A new Sort Algorithm for Multi Core parallel Computers

Prof. Dr. Abdul Monem S. Rahma

110003@uotechnology.edu.iq/

Assist. Prof. Dr.Maisa'a Abid Ali K

110044@uotechnology.edu.iq/

Computer Science Department/University of Technology

Abstract

Sorting is one of the basic problems of tremendous data for computers. In the past computer processes were executed on a single processor and this took a long time. To solve this problems computers used multi cores, to decrease the amount of time taken for solving these operations. The advent of the internet allowed people to pool their processing power; parallel processing. This paper offers such sort algorithms for parallel processing. The proposed algorithm in this paper sorts random numbers and save them in a text file. It uses three parameters: number cutting, multi core, and times. The file is divided in a number of matrix. And uses two main operations: the first operation merges a column with the adjacent column, and the second operation sorts ascending and shared memory in cores P1, P2, P3, and P4, and computes time/millisecond for each process. This algorithm is executed in a computer having multiply cores; CPU 0, CPU 1, CPU 2, and CPU 3.

The outcome of this algorithm is fast, efficient, and produces optimal results; when the number of processors increase, execution time is decrease.

Key word: Sort, Merge, Multi core, Random numbers, Parallel processing, Text file.

المستخلص

ان احد المشاكل الاساسية في عملية فرز البيانات الكبيرة جدا في الحاسبة القديمة ذات معالج واحد قد تاخذ وقت طويل في عملية تنفيذ البيانات. ولحل هذه المشاكل تم استخدام حاسبات متعددة النواة، لتقليل كمية الوقت المستغرق لحل هذه العمليات. وان وصول انتشار الانترنيت واكثر الناس تعمل باتجاه المعالجات متوازية. قدم هذا البحث نوع من خوارزمية الفرز في المعالجات المتوازية ، لذلك اقترح البحث خوارزمية فرز الارقام العشوائية وخزنها في ملف .txt وتم استخدام ثلاث براميترات هي تقطيع الارقام ، الوقت، تعدد النواة. وممكن ان تقسم هذه الارقام العشوائية بواسطة عددمن الاعمدة داخل مصفوفة.

واستخدام عمليتين رئيسيتين وهي : العملية الاولى دمج كل عمودين متجاورة، والعملية الثانية عملية فرز وترتيب تصاعدي واستخدام تقسيم الذاكرة في تعدد النواة في البروسيسرات الاول، والثاني، والثالث، والرابع، وتم حساب الوقت بالملي \ ثانية لكل بروسيسر، هذه الخوارزمية تنفذ داخل حاسبات تشمل وحدة معالجة مركزية 0، وحدة معالجة مركزية1، وحدة معالجة مركزية2، و وحدة معالجة مركزية3.

النتائج التي تم الحصول من الخوارزمية هي جيدة وسريعة وكفوءة وكانت النتائج مثالية ، عندما تزداد عدد المعالجات، يقل وقت التنفيذ.

الكلمات المفتاحية: الفرز، الدمج، تعدد النواة، الارقام العشوائية، المعالجة المتوازية، ملف نصي

1. Introduction

Due to the size of the large data indexing and the need to arrange the use of available computers and memory. Computer systems development for parallel processors includes a number of processing units connected via some network the software wants to batch jobs together. The processing units can be contact and interact with each other uses either sharing memory or messages . The intercourse net for sharing memory systems have been categorize merge and sorting [1], [2], [3].

The diffusion of multiprocessor structure will have a diffusion effect on how we develop software to merge and sort tremendous data. The technology offers a means to advance time rapidity, so software would effectively "rapidities" by itself all time. Therefore; advances in technology will have means increased parallel processing and not increased time quick, and exploitation such as parallel operation is one of the science challenges of modern at many areas of computer science [2], [3]. Computer architecture in parallel processing depends on multiple–instruction multiple–data streams (MIMD) [3].

Sorting is used in many scientific and engineering applications. It is one of the main calculations algorithms . Several sequential sorts take O(NlogN) times of sort N as keys. The many parallelism sorting algorithms as "bitonic sort, sample sort, column sort, and partitioned radix sort" [3], [4].

This sorting an unordered series, series are first merged at big bitonic series, starting with couple of close numbers. Couple of close numbers are formative at growing series and lessening series using a comparison,and interchange process, couple of which form a bitonic series of twice the volume of all of the original series [5].

