Toward Mega-Scale Computing with pMatlab

Chansup Byun and Jeremy Kepner
{cbyun, kepner}@ll.mit.edu
MIT Lincoln Laboratory, Lexington, MA 02420
Vipin Sachdeva and Kirk E. Jordan
{vsachde, kjordan}@us.ibm.com
IBM T.J. Watson Research Center, 1 Rogers St, Cambridge, MA, 02142

Introduction

In recent years, significant progress in processor and system technologies has led to the appearance of many-core processors, such as the Tilera 100 core [1] and mega-core systems [2]. It is desirable to exploit such computing capability for desktop-oriented applications, such as MATLAB®, which are popular for algorithm development in the signal and image processing area.

The MIT Lincoln Laboratory Grid (LLGrid) team has developed pMatlab [3], a parallel MATLAB toolkit that makes parallel programming with MATLAB accessible and simple by using two partitioned global address space (PGAS) data types, parallel maps and distributed arrays. This enables pMatlab programmers to work in their familiar environment of numerical arrays and to parallelize their serial codes with only a few lines of code changes.

In this study, we investigate using pMatlab to exploit supercomputing systems with a large number of compute cores. First, we discuss the technical challenges of porting the pMatlab toolkit to a mega-core system. Next, we present how we resolved these challenges to extend the interactive, on-demand experience of pMatlab. Finally, some preliminary results of running MATLAB and Octave with pMatlab on these systems are presented.

Massively Parallel Systems

In order to provide more computing power to the users, we will need to exploit massively parallel systems, which in turn requires highly scalable software. This study focuses on the IBM Blue Gene/P (BG/P) "Surveyor" system at Argonne National Laboratory [4]. The Surveyor system has 1024 compute cards and each compute card has a quad-core IBM PowerPC chip running at 850MHz with 2GB memory. This system offers us the opportunity to test pMatlab at scales up to 4096 processes.

Porting Parallel MATLAB

The parallel Matlab toolkit is made of two layers. The pMatlab layer provides parallel data structures and library functions and the MatlabMPI layer provides messaging capability [3]. Both layers are written as MATLAB M files in ASCII code, which makes the toolkit easy to port to other systems. The major issue in porting the pMatlab toolkit to the BG/P system was how to incorporate the BG/P job submission mechanism into the existing pMatlab launch mechanism. The other issue was to port a math library that is compatible with pMatlab. Since MATLAB is not available on the BG/P system, an alternative open-source version of MATLAB-equivalent software, Octave, is used for this study.

Single Process Performance

The Octave performance on the Surveyor system was first investigated by comparing the single process performance results obtained on the Surveyor system with those obtained on the LLGrid system using representative computational kernels [5]. The LLGrid system is a Dell server with Intel Xeon 3GHz processor [6]. Also, as a reference, we compared the MATLAB performance on the LLGrid system as well.

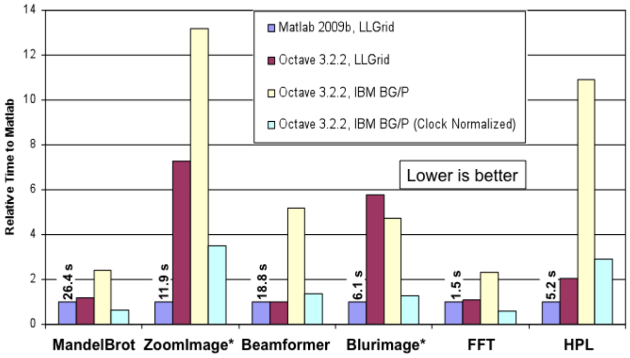

Figure 1: Single process performance comparison between Intel Xeon and IBM PowerPC processors.

Because all the examples discussed here were small, their execution times were dominated by the CPU clock speed. To account for this we also compared the clock-normalized Octave performance on the Surveyor system. These clock normalized performance matches well with the MATLAB performance on the LLGrid system in most cases (see Figure 1).

However, some of examples were slower with Octave as compared to MATLAB on the LLGrid system. We found that the performance of the ZoomImage and Blurimage examples were dominated by the 2-D convolution function with Octave. The performance issue with the 2-D convolution function with Octave has been addressed with an optimized version in the new release that is about 2x faster than the old one [7]. In addition, the High Performance LINPACK benchmark revealed a performance gap between MATLAB and Octave. In this instance, the slow performance in Octave is attributed to the generic BLAS function (DGEMM) called by Octave on the LLGrid

* This work is sponsored by the Department of the Air Force under Air Force contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government. MATLAB® is a registered trademark of The Mathworks, Inc.

system. This can be resolved by using machine specific optimized BLAS libraries.

Point-to-Point Communication Performance

Point-to-point communication is an essential component of parallel applications. pMatlab throughput bandwidth and timing results of point-to-point communication on the Surveyor system are shown in Figures 2 and 3, respectively. Three different process sets of (two, four and eight processes) with one process per compute card, were used to measure the performance on the IBM's GPFS file system. [Note: pMatlab uses the file system for communication.] As the size of messages was increased from 16 bytes to 128 Mbytes, bandwidth performance increased linearly for small messages but leveled out as the message size increased beyond 4 Mbytes. This is primarily due to the limit on the communication link. The bandwidth per processor for a 128 MByte message were 40, 31, and 25 Mbytes/sec for two, four, and eight process sets, respectively. For smaller messages, the time it took to transfer messages remained constant, around 0.14 seconds (see Figure 3). Overall, the point-to-point communication performed as expected on the Surveyor system with the GPFS file system.

