IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2320-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 10, Issue 3, March 2021

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.512

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

Comparative Study of Different Laws for Measuring Speed up Performance in Parallel Computing

Varsha Ganesh [1]

Assistant Professor, Department of Computer Science, CHRIST College (Autonomous), Irinjalakuda, Thrissur, India[1]

ABSTRACT: Parallel computing used to be largely confined to High Performance Computing (HPC), system architectures designed to handle high speed and density calculations. Notable applications for parallel processing (also known as parallel computing) include computational astrophysics, geo-processing (or seismic surveying), climate modelling, agriculture estimates, financial risk management, video colour correction, computational fluid dynamics, medical imaging and drug discovery. Parallel computing works in the following way: each computing problem is broken down into separate parts and each part is processed as a series of instructions by processing cores. The processor contains any number of these parallel cores; in mass parallelism the cores could be in the hundreds. The objective is improved efficiency by getting more instructions executed per unit time. Both hardware and software engineers have adapted parallelism to their respective fields. The processing cores are units on the same chip and part of the same processor architecture; this can also include multiple chips working at the same time like L1 and L2 caches and graphic processing units (GPU) as well as processing cores. Compared to serial computing, parallel computing is much better suited for modelling, simulating and understanding complex, real world phenomena. There are certain parameters are required which can measure the speed up performance of these systems. For this various laws have been developed such as **Amdahl's Law, Gustafson's Law, Sun and Ni's Law etc** .This paper is intended to provide a comparative study of these laws for measuring speedup performance in parallel computing.

KEYWORDS: Graphic Processing Unit, Parallel Computing, Serial Computing

I. INTRODUCTION

Parallel computing used to be largely confined to High Performance Computing (HPC), system architectures designed to handle high speed and density calculations. The experiments with and implementations of the use of parallelism started long back in the 1950s by the IBM. The IBM STRETCH computers also known as IBM 7030 were built in 1959. In the design of these computers, a number of new concepts like overlapping I/O with processing and instruction look ahead were introduced. A serious approach towards designing parallel computers was started with the development of ILLIAC IV in 1964 at the University of Illionis. It had a single control unit but multiple processing elements. On this machine, at one time, a single operation is executed on different data items by different processing elements. The concept of pipelining was introduced in computer CDC 7600 in 1969. It used pipelined arithmetic unit. In the years 1970 to 1985, the research in this area was focused on the development of vector super computer. In 1976, the CRAY1 was developed by Seymour Cray. Cray 1 was a pioneering effort in the development of vector registers. It accessed main memory only for load and store operations. Apart from Cray, the giant company manufacturing parallel computers, Control Data Corporation (CDC) of USA, produced supercomputers; the CDC 7600.A parallel system consists of an algorithm and the parallel architecture that the algorithm is implemented. These algorithms may have different performance on different parallel architecture .That's means an algorithm may perform differently on a linear array of processors and on a hypercube of processors. Achieving the highest possible performance has always been one of the main goals of parallel computing. Sequential algorithms are mainly analyzed on the basis of computing time i.e., time complexity and this is directly related to the data input size of the problem. For example, for the problem of sorting n numbers using bubble sort, the time complexity is of $O(n^2)$. However, the performance analysis of any parallel algorithm is dependent upon three major factors viz. time complexity, total number of processors required and total cost. The complexity is normally related with input data size (n). Various kinds of metrics involved for analysing the performance of parallel algorithms for parallel computers are **Running Time, Speed Up and Efficiency.** By using various laws such as Amdahl's law, Gustafson's law and Sun & Ni's law ,we can analyse the speed-up performance of parallel processing.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

II. RELATED WORK

Achieving the highest possible performance has always been one of the main goals of parallel computing [2]. Various kinds of metrics involved for analysing the performance of parallel algorithms for parallel computers are **Running Time, Speed Up and Efficiency .**This paper highlight speed-up parameters involved for analysing the performance of parallel computers.