Rapid sorting is composed using splits of the stack of data , in which every portion is processed to regulate items in relevance to a selected spliter. Numerous versions of this manner offers likely improvements for quick up the operation, and prevents failure or stop [6].

2. Paralell Processing sorting Methods

There are many storing methods in parallelism operation in computers.

1- Bitonic sort

The bitonic is chain and trend of contrast is limited advance and independent of income chain. Bitonic sort has depended on a bitonic chain. It drops to the collection of sorting nets [8], [9].

A bitonic has a chain $(a_1, a_2, \ldots\ldots a_N)$ is called bitonic if either an indices i such as that $(a_1 < \ldots\ldots < a_i$ and $a_i > \ldots\ldots > a_n)$, or there is a cyclic shift of index such that (1) holds [7]. As shown in Figure (1), the idea of bitonic. The characteristic of the transformed sequence, the a_l and a_r are two bitonic sequences; and all elements of the first sequence are smaller than those of the second sequence [8], [9].

- A chain contains two items is bitonic.
- All bitonic chain has sort uses the algorithms taking before in increased or decreased order.
- The connecting two neighboring chain where the first has increases while the second has decreases provides a bitonic chain [8], [9], [10].

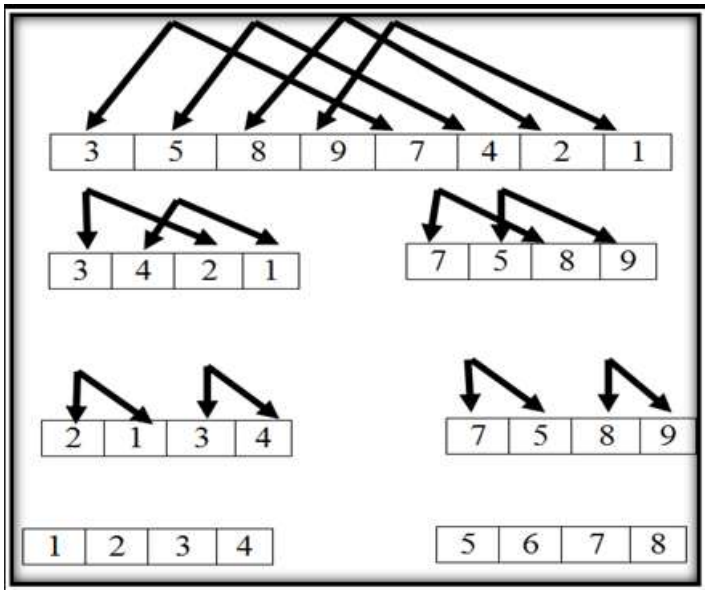


Figure (1): The bitonic sorted.

3. Previous Works in Multi Core

In 2015, Chengfei GU, et.al., the work in operation architecture, the visualize algorithm has been modularized, and all module is efficient understand by the proposed operation architecture. In all module, information processing of varying cores is run in parallel, also information is sent and information processing of every core can execute synchronously, also that the processing timing for SAR visualize is safely minimize. the outcome is an efficiently parallel operation expert for SAR real-time visualize is proposed. "With only an eight-core DSP is perfectly expert for the higher-resolution real-time processing of FMCW SAR" [11].

In 2017, J. Labeit, et.al., the work and deepness to first parallel 2D wavelet tree algorithms matching those of the better present parallel algorithms while demanding asymptotically least memory and the second algorithm realizes the same asymptotic limits for little alphabet volume. These tests offer, both speed and

memory efficiency than present parallel algorithms. It also presents test estimate of the parallel structure of rank and choose structure, which are used in 2D wavelet trees. The outcome of these tests show that memory efficiency, realize good parallel scalability, and out perform present applicatiions for the same problem, and see that a concentrate on memory efficiency can also get better the execution of algorithms [12].

In 2017, Z. Marszałek, the newly version can interest both from quicker calculate on the multi core architectures, and smart programming method which is using efficiently procedures ready in the final programming studios. Repeatedly used algorithms to sort matrix of information in No SQL databases is combined sort, where as No SQL and understood any database without model SQL programming translator. The writer describes how to use the parallelization of the sorting operations for the adjust manner of sorting by combining for largest data sets. The outcomes are offered in theoretic hypothesis and sure in functional benchmark experiment. The manner is contrasted to other sorting manners such as fast sort,stack sort, and combine sort to see possibility qualification. The parallelization mannerof the sorting methods is quicker and helpful for multi-core systems [10].

