Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

An Algorithm for Optimized Time, Cost, and Reliability in a Distributed Computing System

Pankaj Saxena
Department of Computer Applications, Teerthanker Mahaveer University, Moradabad (U.P), INDIA
Email: pankbly@gmail.com
Dr. Kapil Govil
Department of Computer Applications, Teerthanker Mahaveer University, Moradabad (U.P), INDIA
Email: drkapilgovil@gmail.com

----------ABSTRACT----------

Distributed Computing System (DCS) refers to multiple computer systems working on a single problem. A distributed system consists of a collection of autonomous computers, connected through a network which enables computers to coordinate their activities and to share the resources of the system. In distributed computing, a single problem is divided into many parts, and each part is solved by different computers. As long as the computers are networked, they can communicate with each other to solve the problem. DCS consists of multiple software components that are on multiple computers, but run as a single system. The computers that are in a distributed system can be physically close together and connected by a local network, or they can be geographically distant and connected by a wide area network. The ultimate goal of distributed computing is to maximize performance in a time effective, cost-effective, and reliability effective manner. In DCS the whole workload is divided into small and independent units, called tasks and it allocates onto the available processors. It also ensures fault tolerance and enables resource accessibility in the event that one of the components fails. The problem is addressed of assigning a task to a distributed computing system. The assignment of the modules of tasks is done statically. We have to give an algorithm to solve the problem of static task assignment in DCS, i.e. given a set of communicating tasks to be executed on a distributed system on a set of processors, to which processor should each task be assigned to get the more reliable results in lesser time and cost. In this paper an efficient algorithm for task allocation in terms of optimum time or optimum cost or optimum reliability is presented where numbers of tasks are more then the number of processors.

Keywords - **Cost, Distributed Computing System (DCS), Reliability, Task, Time.**

Date of Submission: 14, February 2013 Date of Acceptance: 10, March 2013

I. INTRODUCTION

Distributed Computing System (DCS) is a collection of independent computers interconnected by transmission channels that appear to the users of the system as a single computer. Distributed systems are groups of networked computers. The word distributed in terms such as DCS, referred to computer networks where individual computers were physically distributed within some geographical area. The terms are nowadays used in a much wider sense. Each node of DCS is equipped with a processor, a local memory, and interfaces. The purpose of the distributed system is to coordinate the use of shared resources or provide communication services to the users. In distributed computing, each processor has its own private memory (distributed memory). The processors in a typical distributed system run concurrently in parallel. The required processing power for task assignment [25] applications in a DCS can not be achieved with a single processor. One approach to this problem is to use DCS that concurrently process an application program by using multiple processors. A task is a smallest identifiable and essential piece of a job that serves as a unit of work, and as a means of differentiating between the various components of a project. It can also be understand as usually assigned piece of work to the processor often to be finished within a certain time. A task is a piece of code that is to be executed and task allocation [2, 10, 16] refers to the way that tasks are chosen, assigned, and coordinated. Execution time is the time in which a single instruction is executed. Execution cost [1, 5] can be termed as the amount of value of resource used. The execution cost of a task depends on the processor on which it is executed (heterogeneous processors) and the communication between two tasks depends only on whether or not they are assigned to the same processor (homogeneous network). Cost factor can be reduced by taking advantage of heterogeneous computational

Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

capabilities. Reliability [3, 4, 8] is defined to be the probability that the system will not fail during the time that it is processing the tasks. The problem of task assignment [17, 20, 21] in DCS has been studied for many years with many variations. It is one of the fundamental optimization problems in DCS. We consider that communicating tasks are to be assigned to different processors with communication links to minimize the overall execution and communication cost [7] and to maximize the reliability [11]. To optimize the performance of a DCS, several issues arise such as the minimization of time and cost as well as maximization of system reliability [9, 26].

2. OBJECTIVE

The objective of this paper is to design an algorithm to enhance the performance of distributed network in terms of optimized processing time, cost and reliability [15]. The objective of this paper includes allocation of tasks statically [6, 14] and to calculate the overall optimized processing time, cost and reliability in a Distributed Computing System (DCS). The main objective is to give an algorithm in a better and easy way because many algorithms exist in this reference so it is also an objective to experiment the algorithm on different inputs for developing a skilled algorithm. There are certain other things such that a proper balancing of load [24] on processors in a way that no processor should be in an idle mode is also an objective in preparing the algorithm. Objective is also that there must be the full advantage by taking the tasks statically [19] in completing the work for getting optimized time, cost and reliability in an efficient manner.

