Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 5, May 2014, pg.815 – 820

RESEARCH ARTICLE

SERIAL COMPUTING vs. PARALLEL COMPUTING: A COMPARATIVE STUDY USING MATLAB

Abhay B. Rathod[1], Rajratna Khadse[2], M Faruk Bagwan[3]

Department of Electronics and Telecomm. Engg, Jawaharlal Darda Institute of Engg. and Tech. Yavatmal, India
[1] abhaybr@rediffmail.com; [2] khadserp@gmail.com; [3] mfbagwan@yahoo.com

Abstract— parallel computing has been around for many years but it is only recently that interest has grown due to the introduction of multi core processor at a reasonable price for the common people. The goal of this paper is to analyze and compare serial algorithm with parallel algorithm using parallel matlab toolbox.

Keywords— Central Processing Unit (CPU); Graphics Processing Unit (GPU) ; Parallel Computing Toolbox; Multicore Processor; parallelism

I. INTRODUCTION

The Parallel Computing is evolved from serial computing that attempts to emulate what has always been the state of affairs in natural World [6].

Serial and Parallel Computing [6]

Traditionally, software has been written for **serial** computation:

- To be run on a single computer having a single Central Processing Unit (CPU).
- A problem is broken into a discrete series of instructions.
- Instructions are executed one after another.
- Only one instruction may execute at any moment in time.

Parallel computing is the simultaneous use of multiple compute resources to solve a computational problem:

- To be run using multiple CPUs.
- A problem is broken into discrete parts that can be solved concurrently.
- Each part is further broken down to a series of instructions.
- Instructions from each part execute simultaneously on different CPUs or core.

MATLAB provides parallel computing via its Parallel Computing Toolbox. It solves computationally and data-intensive problems using multicore processors, GPUs, and computer clusters [12]. Parallel Computing Toolbox software allows us to run as many as eight MATLAB workers on our local machine in addition to our MATLAB client session [5].

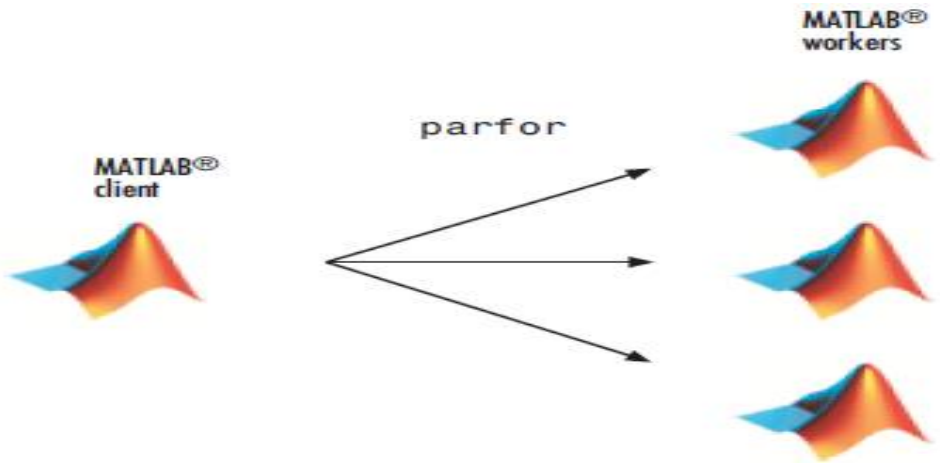

Fig. 1 Parallel Matlab using local workers for execution of task[5].

II. SERIAL VS. PARALLEL PROGRAMMING

A. Serial programming

Serial programming involves a consecutive and ordered execution of processes one after another. In other words with serial programming, codes or processes are run one after another in a succession fashion.

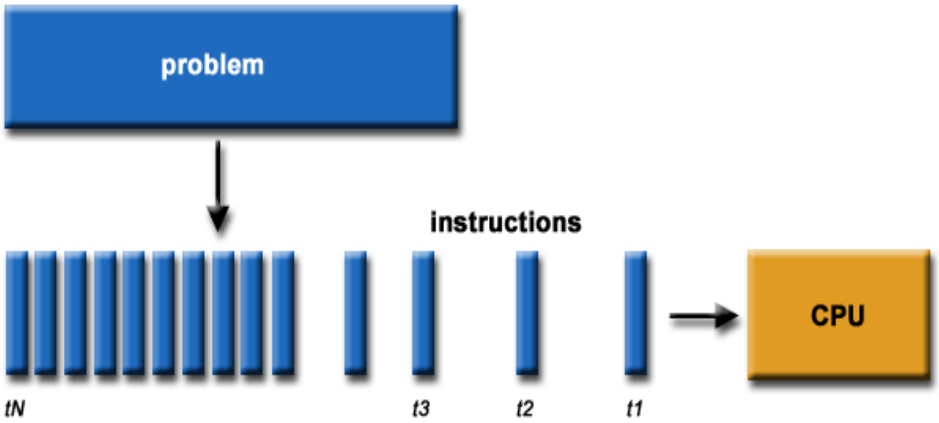

Fig. 1 Serial programming (Source: https://computing.llnl.gov/tutorials/parallel_comp/)