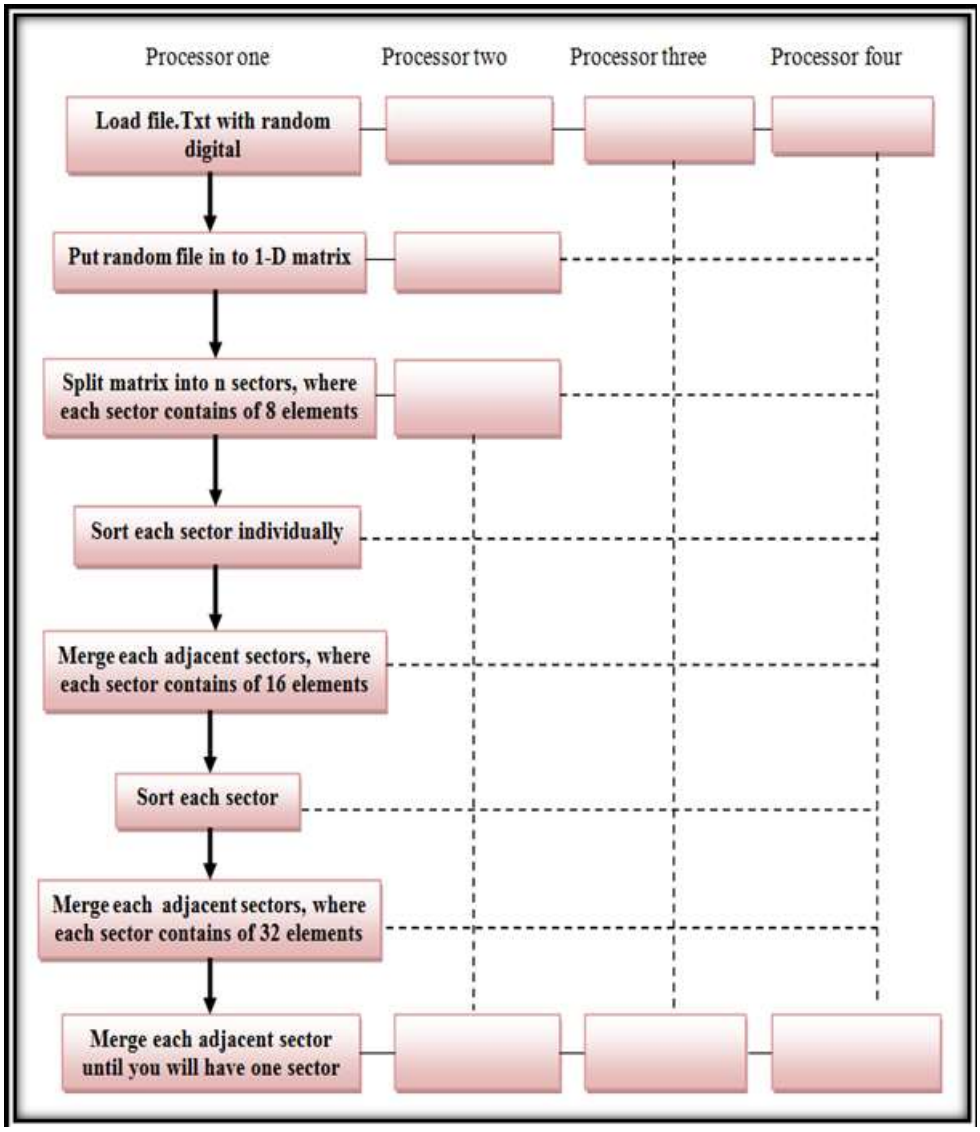
4. Proposed system

The proposed system offers a new algrithm for sorting in parallel processing computer based on multicore. The implementation system in computer multi core consists of four central processing units CPU1, CPU2, CPU3, CPU4. The system consists of four phases in each phase implementation sorting

algorithm depend on concept pipeline in multicore, but different in executed times between four processors and each algorithm usese three prameters: number of sector, number of processor, and times.

