SUPPORTING EFFICIENT GRAPH ANALYTICS AND SCIENTIFIC COMPUTATION USING ASYNCHRONOUS DISTRIBUTED-MEMORY PROGRAMMING MODELS

Abstract

by Sayan Ghosh, Ph.D.
Washington State University
May 2019

Chair: Assefaw H. Gebremedhin

Future High Performance Computing (HPC) nodes will have many more processors than the contemporary architectures. In such a system with massive parallelism it will be necessary to use all the available cores to drive the network performance. Hence, there is a need to explore one-sided models which decouple communication from synchronization. Apart from focusing on optimizing communication, it is also desirable to improve the productivity of existing one-sided models by designing convenient abstractions that can alleviate the complexities of parallel application development. Classically, a majority of applications running on HPC systems have been arithmetic intensive. However, data-driven applications are becoming more prominent, employing algorithms from areas such as graph theory, machine learning, and data mining. Most graph applications have minimal arithmetic requirements, and exhibit irregular communication patterns. Therefore, it is useful to identify approximate methods that can enable communication-avoiding optimizations for graph applications, by potentially sacrificing some quality.

The first part of this dissertation addresses the need to reduce synchronization by exploring one-sided communication models and designing convenient abstractions that serve the need of distributed-memory scientific applications. The second part of the dissertation is about evaluating

the impact of approximate methods and communication models on parallel graph applications.

We begin with the design and development of an asynchronous matrix communication interface that can be leveraged in parallel numerical linear algebra applications. Next, we discuss the design of a compact set of C++ abstractions over a one-sided communication model, which improves developer productivity significantly. Then, we study the challenges associated with parallelizing community detection in graphs, and develop a distributed-memory implementation that incorporates a number of approximate methods to optimize performance. Finally, we consider a half-approximation algorithm for graph matching, and evaluate the implications of different communication models in its distributed-memory implementation. We also examine the effect of data reordering on performance.

In summary, this dissertation provides concrete insights into designing low-overhead high-level interfaces over asynchronous distributed-memory models for building parallel scientific applications, and presents empirical analysis on the effect of approximate methods and communication models in deriving efficiency for irregular scientific applications using distributed-memory graph applications as a use-case.