METRICS FOR PERFORMANCE EVALUATION

Running Time

The running time is the amount of time consumed in execution of an algorithm for a given input on the N-processor based parallel computer [5][6]. The running time is denoted by T(n) where n signifies the number of processors employed. If the value of n is equal to 1, then the case is similar to a sequential computer. The relation between Execution time vs. Number of processors is shown in Figure 1

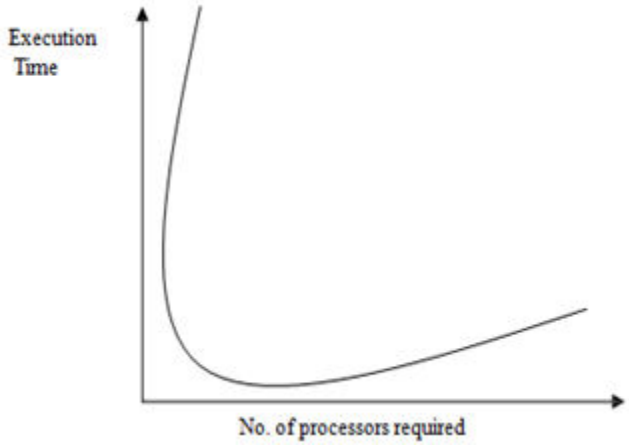

Fig : 1 Execution time vs. No of process

It can be easily seen from the graph that as the number of processors increases, initially the execution time reduces but after a certain optimum level the execution time increases as number of processors increases. This discrepancy is because of the overheads involved in increasing the number of processes.

Speedup

It is a measure of performance. Speed up is the ratio of the time required to execute a given program using a specific algorithm on a machine with single processor (i.e. T (1) (where n=1)) to the time required to execute the same program using a specific algorithm on a machine with multiple processors (i.e. T (n)). Basically the speed up factor helps us in knowing the relative gain achieved in shifting from a sequential machine to a parallel computer. It may be noted that the term T(1) signifies the amount of time taken to execute a program using the best sequential algorithm i.e., the algorithm with least time complexity.

T (1) is the execution time with one processing unit
T (n) is the execution time with n processing units

$$S(n) = \frac{T(1)}{T(n)}$$

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

Let us take examples and illustrate the practical use of speedup in Table: 1.

Number of CPU	1	2	4	8	16
T(n)	1000	520	280	160	100
S(n)	1	1.9230	3.5714	6.25	10

Table 1: Analysis of speed up performance

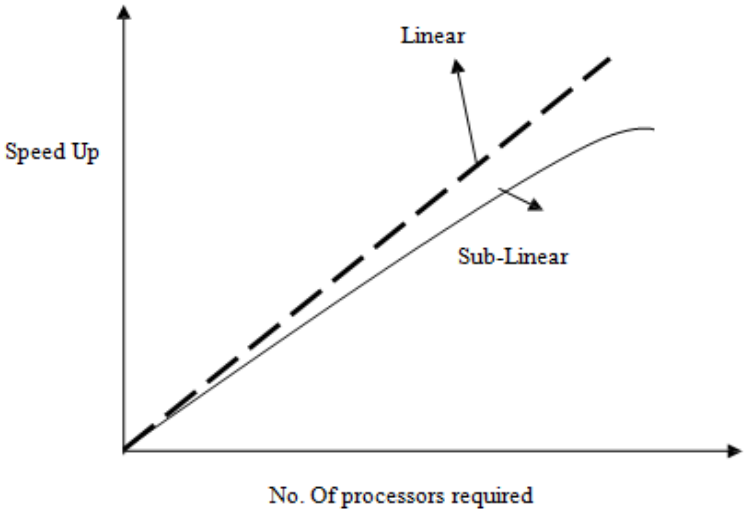

Fig: 2 Speed Up Vs Number Of Processors

Efficiency

The efficiency of a program on a parallel computer with k processors can be defined as the ratio of the relative speed up achieved while shifting the load from single processor machine to k processor machine where the multiple processors are being used for achieving the result in a parallel computer [7]. This is denoted by E (k).
E_k is defined as follows:

$$E(k) = s(k)/k$$

The value of E(k) is directly proportional to S(k) and inversely proportional to number of processors employed for performing the computation. The relation between E(k) vs. Number of processors is shown in Figure 3.