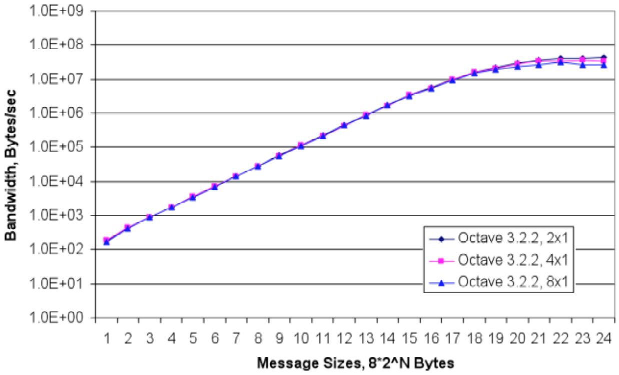

Figure 2: Point-to-point communication performance per processor in terms of throughput bandwidth on an IBM Blue Gene/P system.

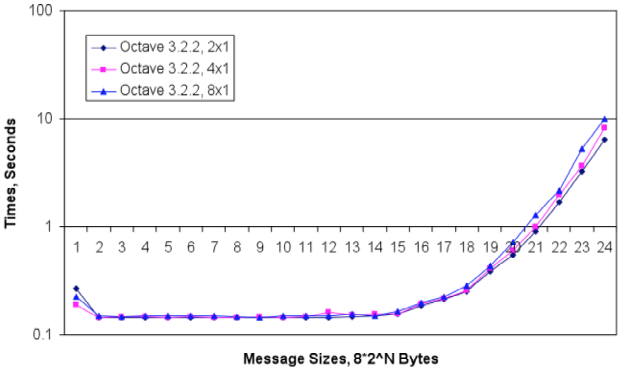

Figure 3: Point-to-point communication performance in terms of time on an IBM Blue Gene/P system.

Scalability

In order to fully utilize the BG/P system, the pMatlab software must scale to a large number of computing cores. To test for scalabilty we ran the High Performance Computing Challenge Stream Benchmark. The benchmark measures bandwidth between a processor and its main memory. This benchmark is a representative of a wide range of MATLAB signal and image processing applications. The aggregated memory bandwidth results from Surveyor are shown in Figure 4. The results were obtained by executing the parallel Stream benchmark code using pMatlab and Octave, up to 1024 processes. As shown in this figure, pMatlab shows nearly linear scalability up to 1024 processes. With 1024 processes, the triad case was able to achieve almost 550GBytes/sec aggregated memory bandwidth performance with 90% parallel efficiency.

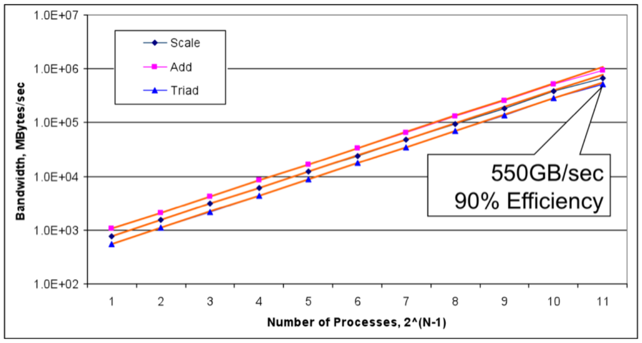

Figure 4: Parallel Stream benchmark on an IBM Blue Gene/P system.

Summary

We have demonstrated that, the clock normalized, single processor Octave performs on par with MATLAB. We further demonstrated that Octave with pMatlab scales up to 1024 processes on the IBM BG/P Surveyor system. We expect that it will scale to an even higher number of processes. Although the IBM BG/P system has a slower PowerPC chip with smaller memory, the combined Octave and pMatlab indicate that the computing power of a mega-core system is a viable choice for signal and image processing applications.

References

[1] Tilera, *The World's First 100-core Processor with the New TILE-Gx Family*, [Online] Available: http://www.tilera.com/news_&_events/press_release_091026.php, 2009.

[2] IBM, *IBM Triples Performance of World's Fastest, Most Energy-Efficient Supercomputer* [Online], Available: http://www-03.ibm.com/press/us/en/pressrelease/21791.wss, 2007.

[3] J. Kepner and N. T. Bliss, "Parallel Matlab: The Next Generation", HPEC2003, Sep 23-25, 2003.

[4] Argonne National Laborarotry, *Surveyor System*, [Online], Available: https://wiki.alcf.anl.gov/index.php/General, 2010

[5] J. Kepner, *Parallel MATLAB for Multicore and Multinode Computers*, First Edition, SIAM, 2009.

[6] A. Reuther, B. Arcand, T. Currie, A. Funk, J. Kepner, M. Hubbell, A. McCabe, P. Michaleas, "TX-2500 – An Interactive, On-Demand Rapid-Prototyping HPC System," HPEC 2007, Lexington, MA, Sep. 2009.

[7] Private Communication with the Octave developer, 2010