This system uses selected sequence of randoms digits keeps in file ".Txt", this file includes the largest number of digits randoms, and converts this random digits to matrix one dimension, and split this matrix to number of sector one dimension 8,16, 32, 64, 128, and rearrange each sector by using one, two, three, and four processors, uses multicore processor in computer. This system can be used in parallel processing multicore to increase speed of computer during communication network, and calculated the time in millsecond to measure capability of each processor in computer by depending on pipeline algorithm. In system insert file ".txt," multi input multi output. The basic algorithm is two stages, sort and merge, sorting N numbers performs log N merge stages. It can be used visual basic.net 2015 for implementation sysetm in four threading.

Figure (2) indicates general flowchart of proposed system in each processor, process1, process2, process3, process4.



Figure(2): General flowchart of proposed system in multi core system.

The two operations in multi core system, sorting and merging work for aspect of pipeline in work for one processor, two processors, three processors, and four processors. Figure (3) explains general operation of pipeline in each cycle for prcessors in system.

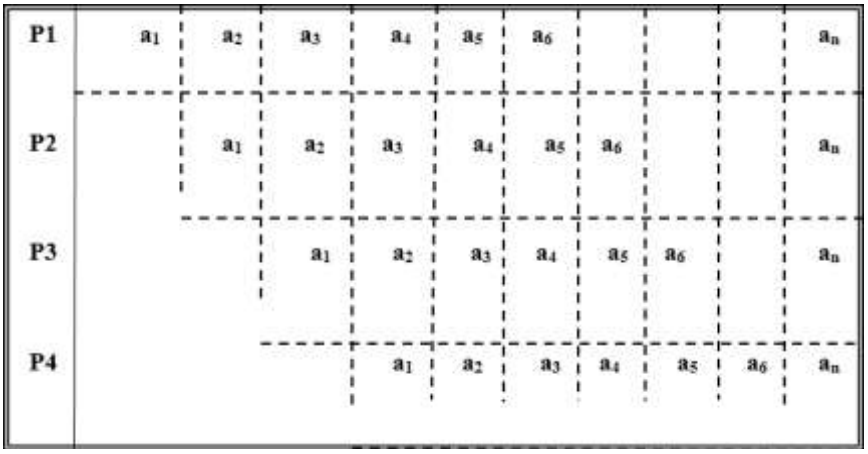


Figure (3): General operation of ppipeline in multi core system.

1- Sorting Algorithm in Multi Core of Process One:

This algorithm compute the total time of sort and merge in one processor for file random digits, by using share memory in each processor, below explain detail of Algorithm.

Process: Sort and Merge Algorithm in multi core of one processor
Input: Text file include random numbers.
Output: Sorted file.
Initial
A = Load file random digits
B = Split file in one processor p1 for 8, 16, 32, 64, 128
C = Sorted file
D = Merge file
E = Calculated total time
F = Final sorted file
Step 1:Load file random digits into A.
Step 2: Split file to share memory is divided for 8, 16, 32, 64, 128 parts into B.
Step 3: Sorting each part [from 8, 16, 32, 64, 128] into C, and calculated time sort.
Step 4: Merge each part [from 8, 16, 32, 64, 128] into D, and calculated time merge.
Step 5: Calculated total time for one processor into E.
Step 6: Put (the sorted result into F).
Step 7: End

2- Sorting Algorithm in Multi Core of Process Two:

This algorithm compute the total time of sort and merge in two processors for file random digits, by using share memory in each processor.

Process: Sort and Merge Algorithm in multi core of two processors
Input: Text file include random numbers.
Output: Sorted file.
Initial
A = Load file random digits.
B = Split file in two processors P1 and P2
C = Split file in each processor 8, 16, 32, 64, 128
D = Sorting file
E = Merge file
F = Calculated total time
G = Final sorted file.
Step 1: Load file random digits into A.
Step 2: Split file in P1 and P2 into B.
Step 3: Split file to share memory in P1 and P2 is divided for each processor for 8, 16, 32, 64, 128 parts into C.
Step 4: Sorting each part[from P1 and P2 for 8, 16, 32, 64, 128] into D, and calculated time sort.
Step 5: Merge each part [from P1 and P2 for 8, 16, 32, 64, 128] into E, and calculated time merge.
Step 6: Calculated total time for two processors into F.
Step 7: Put (the sorted result into G).
Step 8: End

3- Sorting Algorithm in Multi Core of Process Three:

This algorithm compute the total time of sort and merge in three processors for file random digits, by using share memory in each processor, below explain detail of algorithm.

