MATLAB*P: Architecture

Ron Choy, Alan Edelman
Laboratory for Computer Science
MIT
Outline

• The “p” is for parallel
• MATLAB is what people want
Supercomputing in 2003

• The impact of the Earth Simulator
• The impact of the internet bubble
• The rise of clusters
• The few big players left
The Rise of Clusters

• Cheap & everyone seems to have one
### Some MIT Clusters

<table>
<thead>
<tr>
<th>Site</th>
<th>Name</th>
<th>Processors/Nodes</th>
<th>Type</th>
<th>Memory</th>
<th>Interconnect</th>
<th>Purpose</th>
<th>Built/Hardware updated</th>
<th>Maintained</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCS Computer Architecture Group</td>
<td>cagfarm</td>
<td>100/50</td>
<td>1GHz to 2.2GHz P4 Xeon</td>
<td>2G x 50 = 100GB</td>
<td>Gigabit Ethernet</td>
<td>Research</td>
<td>??</td>
<td>Tech Square Inc.</td>
</tr>
<tr>
<td>Parsons Laboratory (Building 49)</td>
<td>daily.mit.edu</td>
<td>84/42</td>
<td>Pentium 4, Xeon</td>
<td>84GB</td>
<td>Myrinet and Ethernet</td>
<td>Environmental data assimilation</td>
<td>2002</td>
<td>Internal to project + Aspen</td>
</tr>
<tr>
<td>LCS Computer Graphics Group - 1</td>
<td>photon</td>
<td>64/32/128 (HyperThreading)</td>
<td>Intel Xeon 2.4GHz</td>
<td>80GB</td>
<td>Gigabit Ethernet</td>
<td>Research</td>
<td>??</td>
<td>Department sysadmin</td>
</tr>
<tr>
<td>LCS Computer Graphics Group - 2</td>
<td>tessereae</td>
<td>64/32</td>
<td>Intel P4 400MHz FSB</td>
<td>20GB</td>
<td>Fast Ethernet</td>
<td>Research</td>
<td>??</td>
<td>Department sysadmin</td>
</tr>
<tr>
<td>Materials Science, Building 13</td>
<td>issmbc.mit.edu</td>
<td>60/30</td>
<td>Athlon 1900Xeon 2.0/Alpha 833</td>
<td>2G x 50 = 600GB</td>
<td>Ethernet/Myrinet</td>
<td>First-principles calculations in materials/Teaching SMA5107</td>
<td>2000</td>
<td>Postdocs and Students</td>
</tr>
<tr>
<td>CTP Lattice Group/LNS</td>
<td>npd.lns.mit.edu</td>
<td>48/12</td>
<td>Alpha EV67 666MHz</td>
<td>120GB</td>
<td>Myrinet</td>
<td>Research in LQCD</td>
<td>1998</td>
<td>Andrew Pochnisky</td>
</tr>
<tr>
<td>Laboratory for Nuclear Science</td>
<td>?</td>
<td>40/20</td>
<td>Pentium III @ 400 MHz</td>
<td>512MB x 20 = 10GB</td>
<td>Fast Ethernet</td>
<td>Scientific Computing for LNS community</td>
<td>1998</td>
<td>J. Maynard Gellinas</td>
</tr>
<tr>
<td>Applied Mathematics</td>
<td>headside</td>
<td>32/16</td>
<td>Athlon MP 2000+</td>
<td>1GB x 16 = 16GB</td>
<td>Gigabit Ethernet</td>
<td>Research in materials simulations, granular flows, electrosorption, percolation, fractal growth, and other physical systems.</td>
<td>Sept 2002</td>
<td>Department sysadmin</td>
</tr>
<tr>
<td>MIT Nuclear Engineering Department</td>
<td>echelon</td>
<td>30/30</td>
<td>Athlon 1700 1.1GHz 0C 1.5GHz</td>
<td>0.5G * 30 = 15GGB FC2100</td>
<td>Fast Ethernet</td>
<td>Monte Carlo N Particle code for reactor criticality</td>
<td>2002</td>
<td>Graduate Student</td>
</tr>
<tr>
<td>LCS Applied Computing Group</td>
<td>beowulf.ics</td>
<td>16/8</td>
<td>Athlon MP 1800+</td>
<td>16G x 8 = 80GB</td>
<td>Fast Ethernet</td>
<td>Research and Teaching 18.3376.338/SM05605</td>
<td>Jan 2002</td>
<td>Ron Choy</td>
</tr>
<tr>
<td>LCS Computer Graphics Group - 3</td>
<td>?</td>
<td>8</td>
<td>MIPS R10000/R12000 500MHz</td>
<td>6GB</td>
<td>Fast Ethernet</td>
<td>Research</td>
<td>??</td>
<td>Department sysadmin</td>
</tr>
<tr>
<td>LIDS Optical &amp; Satellite Networking Group</td>
<td>throbac.mit.edu</td>
<td>8/14</td>
<td>Athlon MP 2100+</td>
<td>2G x 4 = 8GB</td>
<td>Fast Ethernet</td>
<td>Research in Linear Programming, Optimization, Convex analysis, Network simulations</td>
<td>Nov 2002</td>
<td>Departmental sysadmin/Grad Students</td>
</tr>
</tbody>
</table>
Rise of Clusters