3. NOTATIONS

T	:	Set of tasks
P	:	Set of processors
CM	:	Communication Matrix
PCTR	:	Processor Cost Time Reliability
MPCTR	:	Modified Processor Cost Time Reliability
FPCTR	:	Fused Processor Cost Time Reliability

4. TECHNIQUE

We have considered a set of task T, which contains three tasks t_1, t_2, and t_3, a set of processors P which contains three processors p_1, p_2 and p_3, also every task contains some number of modules. For obtaining the optimal time or cost or reliability for each task initially the emphasis will be on those modules of tasks which have the maximum probability of data transfer. Now, in case of time and cost the elements will be added and in case of reliability they will be multiplied. Now we have taken a matrix in which the time, cost and reliability of modules are defined and define a communication matrix by considering the communication between tasks. On the basis of highest communication we get a matrix namely FPCTR (,,). Now from this table we can get the separate tables for time, cost and reliability. Load count is taken as an integer variable which contains binary values either 1 or 0. We will assign it a value 0 to the processor if no task is assigned otherwise a value 1 will be assigned to the load count. By considering that the preference should be given to the idle processor we assign load count as 1 or 0. Now, in each table we will do the addition of each row and will also take the average of each row on the basis of sum of each row. Now, we will subtract the values from average. Negative and zero values will not be considered. For time and cost [13, 18] minimum value will be allocated and for reliability maximum value will be considered. Now the tasks can be allocated [22, 23] for getting the optimized results in terms of time, cost & reliability [12], also Etime, Ecost and Ereliablity can be calculated. The function for obtaining the overall assignment time [Etime], cost [Ecost], and reliability [Ereliablity] is as follows-

The function for obtaining the overall assignment time [Etime], cost [Ecost], and reliability [Ereliablity] is as follows-

$$\text{Etime} = \left[\sum_{i=1}^{n} \left\{ \sum_{j=1}^{n} ET_{ij} x_{ij} \right\} \right] \quad \text{(i)}$$

$$\text{Ecost} = \left[\sum_{i=1}^{n} \left\{ \sum_{j=1}^{n} EC_{ij} x_{ij} \right\} \right] \quad \text{(ii)}$$

$$\text{Ereliablity} = \left[\prod_{i=1}^{n} \left\{ \sum_{j=1}^{n} ER_{ij} x_{ij} \right\} \right] \quad \text{(iii)}$$

Where, $x_{ij} = \begin{cases} \geq 1, \text{if } i^{th} \text{ task is assigned to } j^{th} \text{ processor} \\ 0, \text{Otherwise} \end{cases}$

5. ALGORITHM

Step I : Start Algorithm

Step II : Take the set of different tasks T, Set of different processors P and different modules in each task.

Step III : Input the matrix PCTR (,,). Select time/cost/reliability data corresponding to each task as needed.

Step IV : Input matrix CM (,,) by considering the communication time between modules of each task.

Step V : Consider each task on the basis of time or cost or reliability.

Step VI : On the basis of Step 5 we get the matrix MPCTR (,,). This matrix will be derived from matrix PCTR (,,).

Step VII : Fused the modules of tasks in MPCTR (,,), on the basis of highest communication we get the matrix FPCTR (,,).

Step VIII : From FPCTR (,,) we can take separate tables for time, cost and reliability.

Step IX : While (all tasks! = allocated)

Step X : {

a. We store the addition and average of each row for time, cost and reliability separately into a table.

Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

b. on selecting corresponding processing time/cost/reliability by subtracting it from average of corresponding row we get revised processing RPRM (,). Negative and zero values will not be considered.

c. for time and cost minimum value will be allocated and for reliability maximum value will be considered from Table I,II and III respectively.

d. assign a load count 1 if the task is allocated , otherwise a 0 check value will be assigned, by doing this the preference will be given to the processor which has a 0 load count to distribute the tasks for balancing the load

}

Step XI : Compute the processor wise overall processing time, cost and reliability.

