Characterizing MPI Messaging and Communication Costs

Mary Thomas

March 1, 2014

1 Introduction

The execution time of a parallel algorithm is defined as the elapsed time from when the first processor begins execution to when the last processor completes its tasks, and the generalized for can be written as:

$$T_{exec}(N,P) = T_{comp}(N,P) + T_{comm}(N,P) \tag{1}$$

Where $T_{exec}(N,P)$, $T_{comp}(N,P)$, and $T_{comm}(N,P)$ are the total execution or run time, the total computational time, and the total communication time require to run a problem of size N on P processing elements (PE's). The performance model can be extended to include the cost of message passing by developing a term based on the message startup time (t_s) and the communication bandwidth of the system (t_w).

The performance characteristics of any model can be modeled and measured, which will identify where there might opportunities for parallelization and to predict the run-times of simulations. Typically, the computation and communication costs are measured using some timing routines. Choosing what parts of the model to measure or analyze depends on several factors, including:

- In-depth knowledge of the model and where the largest calculations occur (often called "Big-O", or O analysis).
- Empirical knowledge is used to help calibrate the model.
- Unless it plays a major role in the performance, hardware details are ignored.
- Scale analysis is used to identify and eliminate those parts of the code that are insignificant.

A major goal of any parallel program includes reduction of unnecessary communication time. In this homework module we will investigate the factors that can affect program run-times associated with MPI Messages.

2 MPI Communication

The message passing communication time required to send N words (or Bytes) can be written as defined as

$$T_{comm} = t_{startup} + t_{bw} \tag{2}$$

where $t_{startup}$ is the message startup time (latency), and t_{bw} is the message passing saturation bandwidth. Message latency is the time required to set up communications on the nodes and to prepare them to send a message. MPI latency is usually estimated to be *half of the time* of a ping pong operation with a message of size zero. Examples of results from a ring program can be seen in Figure 1.

2.1 MPI Latency or Startup Time

Message latency is the time required to set up communications on the nodes and to prepare them to send a message. MPI latency is usually estimated to be *half of the time* of a ping pong operation with a message of size zero. In pingpong, packets of information are exchanged between two processing elements and the time required to do this is measured. The message startup time is especially critical when working with very fine-grained applications which have more frequent communication requirements. It is a function of the number of messages that need to be sent, which in turn depends on the number of processors communicating and the number and size of the message packets.

2.2 MPI Bandwidth

The bandwidth is defined as the peak rate at which data packets can be sent across the network. This is important for coarse-grained codes that send fewer messages but typically need to communicate larger amounts of data. It is important to have some knowledge about the network(s) being used and the communication protocol, both of which limit the maximum expected (or peak) bandwidth. The bandwidth can be estimated using the ping-pong and ring programs. The packets of information consist of an array of dummy integer or floating point numbers (would these have different run times?) that vary in length from one floating point number to a large

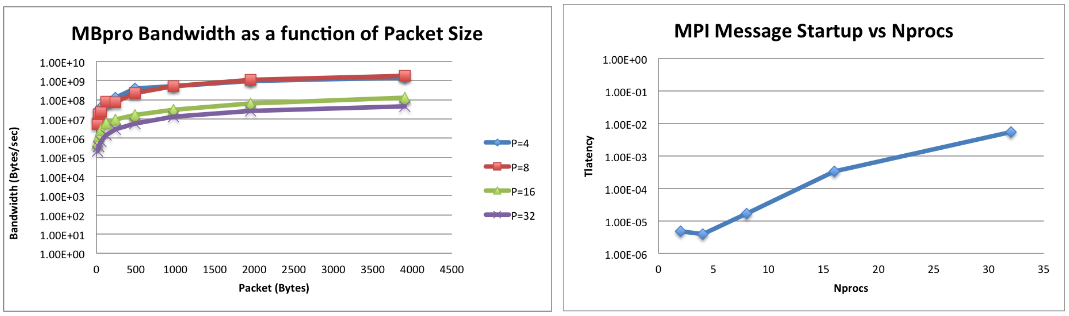

Figure 1: Figures a (left) and b (right) show MPI Message bandwidth and startup times on the MacBookPro, using the MPICH-2 MPI Library)

number of floating point numbers. These packets are sent back and forth from one process to another NC times and the total amount of time required is recorded. From this timing data the average amount of time per send/receive is computed as well as the saturation bandwidth of the system (typically Mbytes/sec).

In a ring program you can send data around a ring of PEs, starting when P_0 sends its data to P_1, and stopping when P_0 receives data from P_{n-1} (unidirectional). In bidirectional ring (exchange), all processors send data to each other. An example of the ring output is shown in Figure 1. The saturation bandwidth is obtained when the bandwidth does not improve as the packet size increases.

References

[1] P. Pacheco, *Parallel Programming with MPI*. Morgan Kaufmann Publishers Inc., 1997.

[2] M. P. Thomas, “Introduction to Scientific Computing,” 2014.

[3] W. Kendall, “MPI Tutorial.com, url = http://mpitutorial.com/, urldate = 2/14/2014, year = 2014.”