



Parallelism Provided by MATLAB: A Survey

Divya Kundra*

Assistant Professor, Deen Dayal Upadhyaya College,
New Delhi, India

Abstract—MATLAB is a high level programming language used for various scientific and engineering calculations that is developed by MathWorks. It provides parallelism in 3 manners – multithreaded parallelism, explicit parallelism and distributed computing. This paper reviews these parallelism and also discusses the supporting toolboxes.

Keywords— Parallelism, CPU, Parallel Computing Toolbox, MATLAB Distributed Computing Server, Distributed Computing

I. INTRODUCTION

MATLAB (Matrix Laboratory) is a fourth generation programming language developed by MathWorks. It provides interactive environment for problem exploration and design, offers various mathematical functions and development tools for improving code and features for integrating program with programs written in other languages. Due to presence of microprocessors having multicore, MATLAB has evolved to provide feature of parallelism. MATLAB supports 3 kinds of parallelism: multithreaded, explicit parallelism and distributed computing. In multithreaded parallelism, some of MATLAB's inbuilt functions implicitly provide multithreading. A single MATLAB process generates multiple instruction streams. CPU cores execute these streams while sharing the same memory. In explicit parallelism, multiple instances of MATLAB run on separate processors often each with its own memory and simultaneously execute the code that externally invokes these instances. In distributed computing, multiple instances of MATLAB run independent computations on each computer, each with its own memory. There is a coexistence between them e.g. a distributed computing job might call multithreaded function on each machine and then use a distributed array to collect final results. Distributed Computing is an extension of explicit parallelism and can be easily extended to explicit parallelism using MATLAB Distributed Computing Server toolbox. Use of Parallel Computing Toolbox is done to make use of different cores of CPU and also to access the GPU.

A. Parallelism

In sequential programming, there is an ordered relationship of execution of instructions where only a single instruction executes at a particular instance of time. The program is executed over a single processor only. Serial implementation done on a single thread provides a base for evaluating comparisons from other models. In contrast to sequential processing, parallel processing lets execution of multiple tasks at the same time [1]. In parallel processing, the instructions are distributed to different processors which work simultaneously in order to complete the task fast. The ease and success of parallelism depends on how much synchronization exists between the divided tasks. Speedup will be maximum when tasks are independent i.e. there is no communication between tasks executing in parallel [2]. Parallel computing is done on multi-core computers.

B. Multithreaded Parallelism

In MATLAB by default implicit multithreading¹ is provided for expressions that are combinations of element wise operations. In this type of parallelism, multiple instruction streams are generated by one instance of MATLAB session. Multiple cores share the memory of a single computer to access these streams. The 3 basic requirements to achieve it are:

- Each element wise operation should be independent of each other.
- Size of data should be big enough so that speed up achieved by the concurrent execution exceeds the time required for partitioning and managing different threads.
- The function in consideration should preferably be complex and challenging enough so that higher speed ups are achieved while multithreading.

To fulfil these requirements, independent and big tasks vectorisation are essential for implicit parallel computations. Vectorisation is one of the most efficient ways of writing the codes in MATLAB [3]. It performs operations on large matrices through a single command at once instead of performing each operations one by one inside the for loop. An effective way of using multithreading is replacing for loops by vector operations. Code using vectorisation uses optimised multithreaded linear algebra libraries and thus generally run faster than its counterpart for loop. MATLAB

¹<http://www.mathworks.com/matlabcentral/answers/95958-which-matlab-functions-benefit-from-multi-threaded-computation>

uses implicit multithreading using commands such as- arrayfun. Syntax is as shown by Equation (1). arrayfun applies the function specified in function handle 'func' to each element of equal sized arrays. The order of execution of function on the elements is not specified, thus tasks should be independent of each other. In Equation (1) function 'func' is applied to corresponding cell elements of equal sized arrays A1, A2,..., An and the respective result is returned in B1,B2,...,Bm. The mechanism of arrayfun is shown in Figure 1 where independent threads apply function 'func' on corresponding elements of 2 arrays A1 and A2.