Step XII : Display the result.

Step XIII : End of algorithm.

6. IMPLEMENTATION

Let us consider a set of tasks T. This set consists three tasks t_1, t_2, and t_3. We may define it as, T= {t_1, t_2, t_3}. Now, consider the task t_1 has a set of task M_1. This set consists the five modules. We may define it as, M_1= {m_{11}, m_{12}, m_{13}, m_{14}, m_{15}}. Now, for task t_2 considers the set of task M_2.This set consists the four modules. We may define it as, M_2= {m_{21}, m_{22}, m_{23}, m_{24}}. Now, for task t_3 considers the set of task M_3. This set consists the six modules. We may define it as, M_3= {m_{31}, m_{32}, m_{33}, m_{34}, m_{35}, m_{36}}. The total number of processors are three and it can be define as, P= {p_1, p_2, p_3}. The graphical representation of this problem is shown in figure 1.

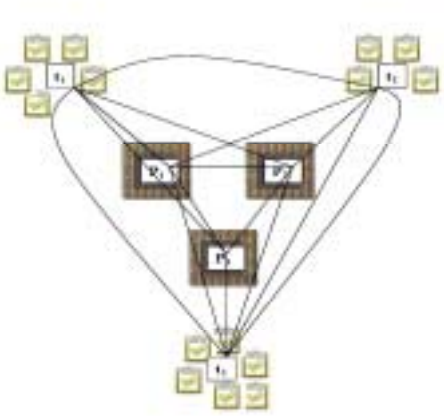

Figure1: Task allocation in DCS

For different task t_1, t_2 and t_3 there are three set of tasks M_1, M_2 and M_3. These set of tasks contains different individual tasks components which are called modules. The processing time (t), cost (c) and reliability (r) of each module of every task on various processors are known and mentioned in the matrix namely, PCTR (,,)-

	Processors	p_1	p_2	p_3
Tasks	Modules	t – c – r	t – c – r	t – c – r
	m_{11}	150 – 2000 – 0.999456	140 – 2000 – 0.999201	130 – 2700 – 0.999301
	m_{12}	160 – 2500 – 0.999321	150 – 3000 – 0.999225	120 – 2800 – 0.999384
t_1	m_{13}	110 – 2600 – 0.999123	110 – 2100 – 0.999223	130 – 2900 – 0.999390
	m_{14}	125 – 2900 – 0.999505	100 – 2700 – 0.999224	110 – 2100 – 0.999701
	m_{15}	130 – 2100 – 0.999555	090 – 2800 – 0.999327	100 – 2600 – 0.999786
	m_{21}	180 – 3000 – 0.999781	090 – 2500 – 0.999329	080 – 2500 – 0.999772
t_2	m_{22}	190 – 2200 – 0.999981	080 – 2800 – 0.999444	070 – 2400 – 0.999775
	m_{23}	080 – 2600 – 0.999220	170 – 2900 – 0.999222	155 – 2300 – 0.999903
	m_{24}	160 – 2700 – 0.999222	150 – 3000 – 0.999701	160 – 2100 – 0.999103
	m_{31}	100 – 2100 – 0.999321	155 – 2000 – 0.999782	160 – 2200 – 0.999144
	m_{32}	190 – 2500 – 0.999425	175 – 2100 – 0.999288	180 – 2700 – 0.999405
t_3	m_{33}	110 – 2600 – 0.999427	180 – 2500 – 0.999277	290 – 2900 – 0.999407
	m_{34}	105 – 2800 – 0.999428	190 – 2700 – 0.999275	210 – 2800 – 0.999409
	m_{35}	150 – 2900 – 0.999429	140 – 2900 – 0.999280	200 – 2700 – 0.999210
	m_{36}	140 – 2300 – 0.999450	110 – 2300 – 0.999281	170 – 2500 – 0.999221

The considered communication time among the modules of each task is mentioned in the matrices, namely CM (,,).