B. Serial coding

```
clc;
clear all;

min_matrix =500;
max_matrix =500;

serial = zeros([1 length(min_matrix:max_matrix)]);
count = 1;
for mat_size = min_matrix:max_matrix

   %Generate random matrices
   matrix1 = rand([mat_size mat_size]);
   matrix2 = rand([mat_size mat_size]);
   matrix1 = round(matrix1 * 100);
```



```
matrix2 = round(matrix2 * 100);

tic;
result_norm = matrix1 * matrix2;
serial(count) = toc;

clc;
disp('Matrix 1');
disp(matrix1);
disp('Matrix 2');
disp(matrix2);

disp('Normal Result');
disp(result_norm);
fprintf('Normal Time:%0.04f s\n',serial(count));

end
```

C. Limits to serial programming

Both physical and practical reasons pose significant constraints to simply building ever faster serial computers [3]:

- Transmission speeds - the speed of a serial computer is directly dependent upon how fast data can move through hardware. Absolute limits are the speed of light (30 cm/nanosecond) and the transmission limit of copper wire (9 cm/nanosecond). Increasing speeds necessitate increasing proximity of processing elements.
- Limits to miniaturization - processor technology is allowing an increasing number of transistors to be placed on a chip. However, even with molecular or atomic-level components, a limit will be reached on how small components can be. Economic limitations - it is increasingly expensive to make a single processor faster. Using a larger number of moderately fast commodity processors to achieve the same (or better) performance is less expensive.
- Current computer architectures are increasingly relying upon hardware level parallelism to improve performance:
  - Multiple execution units
  - Pipelined instructions
  - Multi-core

D. Parallel Programming

Parallel programming involves the concurrent computation or simultaneous execution of processes or threads at the same time. Parallel programming, we have multiple processes execute at the same time.

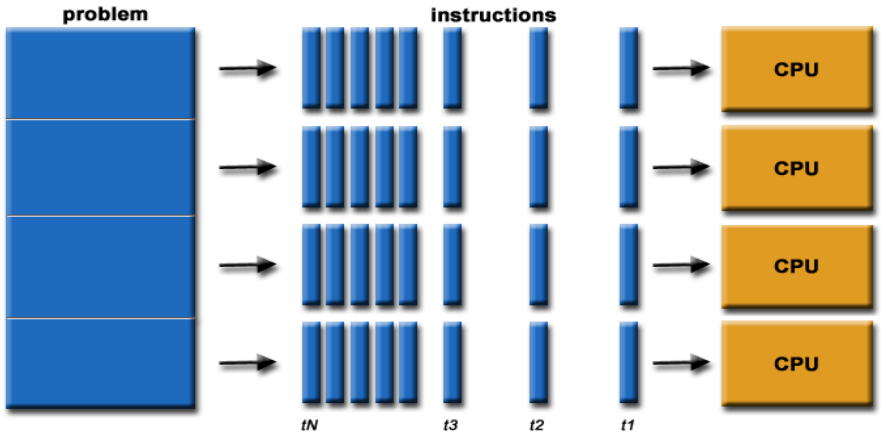

Fig. 3 Parallel programming (Source: https://computing.llnl.gov/tutorials/parallel_comp/ image

In [3] author discusses the advantages of Parallel computing they are as follows

- **Save time and/or money:**

In theory, throwing more resources at a task will shorten its time to completion, with potential cost savings. Parallel computers can be built from cheap, commodity components.

- **Solve larger problems**:

II. Many problems are so large and/or complex that it is impractical or impossible to solve them on a single computer, especially given limited computer memory.

III.

- **Provide concurrency:**

A single compute resource can only do one thing at a time. Multiple computing resources can be doing many things simultaneously

- **Parallel programming involves:**
  - Decomposing an algorithm or data into parts
  - Distributing the parts as tasks which are worked on by multiple processors simultaneously
  - Coordinating work and communications of those processors or cores.

