Parallel Processing Problem and Solution - A Case Study on MATLAB Parallel Computing Toolbox Distributed Arrays

Wan Asywad bin Zainal Abidin [#1], Mohamed Faidz Mohamed Said [#2]

[#] *Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA*
70300 Seremban, Negeri Sembilan, MALAYSIA

[1] asywadzainal@gmail.com

[2] faidzms@ieee.org

***Abstract*—MATLAB is a software that provides a programming language used for calculations of the matrix, developing and running algorithms, creating user interfaces and visualisation. It is a matrix laboratory for solving many technical computing problems with its applications system. Its matrix laboratory is used to solve many technical computing problems with its applications system. The purpose of this research is to provide understanding about the Parallel Computing Toolbox (PCT) in MATLAB. Moreover, there are several related benefits of using parallel processing involving distributed arrays. During a large data processing, a great amount of computing power is required to process the data. The PCT is applied to present more solutions that distributed arrays can trade-off to easily provide the required performance of large data by using the toolbox provided. Finally, MATLAB parallel computing provides the needed solution for any given problem without taking extra time and cost. Thus, the uses of parallel processing or parallel computing have become one of the prevalent common ways to help users in solving many application problems nowadays.**

***Keyword:* MATLAB, distributed arrays, Parallel Computing Toolbox (PCT)**

I. INTRODUCTION

In MATLAB, many functions operate on distributed arrays rather than operate in arrays that are occupied in a single workspace [1]. Some of the functions cannot fully work in distributed arrays because there might exhibit certain limitations on that functions. Piotr Luszczek in his article Parallel Programming in MATLAB said, "The trend so far has been to double the number of cores every few years. This translates into a doubling of computational power. Harnessing that power will require the right software and writing that software will need the right software tools" [2]. The distributed arrays offered in MATLAB is one of the alternatives for saving some memory in computer or machine to give support while processing data.

A. Definition

Some terms need proper technical definitions for the research contents. Distributed array is defined as one variable which is splitting over its multiple workers. This is called as an entity. The distributed arrays have main interest on a cluster which combines memory between many machines [3].

MATLAB is a programming language that is used for calculations of the matrix, developing and running algorithms, creating user interfaces and visualisation. It is a matrix laboratory for solving many technical computing problems with its applications system.

II. HISTORICAL BACKGROUND

Distribution arrays are one type of array which is generated and being accessed in MATLAB. According to Grothoff, Palsberg et al., 2007, there is a data structure that is primarily used in language, that is distributed arrays [4]. In the research, they also contribute that array index is called a point. Then, the set of array indices for the terminology is known as a region. The data location in storage or the code executed is called a place. Lastly, the mapping from region to places is known as a distribution. This is to elevate the location of individual array elements to a language concept. Distributed array is used to fit in some memory which is too large for a single machine [5]. By using this distributed array, MATLAB can improve the speed of process for the data to be stored in a single machine with a collection of data. Another name called for distribution arrays are distribution shared memory. Moreover, a system for distribution memory system is known as multicomputer that is defined as consisting of multiple independent processing nodes with modules of local memory. This module is connected as a general interconnection network [6].

There are two steps for creating distributed arrays. The first step is creating and loading from data-store. This step will partition data among the works of MATLAB. These can help to reduce the size of the file that is too large in the memory. The second step is to use array or matrix creation functions such as zeros, ones, randn and rand. This step can just be appending distributed after the usual arguments for the functions.

In MATLAB, there are over 400 existing functions enhanced and overloaded such as math, matrix manipulations, signal processing and linear algebra. It supports for both dense and sparse arrays. Furthermore, it can support for new data types such as distributed table, distributed date-time arrays and distributed string [7].

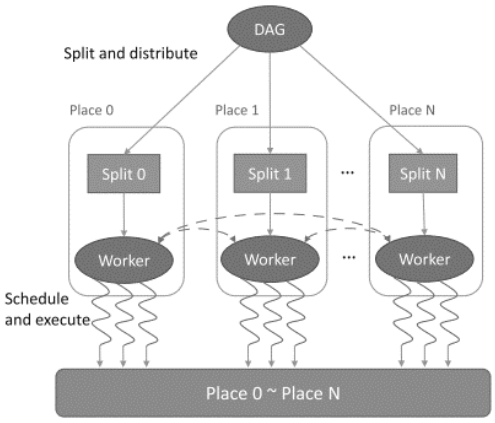

Fig. 1. DPX10 framework for logical flow

According to Wang, et al. [8], the distributed arrays have been used to improve by reducing the time taken by the parallel programming process. It is classified according to several parallel computing models which are shared memory model and distributed memory model. Fig. 1 above shows the logical flow that distributes by using a framework of DPX10. This distributed memory model has a good scalability, but the cost for communication in this model is quite high. The shared memory model is favoured to the data communication but it has a problem of scalability.

