Early Experiences in running Many-Task Computing workloads on GPGPUs

Scott J. Krieder*, Ioan Raicu*†
*Department of Computer Science, Illinois Institute of Technology
†MCS Division, Argonne National Laboratory

Abstract*—Many-task computing (MTC) aims to bridge the gap between two computing paradigms, high throughput computing (HTC) and high-performance computing (HPC). MTC emphasizes using many computing resources over short periods of time to accomplish many computational tasks (i.e. including both dependent and independent tasks), where the primary metrics are measured in seconds. MTC denotes high-performance computations comprising multiple distinct activities, coupled via file system operations. Swift is a particular implementation of the MTC paradigm, and is a parallel programming system that has been successfully used in many large-scale computing applications across the TeraGrid and now XSEDE. It has been adopted by the scientific community as a great way to increase productivity in running complex applications via a dataflow driven programming model, which intrinsically allows implicit parallelism to be harnessed based on data access patterns and dependencies. Swift has been shown to run well on tens of thousands of nodes with task graphs in the range of hundreds of thousands of tasks. This work aims to enable Swift to efficiently use accelerators (such as NVIDIA GPUs and Intel MIC) to further accelerate a wide range of applications. This work evaluates a real biochemistry application, namely the the Open Protein Simulator (OOPS), which builds on the Protein Library (PL). OOPS is multipurpose and allows extensions to perform various simulation tasks relevant for life scientists, such as protein folding or protein structure prediction. We have taken parts of this application and ported to NVIDIA GPUs via the CUDA programming language, in order to accelerate OOPS computations via Swift. This work presents preliminary results in the costs associated with managing and launching concurrent kernels on NVIDIA FERMI GPUs, through the Swift system. We expect that our results to be applicable to several XSEDE resources, such as Forge, Keeneland, and Lonestar, where currently Swift can only use the general processors to execute workloads and the GPUs are left idle.

***Keywords*-Many-Task Computing, Swift, GPGPU, CUDA, OOPS**

I. Introduction

Text. [1]

II. Many-Task Computing

III. Future Work

IV. Conclusions

References

[1] S. J. Krieder and I. Raicu, "An overview of current and future computing accelerator architectures," 1st Greater Chicago Area System Research Workshop Poster Session, 2012.