Project Proposal

Jian Han Ken Mixter
jh@cs.ucsb.edu kmixter@cs.ucsb.edu

1.) Project Goals

MATLAB*P is a parallel version of MATLAB. It provides a MATLAB interface but implements parallel computation by using MPI libraries. All parallel computation is transparent to the user. MATLAB*P version 2 has been successfully used in parallelizing two real applications, one involving video image processing (with collaborators at the MIT AI Lab) and one involving atmospheric modeling (with collaborators in the Earth Sciences Department at MIT). However, many features remain to be implemented to make MATLAB*P a complete system. In particular, two such features are distributed matrix construction operators and matrix indexing. In the current MATLAB*P, only supports some of the most common constructors in MATLAB are supported. Many constructors that can be used in MATLAB cannot be used in *P. For example, we can not currently declare E = [A B, C D] when A, B, C, D are all dense objects. The goal of this project is:

1.) To implement a wide array of construction operators that appears in MATLAB*P (for both dense and sparse objects) and allow the user to construct an object mixed with dense & sparse matrices. For example, the user should be able to declare E = [A B, C D], (where A B are dense matrices and C D are sparse matrices.)
2.) To implement matrix indexing in sparse objects. This will make it possible for the user to access and operate on individual elements in sparse objects.

Implementing the first goal should make *P easier to use, particularly for the users who already accustomed to matrix manipulation in MATLAB. Implementing the second goal is essential for implementing complete sparse object support in *P.

2.) Project Plan

For this project, we need to modify the dense object class files (.m file) and change the communication between the client and server (C files). Also, we need to implement all the operations related to sparse objects, including construction and matrix indexing. The workload of this project is shown in the following table.

Object	Constructor	Matrix indexing
Dense	Modify/change	N/A
Sparse	Write new code	Write new code

Schedule

1.) 4/26 – 5/3 Read *P code, become accustomed to hacking the kernel and understand the structure of the current code
2.) 5/4 – 5/10 Implement all the constructors for dense objects.
3.) 5/11 – 5/ 25 Implement sparse constructors and sparse matrix indexing
4.) 5/25 – 5/30 Test and prepare final report