$$[B1, \dots, Bm] = \text{arrayfun}(\text{func}, A1, \dots, An) \quad (1)$$

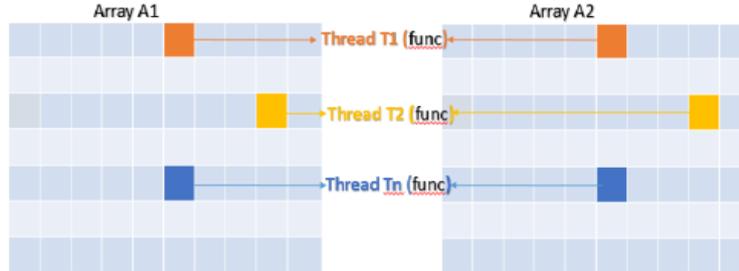


Fig1: arrayfun mechanism

C. Parallel Computing Toolbox

MATLAB provides the facility to apply explicit parallelism with Parallel Computing Toolbox (PCT). PCT lets to solve computationally and data intensive problems using multicore CPUs, GPUs and computer clusters. The basic requirement to apply parallelism is that tasks should be independent. The use of PCT is that without the help of lower level programming of CUDA or MPI, a novice user can very well apply principles of parallelism. The toolbox lets to use full processing power of multicore desktops by executing applications on different MATLAB computational engines. Without changing the code, the same application can be run over a computer cluster or grid computing using MATLAB Distributed Computing Server Toolbox.

D. Explicit Parallelism on CPU

MATLAB has the provision of applying external parallelism over set of independent tasks through parfor. parfor allows execution of the loop iterations in parallel on labs. Labs are workers which are executed on processor cores. Syntax of parfor is shown by Equation (2). 'loopvar' is the variable for which iterations occurs, 'initval', 'endval' are limits of the loop which must be finite integers and 'statements' represent the code required to be parallel.

$$\text{parfor loopvar} = \text{initval}:\text{endval}, \text{statements}, \text{end} \quad (2)$$

Using parfor separate process is created for each worker having its own memory and own CPU usage. Workers are headed by a client process which creates and manages them. When parfor is executed, the MATLAB client coordinates with the MATLAB workers which form a parallel pool. The code within the parfor loop is distributed to workers and it executes in parallel in the pool. The required data needed by workers to do the computations is send from client to all the workers and the results from all the workers are² collected back by client as shown in Figure 2. The body of the parfor is an iteration which is executed in no particular order by the workers. Thus, the loop iterations are needed to be independent of each other. If there is dependency between different loop iterations, an error will be produced on using parfor and the dependency is required to be removed in order to proceed. If number of iterations equals the number of workers in the parallel loop, each iteration is executed by one worker else a single worker may receive multiple iterations at once to reduce the communication overhead. Thus parfor distributes loop iterations in chunks to be executed by the worker. parfor can be useful in situations where there are many iterations of simple calculation loop that can distributed to a large number of workers so that each workers completes some portion of the total iterations. It can also be used in cases where loop iteration take long time to execute so that by simultaneous execution by multiple workers it can be completed faster. To start a pool of workers Equation (3) is used where name of the parallel pool forming a cluster is to be specified in the 'profilename' and size of the cluster in the 'poolsize' argument.