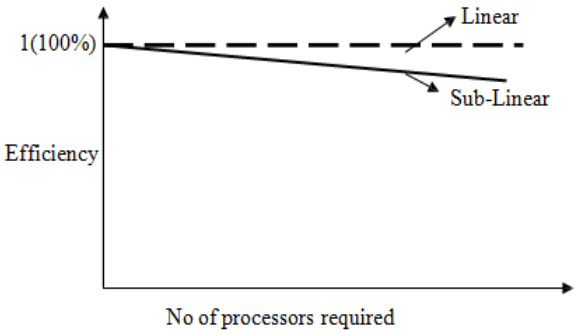

Fig 3: Efficiency vs. No of process

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

III. METHODOLOGY

Amdahl's Law

The speed up factor helps us in knowing the relative gain achieved in shifting the execution of a task from sequential computer to parallel computer and the performance does not increase linearly with the increase in number of processors. Due to the above reason of saturation in 1967 Amdahl's law was derived.
Amdahl's law [1][11] states that a program contains two types of operations i.e. complete sequential operations which must be done sequentially and complete parallel operations which can be executed on multiple processors.
Let us consider a problem say P which has to be solved using a parallel computer. According to Amdahl's law, there are mainly two types of operations. Therefore, the problem will have some sequential operations and some parallel operations.
T (1) is the amount of time to execute a problem using a sequential machine and sequential algorithm. The time to compute the sequential operation is a fraction α (alpha) ($\alpha \leq 1$) of the total execution time i.e. T (1) and the time to compute the parallel operations is (1- α).
Therefore, S (N) can be calculated as under:

$$S(N) = \frac{T(1)}{T(N)}$$

$$S(N) = \frac{T(1)}{X^* T(1) + (1-\alpha) * T(1)/T(N)}$$

Dividing by T(1)

$$S(N) = \frac{1}{\alpha + \frac{1-\alpha}{N}}$$

The value of α is between 0 and 1. Now, let us put some values of α and compute the speed up factor for increasing values of number of processors. We find that the S(N) keeps on decreasing with increase in the value of α in Table 2.

N	α = .5	α = .9	α = .95	α = .99
10	1.818182	1.098901	1.04712	1.009082
100	1.980198	1.109878	1.052078	1.009999
1000	1.998002	1.110988	1.052576	1.010091

Table 2: Analysis of Amdahl's law

There is one major shortcoming identified in Amdahl's law. According to Amdahl's law, the workload or the problem size is always fixed and the number of sequential operations mainly remains same. That is, it assumes that the distribution of number of sequential operations and parallel operations always remains same. This situation is shown in and the ratio of sequential and parallel operations is independent of problem size.

Ls = Sequential Instruction Load
Lp = Parallel Instruction Load

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

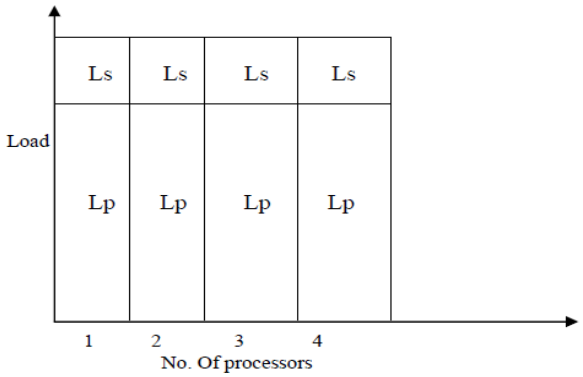

Fig 4: Fixed load for Amdahl's Law

However, practically the number of parallel operations increases according to the size of problem. As the load is assumed to be fixed according to Amdahl's law, the execution time will keep on decreasing when number of processors is increased.