A fast CPU can be built by anyone, but it requires a fast system [9]. This relates to this project where distributed arrays are provided to increase the performance of computing system. When parallel programs are in the occurrence of the distributed arrays, it contains several constraints such as some processors and memory size [10].

Distributed memory model interacted with the connected component labelling (CCL) will become a key step for a wide-range application. There are several results presented for these applications through experimental and theoretical approaches. Through MATLAB, the process of distributed arrays can increase the memory to store the data structure which can improve the performance of the system [6].

III. METHODOLOGY

A. *A Type System for Distributed Arrays*

In this paper, distributed arrays are a key data structure in high-performance computing. The system that is functioning as a multicore system will have a specific programming language to enhance the process of processing data. The new language that has been designed for programmers such as X10, Fortress, Chapel, Titanium, Co-Array Fortran and ZPL has its functionality to process the data [2].

Language	X10	Fortress	Chapel	Titanium	Co-Array Fortran	ZPL
Array index	point	index	index	point	index	index
Set of array indices	region	n/a	domain	domain	n/a	region
Location	place	region	locale	demesne	image	n/a
Array distribution	distribution	distribution	distribution	distribution	n/a	n/a

Fig. 2. Language terminology

Based on X10 programming language depicted in Fig. 2, the distributed arrays are a key for the data structure to be in the high performance of computing system. These languages have one primary data which is distributed arrays. This distributed array is to improve the location of arrays element that is classified as an individual into a language concept.

X10 terminology is used for four basic notions associated with distributed arrays. The basic notations are array index called a point, a set of array indices called a region, a location where data is stored called a location and code is executed and a mapping from regions to places called a distribution.

Then, the research proposed that multi-core systems with non-memory are important in desktop and server computing. A location aware algorithm is executing on a particular core in concept to exploit the locality of distributed arrays [11].

B. *Exploiting Parallelism in Deterministic Shared Memory Multiprocessing*

This research shows that it uses parallelism by using Working Shared Memory (WSM) to create parallelism into threads. In additions, multi-block is requested for priority in each thread and the thread is controlled in the reverse order of their priorities.

Each WSM is allocated to a single thread to let the thread to achieve computation work. Then, the memories are structured to overlay for making the thread to run in parallel without data races. In parallel performance, there is a check by FPDet and it will control the memory access according to WSMs.

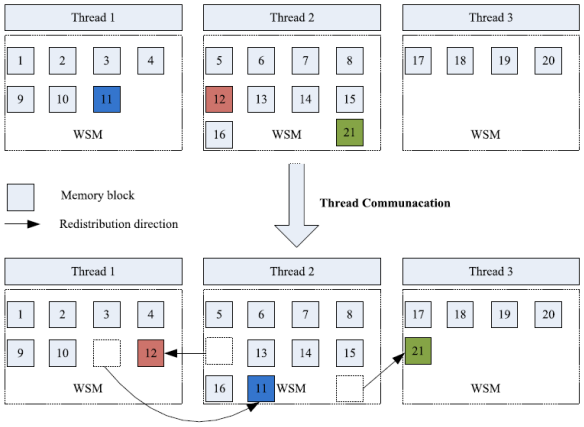

Fig. 3. Threads communication by redistributing WSMs

According to Fig. 3, the multiple pipelined thread pools where each pipelined thread pool consists of multiple threads and each thread takes a different role. The performance by Superscalar model is proposed to be the best compared to the other thread [12]. This shows the competitive performance between all the threads. The performance is compared to their number of processing core and the size of the cache. In this case, the Superscalar are in advantages because it has a large amount of processing core than others. Therefore, the improvement of the performance servers can help to solve many problems in parallel computing.

C. *Using Shared Arrays in Message-Driven Parallel Programs*

In this paper [5], it presents new chances outside the coupling bits of an application that wishes to connect through a shared array. A process is called giving separate modules accesses to commonly shared arrays [5]. In the parallel execution, the loose coupling for this process enables it to have a better flexibility and greater ease of multi-physics simulations.

