CS240A: Applied Parallel Computing Spring 04

HW 4 – Parallel Matvec

Prof. John R. Gilbert *May 24, 2004*

Problem Description

Implement an algorithm in parallel to multiply a sparse matrix with a dense vector. Write your code in C or C++ and if you use F90, you will need to write your own glue code. You should then use the Conjugate Gradient code (Matlab) from *Sparse Matrices in Matlab: Design and Implementation – John R. Gilbert, Cleve Moler and Rob Schreiber* (page 22) to write a conjuagate gradient program for Matlab*P. Due to limited functionality in Matlab*P, you might have to play with that code and modify it slightly so that it works, and you might also want to think of ways to perform simple optimizations.

In the README file, describe in detail

- The algorithm for the local matvec
- The communication scheme for the parallel matvec

You are free to use any sequential sparse matvec code you find on the internet, as long as you cite the source in your README. You may however not use any parallel matvec code that you may find.

Grading

1. Correctness – 30%
2. Performance – 50%
3. Documentation – 10%
4. Effort Logs – 5%
5. Post HW Questionnaire – 5%
6. Extra Credit – 30%

Correctness is absolutely critical to be eligible for the rest of the grade. If your code is incorrect, the grade will solely depend on the discretion of the instructor and the TAs.

You will have to find your own test cases for this problem. You can use sprand as shown in the test example that is included with the homework, but that won't be enough to test your performance. You are encouraged to use

matrices from the matrix market, and meshes and mesh generators included in the meshpart toolbox. These are all easily available online. Talk to the TAs if you need to figure out how to use these resources.

For extra credit, you need to implement a graph partitioning algorithm, such as co-ordinate bisection, use that to permute the sparse matrix to minimize the communication and then use that matrix for the matvec. All this code can be done in Matlab*P in a few lines.

For assessing your performance on this homework, we will try to solve systems using the conjugate gradient code which performs several matvecs.