Process: Sort and Merge Algorithm in multi core of three processors
Input: Text file include random numbers.
Output: Sorted file.
Initial
A = Load File.txt Random digits.
B = Split file in three processors P1, P2, and P3
C = Split file each processor 8, 16, 32, 64, 128
D = Sorting file
E = Merge file
F = Calculated total time
G = Final sorted file
Step 1: Load file random digits into A.
Step 2: Split file in P1, P2 and P3 into B
Step 3: Split file to share memory in P1, P2, and P3 is divided for each processor for 8, 16, 32, 64, 128 parts in C.
Step 4: Sorting each part [from P1, P2, and P2 for 8, 16, 32, 64, 128] into D, and calculated time sort.
Step 5: Merge each part [from P1, P2, and P3 for 8, 16, 32, 64, 128] into E, and calculated time merge.
Step 6: Calculated total time for three processors into F.
Step 7: Put (the sorted result into G).
Step 8: End

4- Sorting Algorithm in Multi Core of Process Four:

This algorithm compute the total time of sort and merge in four processors for file random digits, by using share memory in each processor, below explain detail of algorithm.

Process: Sort and Merge Algorithm in multi core of four processors
Input: Text file include random numbers.
Output: Sorted file.
Initial
A = Load File random digits.
B = Split file in four processors P1, P2, P3, and P4
C = Split file in each processor 8, 16, 32, 64, 128
D = Sorting file
E = Merge file
F = Calculated total time
G = Final sorted file
Step 1: Load file random digits into A.
Step 2: Split file in P1, P2, P3, and P4 into B
Step 3: Split file to share memory in P1, P2, P3, and P4 is divided for each processor for 8, 16, 32, 64, 128 parts into C.
Step 4: Sorting each part [from P1, P2, P3, and P4 for 8, 16, 32, 64, 128] into D, and Calculated time sort.
Step 5: Merge each part [from P1, P2, P3, and P4 for 8, 16, 32, 64, 128] into E, and Calculated time merge.
Step 6: Calculated total time for four processors in F.
Step 7: Put (the sorted result into G).
Step 8: End

5. Test of Result

This section calculates test of result for times of sort, merge and total time implementation in multi core one processor, two

processor, three processor, and four processor in system. Table (1) is indicated the measure of time in multi core system.

Table (1): Indicates the measure of time in multi core system.

No.of Test	P1 milisecond	P2 milisecond	P3 milisecond	P4 milisecond
1000	Sort=5805129 Merge=3605129 Total=9410258	Sort=4349021 Merge=4305359 Total=8654380	Sort=3485348 Merge=3485348 Total=6970696	Sort=1971146 Merge=2127417 Total=4098563
10000	Sort=4985468 Merge=4965478 Total=9950946	Sort=4517514 Merge=4517514 Total=9035028	Sort=3668481 Merge4224759 Total=7893240	Sort=23643791 Merge=3643791 Total=7287582
100000	Sort=7895833 Merge=7867533 Total=15763366	Sort= 5295241 Merge=5451777 Total=10747018	Sort=4329935 Merge=4484313 Total=8814248	Sort=4129536 Merge=4285799 Total=8415335
1000000	Sort=9926077 Merge=9923077 Total=19849154	Sort=8184205 Merge=10059399 Total=18243604	Sort= 5576123 Merge=7762275 Total=13338398	Sort=4833643 Merge=7490551 Total=2324194

6. Analysis Test of Result in multi core

This section is saying for analysis of reselt of sorting algrithm for multi core system for time sort, time merge, and total time sorting, as shown in Figue (4) indicates analysis multi core system in P1, P2, P3, P4 in select 1000 random digits. Figure (5) indicates analysis multi core system in P1, P2, P3, P4 in select 10000 random digits, Figure (6) indicates analysis multi core system in P1, P2, P3, P4 in select 100000 random digits, and Figure (7) indicates analysis multi core system in P1, P2, P3, P4 in select 1000000 random digits. These figures explain difference curves between four processors each state in selected random digits. The size of the file.txt is increasing, because total time is increasing in each processors in multi core system. Figure 4,

Figure 5, Figure 6, Figure 7, is explaining curve in sort, merge, and tatal times is increasing in millisecond.

