MATLAB Parallel Computing

Sida Peng

Cornell University

Aug. 15 2014

Outline

MATLAB

- MATLAB is a computing environment that is halfway between a programming language (where a user must do everything) and a menu-driven application (where the user only makes high level decision)
- Users always have the ability to lay out the precise details of an algorithm themselves.
- They rely on MATLAB commands to access intelligent, flexible, and optimized versions of standard algorithms.

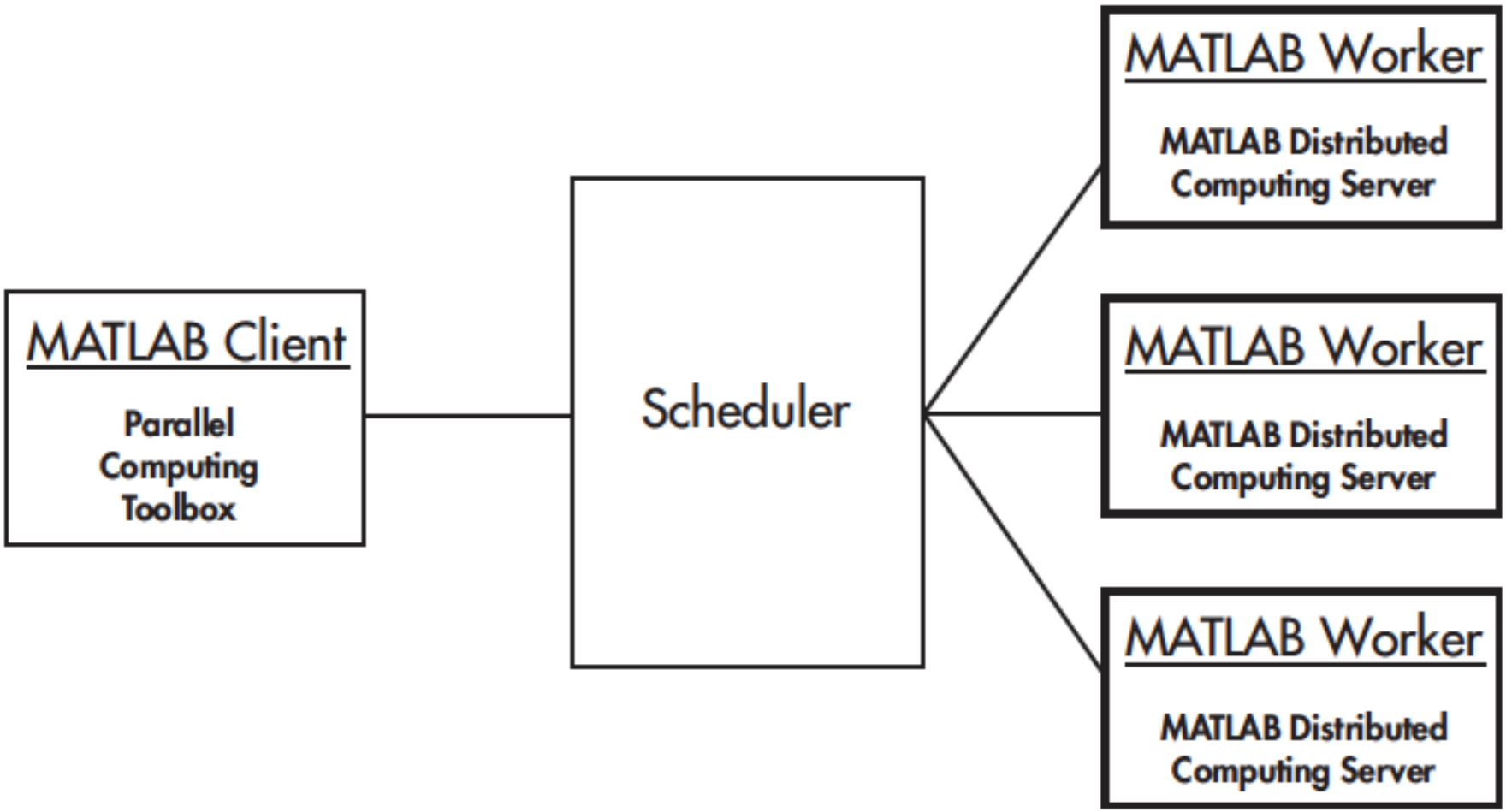
Parallel Computing Toolbox

- MATLAB has developed a Parallel Computing Toolbox which is required for all parallel applications.
- Parallel Computing Toolbox lets you solve computationally and data-intensive problems using multicore processors, GPUs, and computer clusters.