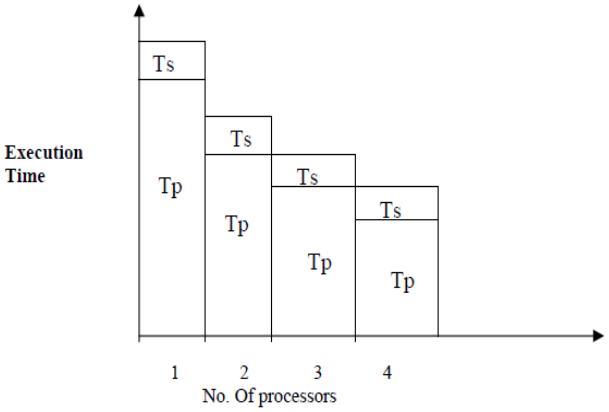

Fig 5: Execution time decreases for Amdahl's law

Gustafson's law

John L. Gustafson (born January 19, 1955) American computer scientist and businessman found out that practical problems show much better speedup than Amdahl predicted. In 1988 Dr. Gustafson wrote the paper Re-evaluating Amdahl's Law, which later became referred to as "Gustafson's Law" or the "Gustafson-Barsis Law." Dr. Gustafson summarizes this work:[8][9]"...The model is not a contradiction of Amdahl's law as some have stated, but an observation that Amdahl's assumptions don't match the way people use parallel processors. People scale their problems to match the power available, in contrast to Amdahl's assumption that the problem is always the same no matter how capable the computer.

"A firestorm resulted after this paper was published. Defenders of traditional serial thinking had believed Amdahl's model allowed them to dismiss parallel processing as academic foolishness, and many scrambled to reassert the validity of Amdahl's assumptions. Now, few argue that parallel processing isn't viable; this paper marked a turning point in the attitude of the computing industry toward parallel processing."

As mentioned, this publication has generated volumes of discussion and commentary in the ensuing years. Gustafson's Law has also become a standard part of the parallel-processing academic curriculum [12].

Amdahl's law is suitable for applications where response time is critical. On the other hand, there are many applications which require that accuracy of the resultant output should be high. In the present situation, the computing

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

power has increased substantially due to increase in number of processors attached to a parallel computer. Thus it is possible to increase the size of the problem i.e., the workload can be increased. How does this operate? Gustafson's Law relaxed the restriction of fixed size of problem and aimed at using the notion of constant execution time in order to overcome the sequential bottleneck encountered in Amdahl's Law. This situation which does not assume a fixed workload is analysed by Gustafson. Thus, Gustafson's Law assumes that the workload may increase substantially, with the number of processors but the total execution time should remain the same.

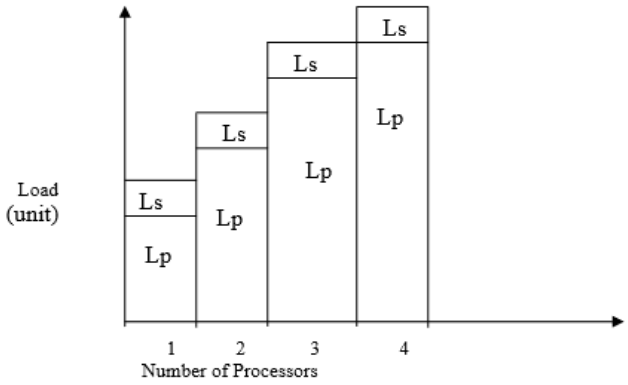

Fig 6: Parallel load increases for Gustafson's law

According to Gustafson's Law, if the number of parallel operations for a problem increases sufficiently, then the sequential operations will no longer be a bottleneck