For task t_1, the matrix CM (1,) is as:

	m_{11}	m_{12}	m_{13}	m_{14}	m_{15}
m_{11}	0	9	8	4	3
m_{12}		0	4	6	1
m_{13}			0	3	7
m_{14}				0	2
m_{15}					0

For task t_2, the matrix CM (2,) is as:

	m_{21}	m_{22}	m_{23}	m_{24}
m_{21}	0	4	9	9
m_{22}		0	8	7
m_{23}			0	2
m_{24}				0

For task t_3, the matrix CM (3,) is as:

Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

	m_{31}	m_{32}	m_{33}	m_{34}	m_{35}	m_{36}
m_{31}	0	3	7	4	1	3
m_{32}		0	6	9	8	2
m_{33}			0	6	5	7
m_{34}				0	9	4
m_{35}					0	2
m_{36}						0

Here it is considered that task t_1 is based on the constraint of execution time (one may choose the either cost or reliability constraint), task t_2 is based on the constraint of cost (one may choose the either time or reliability constraint), and task t_3 is based on the constraint of reliability (one may choose the either time or cost constraint).

Hence, we can use the following form of data from matrix PCTR (,,) i.e. execution time for task t_1, execution cost for task t_2, and execution reliability for task t_3 and can get the matrix namely MPCTR (,,) in the following way-

	Processors	p_1	p_2	p_3
Tasks	Modules	t−c−r	t−c−r	t−c−r
	m_{11}	150−⋯−⋯	140−⋯−⋯	130−⋯−⋯
	m_{12}	160−⋯−⋯	150−⋯−⋯	120−⋯−⋯
t_1	m_{13}	110−⋯−⋯	110−⋯−⋯	130−⋯−⋯
	m_{14}	125−⋯−⋯	100−⋯−⋯	110−⋯−⋯
	m_{15}	130−⋯−⋯	090−⋯−⋯	100−⋯−⋯
	m_{21}	⋯−3000−⋯	⋯−2500−⋯	⋯−2500−⋯
	m_{22}	⋯−2200−⋯	⋯−2800−⋯	⋯−2400−⋯
t_2	m_{23}	⋯−2600−⋯	⋯−2900−⋯	⋯−2300−⋯
	m_{24}	⋯−2700−⋯	⋯−3000−⋯	⋯−2100−⋯
	m_{31}	⋯−⋯−0.999321	⋯−⋯−0.999782	⋯−⋯−0.999144
	m_{32}	⋯−⋯−0.999425	⋯−⋯−0.999288	⋯−⋯−0.999405
	m_{33}	⋯−⋯−0.999427	⋯−⋯−0.999277	⋯−⋯−0.999407
t_3	m_{34}	⋯−⋯−0.999428	⋯−⋯−0.999275	⋯−⋯−0.999409
	m_{35}	⋯−⋯−0.999429	⋯−⋯−0.999280	⋯−⋯−0.999210
	m_{36}	⋯−⋯−0.999450	⋯−⋯−0.999281	⋯−⋯−0.999221

Now, the task t_1 have five modules so on the basis of highest communication the modules m_{11}&m_{12}, m_{13}&m_{15} will be fused. The task t_2 have four modules so on the basis of highest communication the modules m_{21}&m_{23} will be fused. The task t_3 have six modules so on the basis of highest communication the modules m_{32}&m_{34}, m_{31}&m_{33}, m_{35}&m_{36} will be fused. The resulting matrix namely FPCTR (,,) will be:

	Processors	p_1	p_2	p_3
Tasks	Modules	t−c−r	t−c−r	t−c−r
	m_{11}*m_{12}	310−⋯−⋯	290−⋯−⋯	250−⋯−⋯
t_1	m_{13}*m_{15}	240−⋯−⋯	200−⋯−⋯	230−⋯−⋯
	m_{14}	125−⋯−⋯	100−⋯−⋯	110−⋯−⋯
	m_{21}*m_{23}	⋯−5600−⋯	⋯−5400−⋯	⋯−4300−⋯
t_2	m_{22}	⋯−2200−⋯	⋯−2800−⋯	⋯−2200−⋯
	m_{24}	⋯−2700−⋯	⋯−3000−⋯	⋯−2300−⋯
	m_{32}*m_{34}	⋯−⋯−0.998853	⋯−⋯−0.998563	⋯−⋯−0.998988
t_3	m_{31}*m_{33}	⋯−⋯−0.998748	⋯−⋯−0.999059	⋯−⋯−0.998433
	m_{35}*m_{36}	⋯−⋯−0.999879	⋯−⋯−0.998561	⋯−⋯−0.999232

