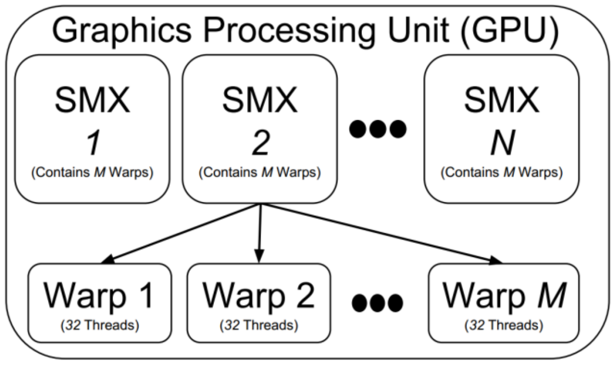

Figure 1:Diagram of GPU Architecture Hierarchy

Recent work extended GeMTC for integration with Swift to support GPGPUs in high-performance computing environments [3]. Figure 2 shows a high-level diagram of GeMTC driven by tasks generated by the Swift. GeMTC launches a daemon on the GPU that enables independent tasks to be multiplexed onto warp-level GPU workers. A work queue in GPU memory is populated from calls to a C-based API, and GPU workers pick up and execute these tasks. After a worker has completed a computation, the results are placed on an outgoing result queue and returned to the caller.

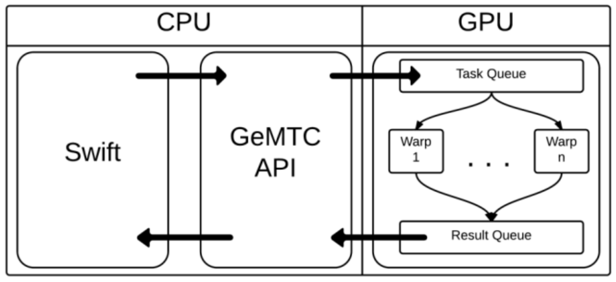

Figure 2: Flow of a task in GeMTC

In addition, preliminary work has already evaluated the feasibility of many-task computing on the Intel Xeon-Phi Co-processor [4]. This work will extend previous work by evaluating new applications in a cloud environment with next generation GPUs and Co-processors on Chameleon.

3. Application Codes

3.1 FACE-IT

FACE-IT [5] provides a growing collection of web-based pipelines that integrate data and software tools, enabling researchers to easily develop data manipulation and analysis tools, apply those tools to data sets, link multiple tools into analysis pipelines, and share these among communities.

3.2 Identifying Cancer-related genes

ExSearch [6] is a novel algorithm simplifying complex gene classifiers in cancer. A machine learning application tuned to analyze class-labeled data using n-tuple feature vectors, it produces generalized workflow identifying genes that relate to cancer. It's an excellent candidate for cloud computing and public community tool access.

3.3 Glass material modeling application

Hocky et. al., describe algorithms for evaluating new cavity methods to measuring the "mosaic length" of glass transition systems. [7] Particles are simulated by molecular dynamics or Monte Carlo methods within cavities with amorphous boundary conditions. These simulations were driven by Swift scripts, which will enable us to easily leverage Chameleon and explore the cloud-based use of MPI.

3.4 Protein structure application

OOPSOpen Protein Simulator is a suite of C++ programs for the prediction of the structure of proteins with minimal use of information derived from sequence-similarity or homology to other proteins. It derives speed and accuracy from the use of simplified models, accurate statistical potentials, and search strategy involving "iterative fixing" in multiple "rounds" of folding. Its community seeks a cloud-hosted simulation portal.

4. REFERENCES

[1] Wilde, Michael, et al. "Swift: A language for distributed parallel scripting." *Parallel Computing* 37.9 (2011): 633-652.

[2] S. J. Krieder and I. Raicu, "Towards the support for many-task computing on many-core computing platforms," Doctoral Showcase, IEEE/ACM Supercomputing/SC, 2012

[3] Scott J. Krieder, Justin M. Wozniak, Timothy Armstrong, Michael Wilde, Daniel S. Katz, Benjamin Grimmer, Ian T. Foster, and Ioan Raicu. 2014. Design and evaluation of the gemtc framework for GPU-enabled many-task computing. (HPDC '14).

[4] J. Johnson, S. J. Krieder, B. Grimmer, J. M. Wozniak,
M. Wilde, and I. Raicu, "Understanding the costs of many-task computing workloads on intel xeon phicoprocessors," in 2nd Greater Chicago Area System Research Workshop (GCASR), 2013.

[5] http://www.faceit-portal.org/home

[6] Wilson, Raphael A., et al. "A novel algorithm for simplification of complex gene classifiers in cancer." Cancer research 73.18 (2013): 5625-5632.

[7] Hocky, Glen M., et al. "Growing point-to-set length scale correlates with growing relaxation times in model supercooled liquids." Physical Review Letters 108.22 (2012): 225506.