SKaMPI – Towards Version 5

Werner Augustin, Michael Haller, Marc-Oliver Straub, and Thomas Worsch
LIIN, Universität Karlsruhe, Germany,
<augustin|haller|worsch}@ira.uka.de, straub@sb-software.de

Summary. SKaMPI is now an established benchmark for MPI implementations. The development of SKaMPI-5 strives for improvements in several directions: (i) extension of the benchmark to cover more functionality of MPI, (ii) construction of a collection of collective algorithm kernels which are not supported by core MPI collective operations, (iii) a redesign of the SKaMPI benchmark allowing it to be extended more easily (thus matching requests from SKaMPI users).

In the present paper we give an overview of the extension of SKaMPI for the evaluation of virtual topologies, describe the foundations of new algorithms for fast all-to-all communication specifically tailored for the case of differing message sizes, and give a first impression of what SKaMPI-5 will look like, for which we now have a prototype running.

1 Introduction

“The MPI standard defines a set of powerful collective operations useful for coordination and communication between many processes. Knowing the quality of the implementations of collective operations is of great interest for application programmers. In particular, one has to decide, whether to use predefined collective operations, which usually lead to more readable programs, or to implement collective operations by using point-to-point primitives. Similarly, it is often unclear, whether to use complex collective operations, like MPI_Reduce_scatter, or to use more primitive collective operations (like in this case MPI_Reduce and MPI_Scatterv).”

The above text [7] describes one of the major motivations for the development of SKaMPI, the Special Karlsruher MPI-Benchmark [5].

SKaMPI (http://liinwww.ira.uka.de/~skampi/) measures the performance of an MPI implementation on a specific underlying hardware. By providing not simply one number, but detailed data about the performance of each MPI operation, a software developer can judge the consequences of design decisions regarding the performance of the system to be built.
The text also indicates, and benchmark results from a wide variety of machines indeed show, that more often than it should be there is for example a collective operation which is implemented in a suboptimal way, asking for self-made replacements. The development of new implementations for collective operations covered by MPI and for more complex collective operations not covered by MPI is thus a natural next step. The same holds for the possibility to explore the influence of virtual topologies.

The rest of this paper is organized as follows. In Section 2 we give a short overview over the components for the new SKaMPI. Section 3 shows how the latest version 4.1 of the current SKaMPI-4 can now be used to investigate the (dis-)advantages of using virtual topologies in an MPI implementation. In Section 4 we sketch a specific case of MPI_Alltoallv which we will use for benchmarking possible algorithms for irregular all-to-all communication. In Section 5 we give an overview of those features which are already present in a prototype of SKaMPI-5 and of those which will be added for its first stable release. We conclude this paper in Section 6.

2 Overview of the New SKaMPI

SKaMPI is now an established benchmark for MPI implementations. Work has continued to include more MPI functions in it. Until version 4.0 virtual topologies had been neglected, because “benchmarking” an implementation of virtual topologies does not fit well into the standard framework of looping over a set of parameter values. In Section 3 we describe how they were included in SKaMPI 4.1 and report some surprising results obtained with it.

For all non-trivial MPI functions the current implementation of SKaMPI does one loop varying one parameter (e.g. message size or number of processes) while keeping all others fixed. There are cases where it would be more convenient to have an easy way to get measurement data for e.g. all pairs of parameter values (e.g. message size and number of processes). Also, SKaMPI should be extensible. This is already a must if one wants to include the measurement of self-written routines (e.g. intended to replace the native implementation of a collective operation). But there are even users who want to use much of SKaMPI for benchmarking OpenMP implementations.

Of course all of these improvements should not sacrifice any features of SKaMPI, nor its stability and portability. In Section 5 we describe a prototype of SKaMPI-5 and first experiences concerning its extendibility.

3 Virtual Topologies in SKaMPI-4

According to the MPI standard [2, Section 6.1], a virtual topology is an extra, optional attribute that one can give to an intra-communicator [...]. A topology [...] may assist the runtime system in mapping the processes onto hardware.