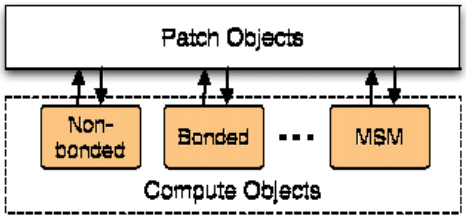

Fig. 4. A method for multi-level summation

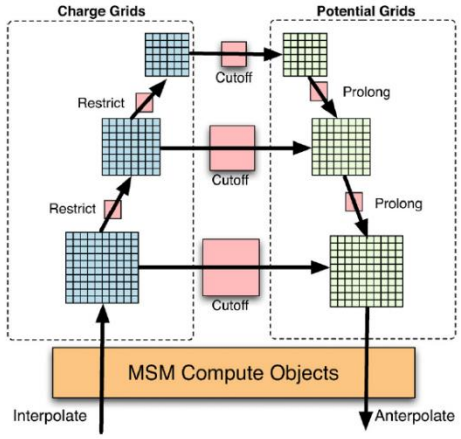

Fig. 5. Multi-level summation as implemented using collections of message-driven objects accessing MSAs

Fig. 4 and Fig. 5 above demonstrate the method for multi-level summations called MSM. This MSM gives the high potential performance. Besides that, the message-driven asynchronous call is a suitable and useful model for programs using parallel processing.

The relationship with this method provides distributed phased arrays. This phased array contains the collection of different antennas and also an element which is connected to a system. Besides, the distributed phased arrays have more benefits over the conventional arrays in many applications [13].

D. *Evaluation of Connected Component Labeling Algorithms for Distributed-Memory Systems*

There are several steps in distributed memory connected component algorithms. Based on Fig. 6, it calculates all locally connected component labelling (CCL) in each process. After the first step, the distributed memory will give local components globally a unique label. Then, the CAG (component adjacency graph) is generated by collapsing involved edges not to span their process boundaries. Lastly, the next step is to find the CCL of CAG. All these steps are to maximise the speed of processing. The process uses significantly less memory than the other alternatives [6].

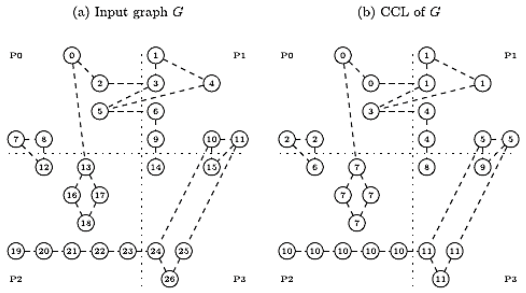

Fig. 6. An input graph, G, along with the corresponding CCL of the local graphs, labelled with globally unique labels - each process is assigned the vertices located in its quadrant

E. *Parallelizing with BDSC, A Resource-Constrained Scheduling Algorithm for Shared and Distributed Memory Systems*

In this paper, BDSC is used in the research by Yang and Gerasoulis's Dominant Sequence Clustering (DSC) which presents the speedups of the parallel MPI vs. sequential versions of ABF and quake using P = 2, 4 and 6 processors [9]. The research also presents that the generation of more clusters with empty slots led to higher process scheduling cost on processors, and higher communication volume between them when the number of processors is increased.

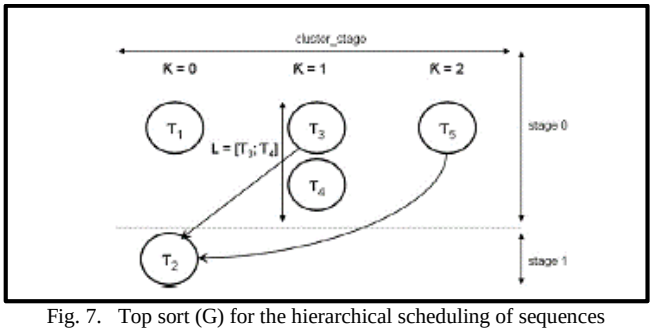

Fig. 7. Top sort (G) for the hierarchical scheduling of sequences

As depicted in Fig. 7, the parallelizing with BDSC is to find solutions based on the graph partitioning and scheduling problems. This is depending on the vertices that represent the computational tasks and edges and also data connections. In addition, distributed multi-processor system is a complex architecture that refers to a set of autonomous, interconnected multiprocessor machines with various capabilities [14].