• A Beowulf is in the top 5 of the top500 list!
• Increased computing power does not necessarily translate to increased productivity
• Someone has to program these parallel machines, and this can be hard
Classes of interesting problems

• Large collection of small problems – easy, usually involves running a large number of serial programs – embarrassingly parallel

• Large problems – might not even fit into the memory of a machine. Much harder since it usually involves communication between processes
Parallel programming
(1990 – now)

• Dominant model: MPI (Message Passing Interface)
• Low level control of communications between processes
• Send, Recv, Scatter, Gather …
MPI

• It is not an easy way to do parallel programming.
• Error prone – it is hard to get complex low level communications right
• Hard to debug – MPI programs tend to die with obscure error messages. There is also no good debugger for MPI.
MPI (2)

• Show some MPI code here …
The result?

- A lot of researchers in the sciences need the power of parallel computing, but do not have the expertise to tackle the programming
- Time saved by using parallel computers is offset by the additional time needed to program them
The result? (2)

• A high level tool is needed to hide the complexity away from the users of parallel computers
Let’s go back to the serial world for a moment
Outline

• The “p” is for parallel
• MATLAB is what people want
What do people want?

• Real applications programmers care less about “p-fold speedups”
• Ease of programming, debugging, correct answer matters much more
MATLAB

• MATrix LABoratory

• Started out as an interactive frontend to LINPACK and EISPACK

• Has since grown into a powerful computing environment with ‘Toolboxes’ ranging from neural networks to computational finance to image processing.

• Many current users know nothing about linear algebra
MATLAB ‘language’

- MATLAB has a C-like scripting language, with HPF-like array indexing
- Interpreted language
- A wide range of linear algebra routines
- Very popular in the scientific community as a prototyping tool – easier to use than C/FORTRAN
MATLAB ‘language’ (2)

- MATLAB makes use of ATLAS (optimized BLAS) for basic linear algebra routines
- Still, performance is not as good as C/FORTRAN, especially for loops
- However MATLAB is catching up with its release 13 – (just-in-time compiling?)
Why people like MATLAB?

• Convenience, not performance!
Wouldn’t it be nice to mix the convenience of MATLAB with parallel computing?
MATLAB*P

- Project started by P. Husbands, who is now at NERSC
- Goal: To provide a user-friendly tool for parallel computing
- Uses MATLAB as a front end to a parallel linear algebra server
MATLAB*P (2)

- Server leverages on popular parallel numerical libraries:
  - ScaLAPACK
  - FFTW
  - PARPACK
  - …
Structure of the System
Structure of the system (2)

- **Client (C++)**
  - Attaches to MATLAB through MEX interface
  - Relays commands to server and process returns
- **Server (C++/MPI)**
  - Manipulate/process data (matrices)
  - Perform the actual parallel computation by calling the appropriate library routines
Functionalities provided

- PBLAS, ScaLAPACK
  - solve, eig, svd, chol, matmul, +, -, ...
- FFTW
  - 1D and 2D FFT
- PARPACK
  - eigs, svds
- Support provided for s,d,c,z
Focus

- Require minimal learning on user’s part
- Reuse of existing scripts
- Mimic MATLAB behaviour
- Data stay on backend until explicitly retrieved by user
- Extendable backend
What is *p? 