Example

P -> It is a problem which has to be solved using parallel computer.
T_s -> It is the time taken for executing sequential operation (it is considered as constant).
$T_p(L,N)$ -> it is time taken for executing parallel operations with L as total load on a parallel, Computer with N processor. That is,
Total time taken for finding the problem is $T = T_s + T_p$.
Therefore,

$$S(N) = \frac{T(1)}{T(N)}$$

$$S(N) = Ts + \frac{Tp(1,L)}{Ts + Tp(N,L)}$$

Also, Tp (1, L) = N* Tp (N, L) i.e. time taken to execute parallel operations having a load of L on one processor is equal to the N multiplied by the time taken by one computer having N processor. If α be the fraction of sequential load for a given problem i.e.,

$$\propto = \frac{Ts}{Ts + Tp(N,L)}$$

Substituting Tp (1, L) = N* Tp (N, L) we get

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

$$S(N) = Ts + \frac{N * Tp(N,L)}{Ts + Tp(N,L)}$$

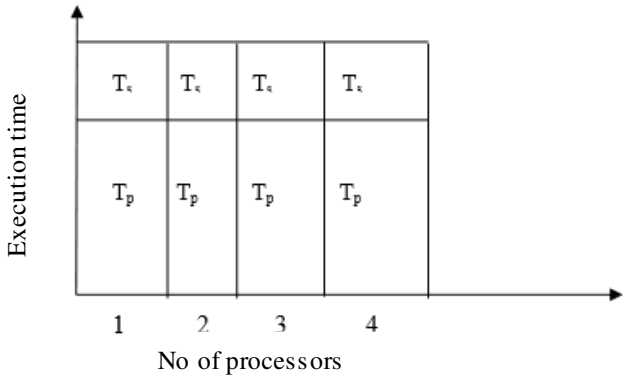

Fig 7: Fixed execution time for Gustafson's law

$$S(N) = \frac{T_s}{T_s + T_p(N,L)} + \frac{N * T_p(N,L)}{T_s + T_p(N,L)}$$

Now changing the equation of S (N) in the form of X

$$S(N) = \alpha + N * (1-\alpha)$$

$$S(N) = N - \alpha * (N-1)$$

Sun-Ni's Law

Sun-Ni's Law (or Sun and Ni's Law, also known as memory-bounded speedup), [13][14] is a memory-bounded speedup model which states that as computing power increases the corresponding increase in problem size is constrained by the system's memory capacity.

In general, as a system grows in computational power, the problems run on the system increase in size.

Analogous to

- Amdahl's law, which assumes that the problem size remains constant as system sizes grow, and
- Gustafson's law, which proposes that the problem size should scale but be bound by a fixed amount of time.

Sun-Ni's Law states the problem size should scale but be bound by the memory capacity of the system.

Sometimes, (especially for FPGA Scientific Acceleration,) memory bandwidth may be more important than memory capacity.

The memory bounded speedup approach, or the Sun-Ni law, is concerned with memory size and how it affects performance. The problem size that can be tackled is affected by the amount of memory available. A limited physical memory means that more time is spent figuring out workarounds to solve a problem within the parallel computing architecture. The approach the Sun-Ni law takes is, if the time limit specified by the fixed-time speedup is met and there is enough memory space, the problem should be scaled to make adequate use of all the available memory.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

This is what the Sun-Ni law does, and the formula considers memory size and relates it to performance. Every processor in a parallel computing architecture has a fixed memory, and the formula relates the problem size to the total available memory capacity. The memory bounded speedup laid out in the Sun-Ni law is, in essence, a generalization of both the fixed-time and fixed-size speedups. Given that the total memory size increases when the number of processors increases, the Sun-Ni law attempts to utilize all that memory space more efficiently.