Parallel Computing Toolbox

- The toolbox allows a user to run a job in parallel on a desktop machine, using up to 4 ”worker” (additional copies of MATLAB) to assist the main copy (client copy).
- If the desktop machine has multiple processors, the workers will activate them, and the computation should run more quickly.
- This use of MATLAB is very similar to the shared memory parallel computing enabled by OpenMP, However, MATLAB requires much less guidance from the user.

Classification of Variables



Parallel Computing Toolbox

Key Problems Addressed by Parallel Computing:

- Run Parallel for-Loops (parfor)
  Parallel Computing Toolbox software improves the performance of loop execution by allowing several MATLAB workers to execute individual loop iterations simultaneously

Parallel Computing Toolbox

- Execute Batch Jobs in Parallel
  When working interactively in a MATLAB session, you can offload work to a MATLAB worker session to run as a batch job. The command to perform this job is asynchronous, which means that your client MATLAB session is not blocked, and you can continue your own interactive session while the MATLAB worker is busy evaluating your code. The MATLAB worker can run either on the same machine as the client, or if using MATLAB Distributed Computing Server, on a remote cluster machine.

Parallel Computing Toolbox

- Partition Large Data Sets (spmd)
  If you have an array that is too large for your computers memory, it cannot be easily handled in a single MATLAB session. Parallel Computing Toolbox software allows you to distribute that array among multiple MATLAB workers, so that each worker contains only a part of the array.

Distributed Computing Server

- MATLAB has developed a Distributed Computing Server or DCS.
- Assuming the user's code runs properly under the local parallel model, then it will also run under DCS with no further changes.
- With the DCS, the user can start a job on the desktop that gets assistance from workers on a remote cluster.
- MATLAB includes a ”batch” command that allows you to write a script to run a job (parallel or not, remote or local) as a separate process.
- This means we can submit a script for running a remote job and then exit the local copy of MATLAB. We can check back later on the remote job status and retrieve the results.

Outline

Parallel Computing

You'll need

- A multicore machine
- Right version of MATLAB (version 2008a or later)
- A copy of the Parallel Computing Toolbox (Type "ver" to see a list of all your toolboxes)
- the source code of your MATLAB program.

Parallel Computing

To do local parallel programming:

- Start MATLAB the regular way. (This copy of MATLAB is called client copy)
- Request a number of extra copies of MATLAB (known as ”workers” or ”labs”)
- Issue the normal command to run the program. The client MATLAB will call on the workers for help as needed
- release the workers

Example

```
1 matlabpool open
2 clear A
3 parfor i = 1:8
4 A(i) = i;
5 end
6 matlabpool close
```

Parallel Computing

- If all is well, the program runs as before
- Output will still appear in the command window in the same way and the data will be available to you.