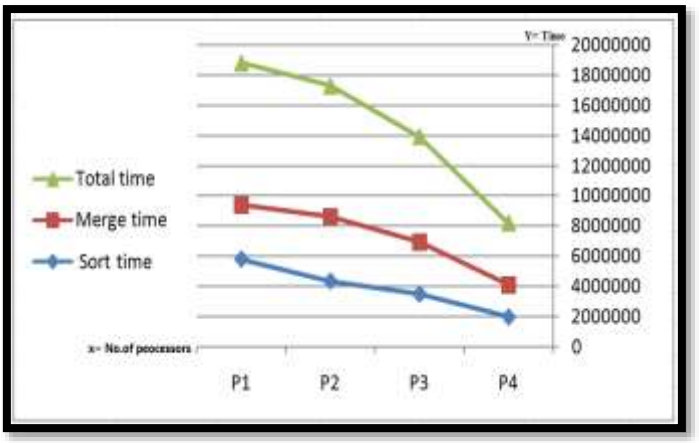


Figure (4): Indicates 1000 random digites in sorting time.

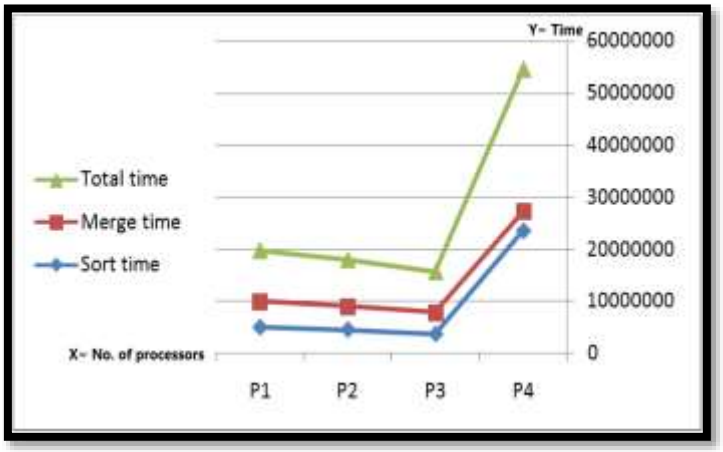


Figure (5): Indicates 10000 random digites in sorting time.

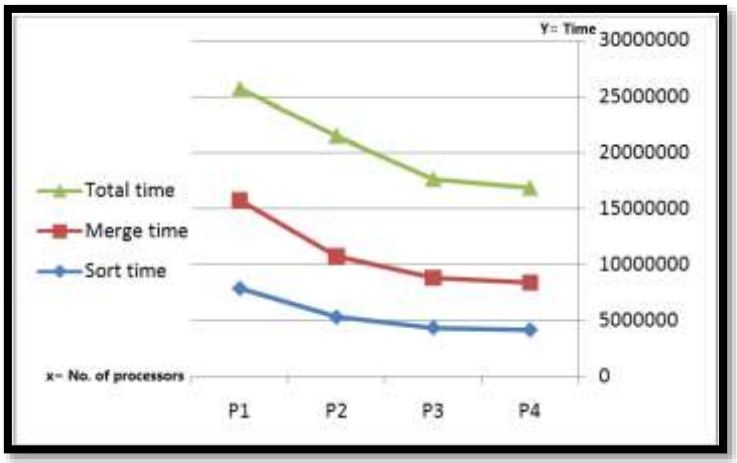


Figure (6): Indicates 100000 random digites in sorting time.

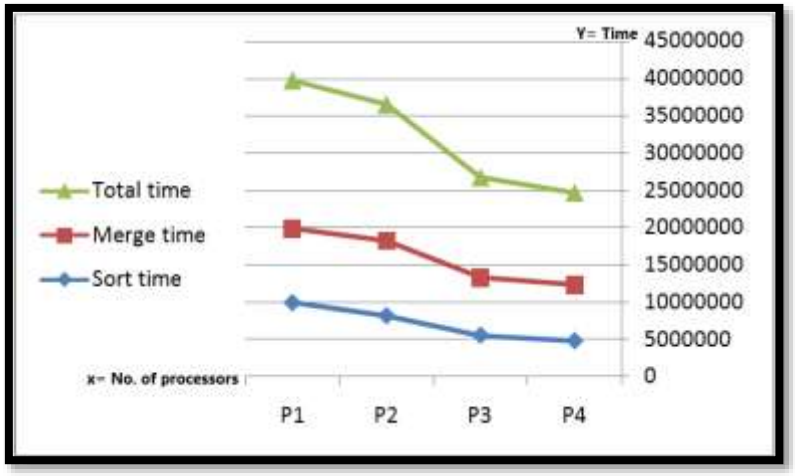


Figure (7): Indicates 1000000 random digites in sorting time.