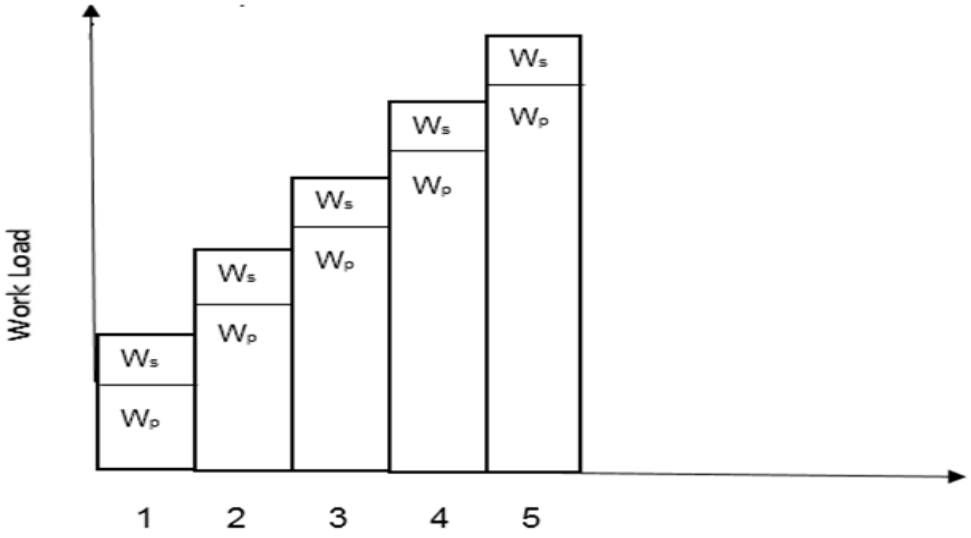

Fig 8: Scaled work load for Sun & Ni's Law

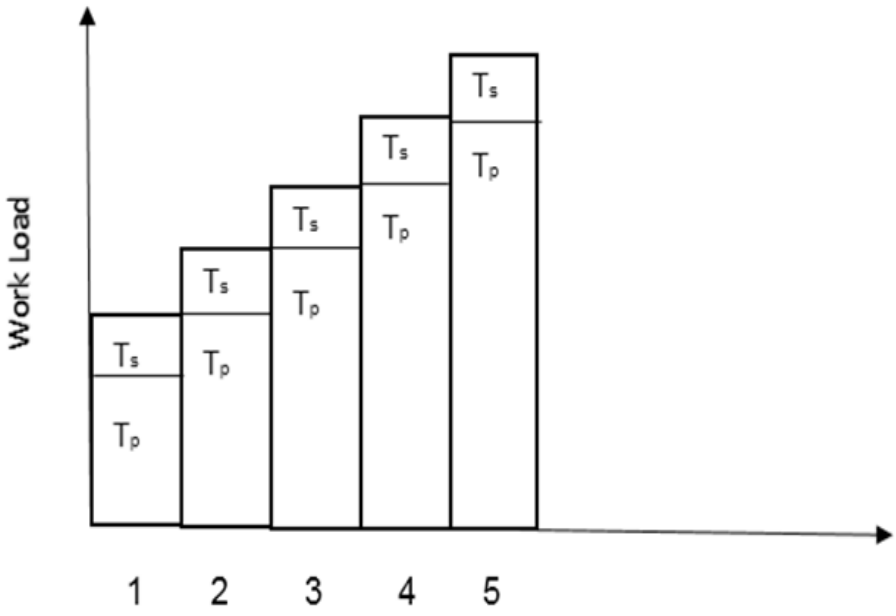

Fig 9: Slightly increasing execution time for Sun & Ni's Law

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

IV. EXPERIMENTAL RESULTS

Here we are considering several number of processors and value of α between 0 to 1 for analysing the speed up laws. Applying the values of n and α in speed formula we get value of S(n) as given in the table 3.

No of Processor	Value of α (0<α≤1)	Amdahl's Law $s(n) = \frac{1}{\alpha + (1-\alpha)/n}$	Gustafson's Law $s(n) = n - \alpha(n-1)$	Sun and Ni's Law $s(n) = \frac{\alpha + G(n)(1-\alpha)}{\alpha + \frac{G(n)}{n} * (1-\alpha)}$
2	0.1	1.81	1.9	1.98
4	0.2	0.86	3.4	3.75
6	0.3	0.8	4.5	5.06
8	0.4	0.74	5.2	5.71
10	0.5	0.68	5.5	5.71

Table 3: Comparative analysis of speed up performance in terms of No of processor and value of α