Outline

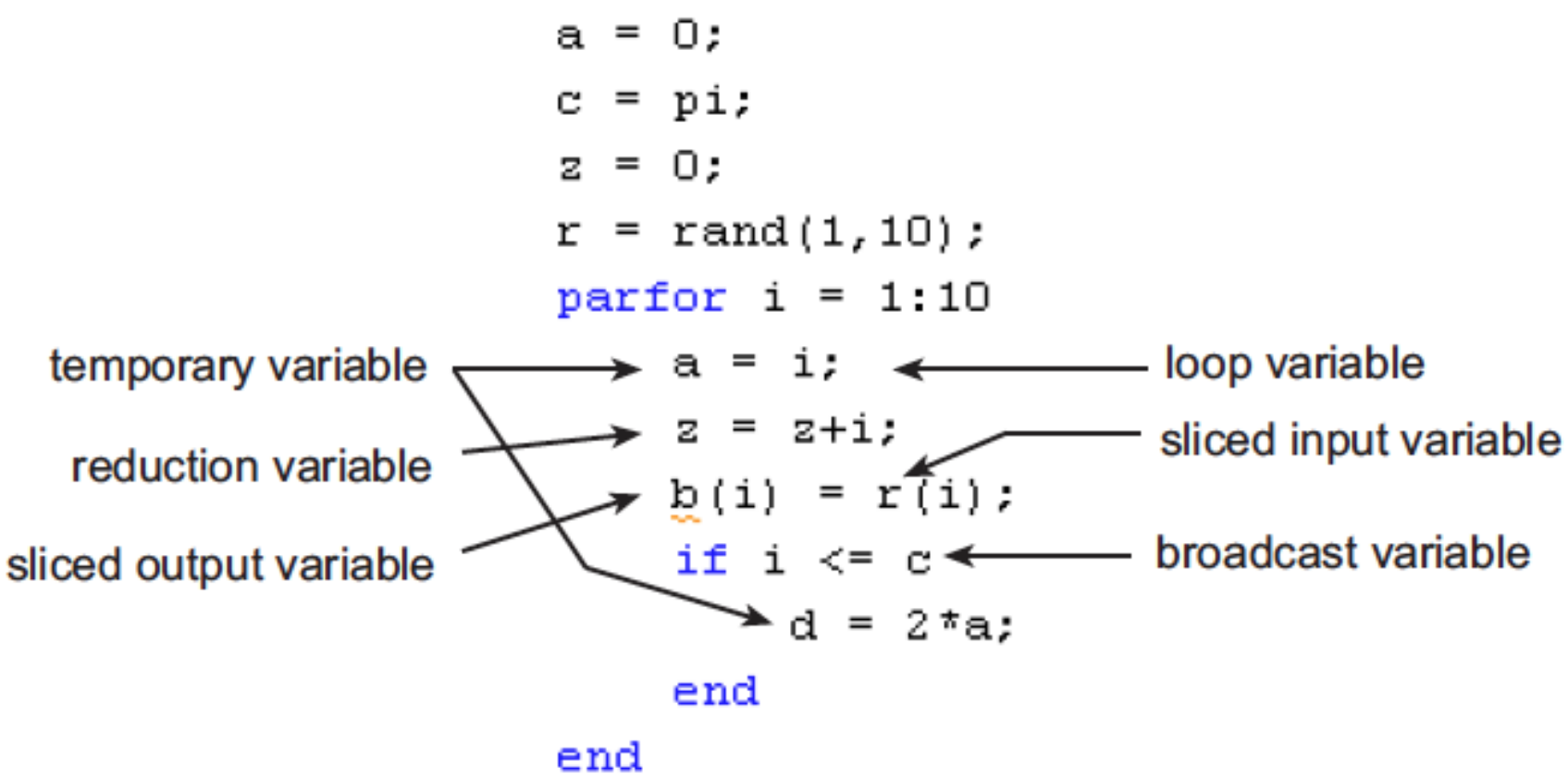
Parfor command

- The simplest way of parallelizing a MATLAB program focuses on the for loops in the program.
- If a for loop is suitable for parallel execution, replace the word ”for” with ”parfor”
- When the MATLAB program is run in parallel, the work in each parfor loop will be distributed among the workers.

Parfor command

- The crucial point is weather a ”for” loop is suitable for parallelization!
- To find out, one need to answer the following question:
- Can the iterations of the loop e performed in any order without affecting the results?
- If the answer is ”yes”, then generally the loop can be parallelized.

Classification of Variables



Classification of Variables

- Loop variable serves as a loop index for arrays
- Assignments to the loop variable are not allowed.
- A sliced variable is one whose value can be broken up into segments, or slices, which are then operated on separately by workers and by the MATLAB client.
- A broadcast variable is any variable other than the loop variable or a sliced variable that is not affected by an assignment inside the loop.
- A temporary variable is any variable that is the target of a direct, non-indexed assignment, but is not a reduction variable.

Reduction Variables

- In some cases, special care must be taken for so called ”reduction variables”.
- A reduction variable accumulates a value that depends on all the iterations together, but is independent of the iteration order.
- Typically, these occur when certain functions such as the max, min, sum, or prod are applied to an indexed loop value.
- Although loop iterations are not completely independent in such a calculation, MATLAB can parallelize a loop contains reduction variables.
- For any reduction variable, the same reduction function or operation must be used in all reduction assignments for that variable.

Control Random Number Streams

- By default, each worker in a cluster working on the same job has a unique random number stream.
- If you need all workers to generate the same sequence of numbers, you can seed their generators all the same.
- For example:

```
1  s = RandStream('twister');  % Default seed 0
2  RandStream.setGlobalStream(s);
```

- Do not use rng('shuffle')

Miscellaneous

- The MATLAB workers correspond to separate threads of execution. But every worker can see or change any variable. That is so called ”shared memory model”
- It is important to initialize all local variables and avoid re-defining of all global variables.
- MATLAB does not require a pre-specification of local or private variables. It will attempt to do this for your automatically. But sometime this results error!
- A loop containing ”break” or ”return” cannot run in parallel
- Parallelism doesn’t pay until your problem is big enough
  For example, number of iterations; time of each iteration.
- Parallelism doesn’t pay until you have a decent number of workers.
  For example, the difference between 0 worker and 1 worker; the difference between 0 worker but 12 workers.