IV. RESULT

Distributed arrays are one of the ways to improve the performance of processing data in MATLAB. For parallel computing, distributed arrays are required to give an advantage such as providing more memory while processing. Therefore, the case study in MATLAB using distributed arrays is a better way to free up some space in memory than others. The

distributed arrays also provide a high-level performance in PCT which is provided by MATLAB.

V. CONCLUSION

In this research, evaluation using another memory model is recommended such as shared memory model with this Parallel Computing Toolbox (PCT). It is expected that with the distributed memory, the computing system would provide more benefits because it has multi-core for its single machine. Besides, the PCT and Distributed Computing Server (DCS) are not economical based on the licenses.

Finally, the uses of parallel techniques are considered as mainly high-level, while the method of distributed arrays are more suitable for the novice user. Considering low-level parallel programming, as an expert user, is also very important to provide further insight on the potential of the PCT. Nevertheless, this was not possible in the research project due to the time constraints.

REFERENCES

[1] Wan Asywad Zainal Abidin. *"170522 CSC580 WAZA", 2017. [Online]*. Available: https://www.youtube.com/watch?v=LpzLX51gPH4 [Accessed : 19-Jun-2017].

[2] J. Dongarra, B. Tourancheau, and P. Luszczek, "Parallel Programming in MATLAB," *The International Journal of High Performance Computing Applications,* vol. 23, no. 3, pp. 277-283, 2009/08/01 2009.

[3] J.-Y. Cho, H.-W. Jin, M. Lee, and K. Schwan, "Dynamic Core Affinity for High-Performance File Upload on Hadoop Distributed File System," *Parallel Computing,* vol. 40, no. 10, pp. 722-737, 2014.

[4] C. Grothoff, J. Palsberg, and V. Saraswat, "A Type System for Distributed Arrays," Dept. of Computer Science University, IBM T.J. Watson Research Center, UCLA Computer Science Dept., 2007.

[5] P. Miller, A. Becker, and L. Kalé, "Using Shared Arrays in Message-Driven Parallel Programs," *Parallel Computing,* vol. 38, no. 1-2, pp. 66-74, 2012.

[6] J. Iverson, C. Kamath, and G. Karypis, "Evaluation of Connected-Component Labeling Algorithms for Distributed-Memory Systems," *Parallel Computing,* vol. 44, pp. 53-68, 2015.

[7] D. Houcque, "Introduction to Matlab for Engineering Students," p. 74, n.d. 2005.

[8] C. Wang, C. Yu, S. Tang, J. Xiao, J. Sun, and X. Meng, "A General and Fast Distributed System for Large-Scale Dynamic Programming Applications," *Parallel Computing,* vol. 60, pp. 1-21, 2016.

[9] D. Khaldi, P. Jouvelot, and C. Ancourt, "Parallelizing with Bdsc, A Resource-Constrained Scheduling Algorithm for Shared and Distributed Memory Systems," *Parallel Computing,* vol. 41, pp. 66-89, 2015.

[10] R. D. I. Keller, D. A. Kramer, and J.-P. Weiss, Facing The Multicore-Challenge III, Berlin ;: Springer, 2013. [Online]. Available: SpringerLink http://dx.doi.org/10.1007/978-3-642-35893-7.

[11] R. Ayari *et al.*, "Multi-Objective Mapping of Full-Mission Simulators on Heterogeneous Distributed Multi-Processor Systems," *Journal of Defense Modeling and Simulation: Application,* pp. 1–12, 2016.

[12] G. S. Choi and C. R. Das, "A Superscalar Software Architecture Model for Multi-Core Processors (Mcps)," *Journal of Systems and Software,* vol. 83, no. 10, pp. 1823-1837, 2010.

[13] David Jenn, Yong Loke, Tong Chin Hong Matthew, Yeo Eng Choon, O. C. Siang, and Y. S. Yam, "Distributed Phased Arrays and Wireless Beamforming Networks," *International Journal of Distributed Sensor Networks,* vol. 5, pp. 283–302, 2009.

[14] S. Varoglu and S. Jenks, "Architectural Support for Thread Communications in Multi-Core Processors," *Parallel Computing,* vol. 37, no. 1, pp. 26-41, 2011.