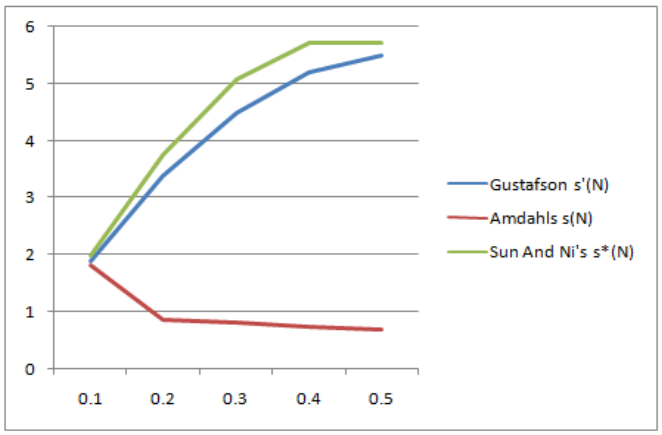

Fig 10 : Graphical representation of Table 3

V. CONCLUSION

The paper explain comparative study of speed up laws such as Amdahl's Law, Gustafson's Law, and Sun & Ni's law in terms of no of processors and value of α. Amdahl's law stated that for a given problem size the speed up does not increases as no of processor increases. In fact the speed tends to become saturated. In Gustafson's law as the machine size increases the problem size (workload) increases so as to keep the fixed execution time for the problem. The Sun &Ni's law generalize both Amdahl's law and Gustafson's law to maximize the use of both processor and memory

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003120

capacities. Measuring the performance in terms of speed up factor is more important in parallel computing. The paper mainly discuss about speed up laws such as Amdahl's Law, Gustafson's Law, and Sun & Ni's law. By comparing the speed up laws in terms of S(n) by considering n= 2 to 10 and α=0 to 1we get an analysis report such that speed up factors S(n), S'(n), S*(n) shall be S*(n)≥S'(n)≥S(n).

REFERENCES

[1] Amdahl, Gene M. (1967). "Validity of the single processor approach to achieving large scale computing capabilities". Proceeding AFIPS '67 (spring) Proceedings of the April 18–20, 1967, Spring Joint Computer Conference: 483–485.
[2] Wang, W. (n.d.). The Limitations of Instruction-Level Parallelism and Thread-Level Parallelism. Retrieved November 30, 2020,
[3] Cray-1. (2020, November 23). Retrieved December 02, 2020, from https://en.wikipedia.org/wiki/Cray-1
[4] Multi-core processor. (2020, November 18). Retrieved December 03, 2020, from https://en.wikipedia.org/wiki/Multi-core_processor
[5] Bernstein, A. J. (1 October 1966). "Analysis of Programs for Parallel Processing". IEEE Transactions on Electronic Computers. EC-15 (5): 757–763.
[6] 10.1145/1465482.1465560Concurrency is not Parallelism", Waza conference Jan 11, 2012
[7] Michael McCool; James Reinders; Arch Robison (2013). Structured Parallel Programming: Patterns for Efficient Computation. Elsevier. p. 61.
[8] Thomas Rauber; Gudula Rünger (2013). Parallel Programming: for Multicore and Cluster Systems. Springer Science & Business Media. p. 2. ISBN 9783642378010.
[9] Gustafson, John L. (May 1988). "Reevaluating Amdahl's law". Communications of the ACM. **31** (5): 532–533. CiteSeerX 10.1.1.509.6892. doi:10.1145/42411.42415. S2CID 33937392. Archived from the original on 2007-09-27.
[10] Flynn, Laurie J. (8 May 2004). "Intel Halts Development of 2 New Microprocessors". New York Times. Retrieved 5 June 2012.
[11] Amdahl, G.M. Validity of the single-processor approach to achieving large scale computing capabilities. In AFIPS Conference Proceedings vol. 30 (Atlantic City, N.J., Apr. 18-20). AFIPS Press, Reston, Va., 1967, pp. 483-485.
[12] Benner, R.E., Gustafson, J.L., and Montry, G.R., Development and analysis of scientific application programs on a 1024-processor hypercube," SAND 88-0317, Sandia National Laboratories, Feb. 1988.
[13] https://en.wikipedia.org/wiki/Sun-Ni_law
[14] https://www.easytechjunkie.com/what-is-the-sun-ni-law.htm

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor:
7.512

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर
NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details