7. Conclusion

In performance evaluation this system, it has shown the sorting methods in parallel processing the result of uses evolved networks in the present era. The large evolved in sorting algorithms, this paper described sorting algorithm in multicore parallel processing, to sort number of random digits depended on calculating sort time and merge time to obtain total time in each processes. In this system noticed time begin reducds with increased number of multi core in computer, this indicates interest in increased communication networks with multi core. the proposed algorithm in system is good implementation and very fast in sorting with uses multi core in one computer, the outcome obtained when uses one processor the average times in millisecond of sort and merge from 09410258 to 19849154, the two processors the average times of sort and merge from 8654380 to 18243604, the three processors the average times of sort and merge from 6970696 to 13338398, and the four processors the average times of sort and merge from 4098563 to 12324194. When uses sorting in 1000 digit the time

implementation in one processor more than implementation in two or three or four processors it can be least time, also as in 10000 or 100000 or 1000000.

References

1. H. El-Rewini, M. Abd-El-Barr, 2005, Advanced Computer Architecture And
2. Parallel Processing, *Published by John Wiley & Sons*, Inc., Hoboken, New Jersey.
3. Published simultaneously in Canada, Book, PP 287.
4. M. herlihy, and N. Shavit, 2008, The Art of Multiprocessor Programming, Morgan
5. Kaufmann Publishers Elsevier Inc., Book, PP 529.
6. Y. Cheol Kim, M. Jeon, D. Kim, and A. Sohn, 2001, Communication-Efficient
7. Bitonic Sort on a Distributed Memory Parallel Computer, Conference, Source:
8. IEEE Xplore, PP 165-170. DOI: 10.1109/ICPADS.2001.934815·
9. S, Rajasekaran , S. Sen, 2005, A Generalization Of The 0–1 Principle For Sorting,
10. Information Processing Letters 94, PP43-47. Available at: www.elsevier.com
11. /locate/ipl
12. S. Rathi, 2016 Elsevier, Optimizing Sorting Algorithms using Ubiquitous multi-
13. core massively parallel GPGPU processors, *7th International Conference on*
14. *Communication, Computing and Virtualization, PP 231-237.*
15. DOI: 10.1016/j.procs.2016.03.030
16. Z. Marszałek, 2017, Parallelization of Modified Merge Sort Algorithm, *Journal of*

17. *Symmetry* 176, , Impact Factor 1.457, PP 1-18. DOI :10.3390/sym9090176.
18. Q. Wanga, J. JaJaa, A. Varshney, 2007 Elsevier Inc., An Efficient And Scalable
19. Parallel Algorithm For Out-Of- Core Isosurface Extraction And Rendering, *Journal*
20. *of processing And Distribuited Computing*, PP 592- 603.
21. DOI: 10.1016/j.jpdc.2006.12.007.
22. T. Rauber, G. Runger, 2010 Springer, Parallel Programming For Multicore And
23. Clustering Systems, *Springer Heidelberg Dordrecht London New York*, Book, PP
24. 463. DOI: 10.1007/978-3-642-04818-0.
25. D. Božidar, and T. Dobravec,2015, Comparison of parallel sorting algorithm, *aculty*
26. *of Computer and Information Science*, University of Ljubljana, Slovenia, Technical Report.
27. D. Taniar, and J. Wenny Rahayu, 2002 Elsevier, Parallel Database Sorting,
    a. *Information Sciences An International Journal,* PP 171-219.
28. Chengfei GU, Wenge CHANG, Xiangyang LI, and Zhaohe LIU, Multi-Core DSP
29. Based Parallel Architecture for FMCW SAR Real-Time Imaging, *Radio*
30. *Engineering, Vol. 24, No. 4, December 2015.* DOI: 10.13164/re.2015.1084
31. J. Labeit, J. Shun, and G. E.Blelloch, Parallel Lightweight Wavelet Tree, Suffix
32. Array and FM-Index Construction, *Journal of Discrete Algorithms,* Published by
33. Elsevier B.V. 2017. Available at:http://dx.doi.org/10.1016/j.jda.2017.04.001