E. Parallel coding:

```
tic;
spmd
result_norm = matrix1 * matrix2;
end;
parallel(count) = toc;

tic;
spmd
result = applyFoxMultiplier(matrix1,matrix2);
end;
fox(count) = toc;

function out_matrix = applyFoxMultiplier(matrix1,matrix2)
rows = size(matrix1,1);
cols = size(matrix1,2);
out_terms = zeros([rows cols]);

if(size(matrix1,1) == size(matrix2,1) && size(matrix1,2) == size(matrix2,2))

   for stage=0:rows-1
      %Stage 0
      for i=0:rows-1
         for j=1:cols
            %Find the mod val
            mod_val = mod(i+stage,rows);
            mod_val = mod_val + 1;

            out_terms(i+1,j) = out_terms(i+1,j) + (matrix1(i+1, mod_val) * matrix2(mod_val,j));
         end
      end
   end
end

out_matrix = out_terms;
```


III. RESULT AND DISCUSSION

Theoretically number of steps required for Normal algorithm is more than that required for the Fox algorithm. Fox algorithm takes n+1 steps whereas the Normal algorithm takes n*n steps to do the same calculations.

TABLE I

NUMBER OF STEPS REQUIRED FOR MATRIX MULTIPLICATION USING NORMAL ALGORITHM AND FOX ALGORITHM:

Matrix size (n)	Normal algorithm (n*n)	Fox algorithm (n+1)
10	100	11
50	2500	51
100	10000	101
150	22500	151

A practical analysis for the performance of serial and parallel algorithm is carried out using Matlab version 7.0 software. Serial algorithm requires less time to execute the code than parallel algorithm if executed sequentially.

TABLE III

SERIAL COMPUTING EXECUTION TIME FOR INTEL I3 DUAL CORE PROCESSOR:

Size of matrix	Number of Workers	Normal algorithm execution time(s)	Fox algorithm execution time(s)
100×100	-	0.0414	0.1544

For Intel i3 Dual Core processor, the parallel computing execution time for matrix multiplication using Normal and Fox algorithm using 2, 4, 6 and 8 workers is carried out and it is found that Fox algorithm is having the least time delay. Again it has shown that if we converts serial algorithm into parallel it takes more time to execute the code where as if we take parallel algorithm such as Fox algorithm for matrix multiplication it takes less time as compare to Serial algorithm.

TABLE IIIII

PARALLEL COMPUTING EXECUTION TIME FOR INTEL I3 DUAL CORE PROCESSOR:

Size of matrix	Number of Workers	Normal algorithm execution time(s)	Fox algorithm execution time(s)
100×100	2	1.8572	0.5246
	4	2.4456	0.4838
	6	2.7044	0.9814
	8	1.0062	0.8295

IV. CONCLUSIONS

Serial processing was the best way of computing data sets until hardware and software technologies finally caught up and made true parallel processing a reality. With the advent of multi-core processors and the universal expectation that the number of cores on a chip will increase exponentially over the next years (in line with the original prediction of Moore's law [11]), with little or no increase in sequential performance in sight, parallelism, be it implicit or explicit, will become fundamental in all application areas where performance matters.

ACKNOWLEDGMENT

The authors wish to thanks Dr. A. W. Kolhatkar, and Dr. S. M. Gulhane for their comments and time to time valuable suggestion.

REFERENCES

[1] D. Vijaya Krishna and Dr. P. Sammulal, **"**Advances in Parallel Computing from the Past to the Future", *International Journal of Advance Research in Computer Science and Management Studies*, Volume 1, Issue 4, pp 16-23, September 2013.

[2] Nimrod Megiddo, "Applying Parallel Computation Algorithms in the Design of Serial Algorithms", *Journal of the Association for Computing Machinery*, Vol. 30, No. 4, pp. 852-865, October 1983.

[3] David A. Bader, Bernard M. E. Moret, and Peter Sanders ,"Algorithm Engineering for Parallel Computation" *Springer-Verlag Berlin Heidelberg*, pp. 1–23, 2002.

[4] Prabhudev S Irabashetti, "Parallel processing in processor organization", *International Journal of Advanced Research in Computer and Communication Engineering*, Vol. 3, Issue 1,pp. 5150-5153, January 2014.

[5] Parallel Computing Toolbox™ 4 User's Guide, Copyright 2004–2009 by The Math Works, Inc.

[6] P. Durgaprasad," Parallel Computing: High Performance", *International Journal of Emerging Technology and Advanced Engineering* , Volume 1, Issue 2, pp.97-101, December 2011.
[7] Ron Choy , Alan Edelman," Parallel MATLAB: Doing it Right",pp.1-27, November 15, 2003.
[8] Daniel S. Priece," Parallel Computing", pp.55-65.
[9] Speeding up MATLAB Applications, The Math Works, Inc.,pp.1-15, 2011.
[10] Nadya Bliss, "Addressing the Multicore Trend with Automatic Parallelization", Lincoln Laboratory Journal, Volume 17, Number 1, pp.187-197, 2007.
[11] Jesper Larsson Träff," What the parallel-processing community has (failed) to offer the multi/many-core generation", Elsevier Inc., pp. 807-812, 2009.

