Parallel Computing with MATLAB® and Simulink®

Dr. Frank M. Graeber
Application Engineering Group
The MathWorks GmbH
Solving Large Technical Problems

**Difficulties**

- Long running
- Computationally intensive
- Large data set

**You could...**

- Wait
- Reduce size of problem

**Solutions**

- Run similar *tasks* on independent processors in *parallel*
- Load *data* onto multiple machines that work together in *parallel*
Development of Parallel Code

Challenges

Jobs often run in scheduled mode

Hard to debug

Cannot access intermediate answers

Hard to diagnose bottlenecks in algorithm

Solution

Work *interactively* in parallel
Easier Parallel Programming

Example: Transposing a Distributed Matrix

P>> D = distributed(A)

P>> E = D'

Using Fortran and MPI

Using MATLAB and MPI

Using Distributed Arrays
Parallel Computing with MATLAB

No code changes

- Toolbox Support:
  - Optimization Toolbox™
  - Genetic Algorithm and Direct Search Toolbox™
  - SystemTest™

Trivial changes

- Parallel for-loops
- Distributed arrays
- Scheduling jobs and tasks

Extensive changes

- MATLAB with Message Passing
Task Parallel for Distributing Tasks

Processes / Tasks

Time

Processes / Tasks

Time
Demo: Monte Carlo Simulation of Coin Tossing

10 Simulations of Flipping 20 Coins at a Time

Number of Heads Out of 20

11 7 12 15 7 9 9 7 8 12
Parallel for-Loops

```matlab
parfor i = 1 : n
    % do something with i
end
```

- Mix task-parallel and serial code in the same function
- Run loops on a pool of MATLAB resources
- Iterations must be order-independent
- M-Lint analysis helps in converting existing `for`-loops into `parfor`-loops
Distributing Jobs with SystemTest
Data Parallel for Handling Large Data Sets
Demo: Conjugate Gradient Benchmark
Distributed Arrays and Parallel Algorithms

- Distributed arrays
  - Store segments of data across participating workers
  - Create from any built-in class in MATLAB
    - Examples: doubles, sparse, logicals, cell arrays, and arrays of structs

- Parallel algorithms for distributed arrays
  - Matrix manipulation operations
    - Examples: indexing, data type conversion, and transpose
  - Parallel linear algebra functions, such as `svd` and `lu`
  - Data distribution
    - Automatic, specify your own, or change at any time
>150 Enhanced MATLAB Functions That Operate on Distributed Arrays

<table>
<thead>
<tr>
<th>Type of Function</th>
<th>Function Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data functions</td>
<td>cumprod, cumsum, fft, max, min, prod, sum</td>
</tr>
<tr>
<td>Data type functions</td>
<td>cast, cell2mat, cell2struct, celldisp, cellfun, char, double, fieldnames, int16, int32, int64, int8, logical, num2cell, rmfield, single, struct2cell, swapbytes, typecast, uint16, uint32, uint64, uint8</td>
</tr>
<tr>
<td>Elementary and trigonometric functions</td>
<td>abs, acos, acosh, acot, acotd, acoth, acsc, acscd, acsch, angle, asec, asecd, asech, asin, asind, asinh, atan, atan2, atand, atanh, ceil, complex, conj, cos, cosd, cosh, cot, cotd, coth, cssc, csd, csch, exp, expm1, fix, floor, hypot, imag, isreal, log, log10, log1p, log2, mod, nextpow2, nthroot, pow2, real, reallog, realpow, realsqrt, rem, round, sec, secd, sech, sign, sin, sind, sinh, sqrt, sqrt2, tand, tanh</td>
</tr>
<tr>
<td>Elementary matrices</td>
<td>cat, diag, eps, find, isempty, isequal, isequalwithnan, isfinite, isinf, isnan, length, ndims, size, str2func, triu</td>
</tr>
<tr>
<td>Matrix functions</td>
<td>chol, eig, lu, norm, normest, svd</td>
</tr>
<tr>
<td>Array operations</td>
<td>all, any, bitand, bitor, bitxor, ctranspose, end, eq, ge, gt, horzcat, ldivide, le, lt, minus, midivide, mrdivide, mtimes, ne, not, or, plus, power, rdivide, subsasgn, subsindex, subsref, times, transpose, uminus, uplus, vertcat, xor</td>
</tr>
<tr>
<td>Sparse matrix functions</td>
<td>full, issparse, nnz, nonzeros, nzmax, sparse, spfun, spzeros</td>
</tr>
<tr>
<td>Special functions</td>
<td>dot</td>
</tr>
</tbody>
</table>
MPI-Based Functions in Parallel Computing Toolbox™

Use when a high degree of control over parallel algorithm is required

- High-level abstractions of MPI functions
  - `labSendReceive`, `labBroadcast`, and others
  - Send, receive, and broadcast any data type in MATLAB

- Automatic bookkeeping
  - Setup: communication, ranks, etc.
  - Error detection: deadlocks and miscommunications

- Pluggable
  - Use any MPI implementation that is binary-compatible with MPICH2
Options for Scheduling Jobs

Task Parallel

```matlab
>> createMatlabPoolJob
or
>> batch

>> createJob(…)
>> createTask(…)
```

Data Parallel

```matlab
>> createParallelJob
```
Parallel Computing with MATLAB

Pool of MATLAB Workers
Run **Four Local** Workers with a Parallel Computing Toolbox License

- Easily experiment with explicit parallelism on multicore machines
- Rapidly develop parallel applications on local computer
- Take full advantage of desktop power
- Separate computer cluster not required
Scale Up to Cluster Configuration with No Code Changes
Configurations

- Save environment-specific parameters for your cluster(s)

Benefits
- Enter cluster information only once
- Modify configurations without changing MATLAB code
- Apply multiple configurations when running within same session
Components / Terminology

- **MATLAB® Distributed Computing Server™**
- **Parallel Computing Toolbox**
- **Third-Party Scheduler**
- **Result**
- **Worker**
- **Client Machine**

Compute Cluster

MATLAB® Distributed Computing Server™
Three User Communities

PERSONAL SUPERCOMPUTING WITH MATLAB

- Technical Computing User: Larger Data and Faster Processing
- System Administrator: Hardware Utilization and Optimal License Use
- HPC User: Easier Programming

Languages:
- Fortran
- C
Documentation on the The MathWorks Website