$$\text{parpool}(\text{profilename}, \text{poolsize}) \quad (3)$$

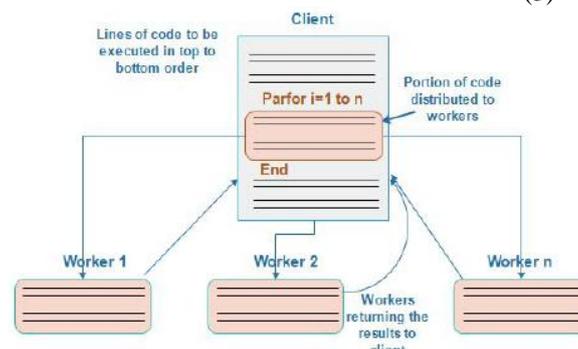


Fig2: parfor mechanism

¹ <http://in.mathworks.com/help/distcomp/parfor.html>

E. Distributed computing

MATLAB Distributed Computing Server solves computational intensive problems on computer clusters, clouds and grids. Using Parallel Computing Toolbox, the program is modeled on multicore desktop computer and then it is scaled upto many computers using MATLAB Distributed Computing Server toolbox as shown in Fig 3¹. The toolbox supports batch jobs, parallel computations and distributed large data. The server includes a build in cluster job scheduler and also provides support for commonly used third party schedulers. Some important specifications² for cluster computers running MATLAB workers as specified by are :

- Around 10 GB of disk space is required to accommodate a typical complete installation of MATLAB Distributed Computing Server.
- Maximum of 1 MATLAB worker per CPU core is recommended.
- Minimum of 1 GB RAM per MATLAB worker is recommended while running workers locally on the desktop. A worker consumes memory equivalent to a MATLAB session running without the MATLAB desktop.
- Several TCP ports are required by MATLAB workers for various worker services and for inter-worker communication.
- Several TCP ports are required by MATLAB workers for various worker services and for inter-worker communication.
- For interactive parallel computations, MATLAB workers running on cluster computers must be able to connect to MATLAB session running on user desktop via TCP. This is not required for running applications in batch.
- A shared file system between user desktops and cluster is strongly recommended. Availability of a shared file system is assumed by default for all the built-in configurations. An API is available to extend these configurations for environments with nonshared file systems.

There is also the facility to connect client machine to MATLAB Distributed Computing Server on the Amazon EC2 Cloud. Its important specifications that needs to be taken care into account are :

- All the outgoing connections by the client machine must be made in domain name- mathworks.com and in amazonaws.com on port 443 (https).
- There should always be a TCP connection between the client and the cluster running on cloud service
- The cloud center should be configured so as to allow connections from external computer's IP address.
- It is required that client machine must be able to make outgoing connections to cluster machines in amazonaws.com directly on ports 27355 and 4000 to 40001 to 4*N, where N is maximum number of workers on a single node.

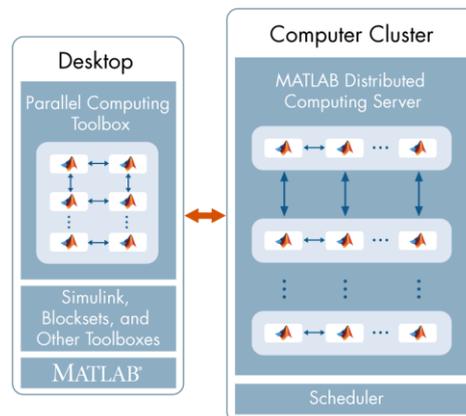


Fig3: Parallel computing with MATLAB- Parallel Computing Toolbox and MATLAB Distributed Computing Server.

II. CONCLUSION

This paper gives the different parallelism techniques supported by MATLAB. Review of all kinds of parallelisms-implicit via multithreading, explicit, using GPU and distributed computing is done. The supporting toolboxes-Parallel Computing Server and Distributed Computing Server are also discussed.

REFERENCES

- [1] Vipin Kumar. Introduction to Parallel Computing. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 2002.
- [2] G. S. Almasi and A. Gottlieb. Highly Parallel Computing. Benjamin-Cummings Publishing Co., Inc., Redwood City, CA, USA, 1989
- [3] Desmond J. Higham and Nicholas J. Higham, Matlab Guide. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2005³

¹http://in.mathworks.com/cmsimages/62006_w1_mdcs_fig1_w1.png

²<http://in.mathworks.com/products/distriben/requirements.html#matlab-workers>