- MATLAB*P creates a new MATLAB class called layoot 
- By providing overloaded functions for this new class, parallelism is achieved transparently 
- *p is layoot(1)
Example

- \( A = \text{randn}(1024*p,1024*p); \)
- \( E = \text{eig}(A); \)
- \( e = \text{pp2matlab}(E); \)
- \( \text{plot}(e,'*'); \)
Example 2

- \( H = \text{hilb}(n) \) \quad \% \ H = \text{hilb}(8000*p) \\
  \quad \begin{align*}
  & J = 1:n; \\
  & J = J(\text{ones}(n,1),:); \\
  & I = J'; \\
  & E = \text{ones}(n,n); \\
  & H = E./(I+J-1); 
\end{align*}
Built-in MATLAB functions

• Hilb is a built-in MATLAB function that get parallelized automatically
• Another example of a built-in MATLAB function that works is cgs
• We do not know how many of them works
Where is the data?

- One often-asked question in parallel computing is: where is the data residing?
- MATLAB*P supports 3 data distributions:
  - $A = \text{randn}(m*p,n)$ – row distribution
  - $B = \text{randn}(m,n*p)$ – column distribution
  - $C = \text{randn}(m*p,n*p)$ – 2D block cyclic distribution
Flow of a MATLAB*P call

- User type $A = \text{randn}(256*p)$ in MATLAB
- The overloaded randn in layoot class is called
- The overloaded method calls MATLAB*P client with the arguments ‘ppbase_addDense’, 256, 3 (for cyclic dist)
- Client relays the command to server
Flow of a MATLAB*P call (2)

- Server receives the command and pass it on to PackageManager
- PackageManager finds appropriate function in package PPBase, makes the call
- A 2D block cyclic distributed randn matrix is created
- Server pass matrix handle to client, which passes it to MATLAB. Handle is stored in A.
MultiMATLAB mode

• Similar to MultiMATLAB from Cornell
• Start up a MATLAB process on each node, and run MATLAB scripts on them in parallel
• Interesting feature is that the data can come from the ‘regular’ mode of MATLAB*P
Example 3

- \( d = \text{randn}(10000 \times p, 1); \)
- \( a = 1:9999 \times p; \) % created in ‘regular’ mode
- \( z = \text{mm}('\text{chi2rnd}',a); \)
- \( e = \text{stebz}(d,z) \)
Types of parallel problems

• Collection of small problems
  – MultiMATLAB mode

• Large problems
  – ‘Regular mode’ – ScaLAPACK, FFTW, …

• We even allow a mixture of the two modes to give additional flexibility
Visualization package

- A term project done by a group of students in a parallel computing class at MIT
- Provides the equivalent of mesh, surf and spy to distributed matrices
- Visualization of very large matrix is now possible in MATLAB
Why parallel MATLAB?

- Why not Maple/Mathematica?
  - MATLAB*P is not really tied to MATLAB – the server does not understand a single line of MATLAB
  - Interfaces could be written for other software ‘easily’
  - We choose MATLAB because it is widely used and easy to use – we want to inherit this property
  - Parallelizing maple/mathematica is a much different problem than parallelizing matlab and probably would have a smaller audience
Why is this interesting?

• Overhead is incurred!
  – Reduction in programming time compensates the overhead
  – Makes large numerical experiments easy
More

- Kinds of matlab users: 0 matlabs, 1 matlab, many matlabs
- Processor maps
- Dynamic addition of processors
- Matlab*p on a grid