Now, from the FPCTR (,,) table we can get three tables namely Table I, Table II and Table III which are given below-

Table I: Processing Time

	m_{11}*m_{12}	m_{13}*m_{15}	m_{14}
p_1	310	240	125
p_2	290	200	100
p_3	250	230	110

Table II: Processing Cost

	m_{21}*m_{23}	m_{22}	m_{24}
p_1	5600	2200	2700
p_2	5400	2800	3000
p_3	4300	2200	2300

Table III: Processing Reliability

	m_{32}*m_{34}	m_{31}*m_{33}	m_{35}*m_{36}
p_1	0.998853	0.998748	0.999879
p_2	0.998563	0.999059	0.998561
p_3	0.998988	0.998433	0.999232

Now for Table I which is for time factor, we will take the sum and average of each row, then will subtract the processing time from average. Now Assign a Load Count 1 if the task is allocated, otherwise a 0 Load Count will be assigned, by doing this the Preference will be given to the

Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

processor which has a 0 Load Count to distribute the tasks for balancing the load. Negative and zero values will not be considered. Now for time and cost we will consider the minimum value and for reliability maximum value will be taken for allocating to the processor from the above table. By implementing these steps in Table II which is for cost factor and in Table III which is for reliability factor we will get the following three tables-

Table IV: Overall Processing Time

processors	Allocated task	Time	Load Count	Etime
p_1	$m_{13} * m_{15}$	240	1	
p_2	$m_{11} * m_{12}$	290	1	640
p_3	m_{14}	110	1	

Results of Table IV can be shown graphically as in figure 2-

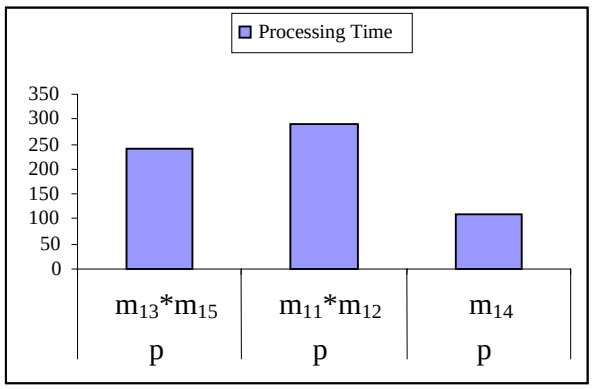

Figure2: Processing Time by Processors

Table V: Overall Processing Cost

processor	Allocated Task	Cost	Load Count	Ecost
p_1	$m_{22} * m_{23}$	5600	1	
p_2	m_{21}	2800	1	10700
p_3	m_{24}	2300	1	

Results of Table V can be shown graphically as in figure 3-

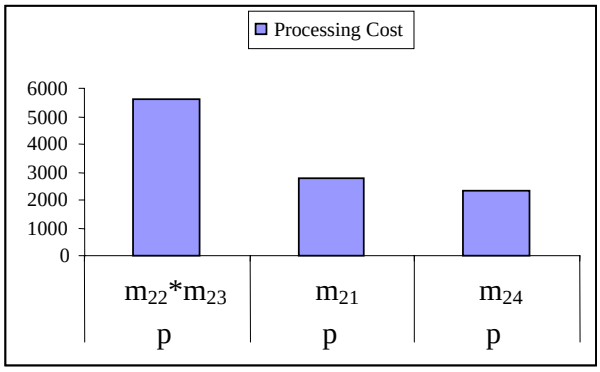

Figure 3: Processing Cost in Completing Tasks

Table VI: Overall Processing Reliability

Processor	Allocatede Task	Reliablity	Check Value	Ereliablity
p_1	$m_{34} * m_{35}$	0.998853	1	
p_2	$m_{31} * m_{32}$	0.998561	1	0.995852
p_3	$m_{33} * m_{36}$	0.998433	1	

Results of Table VI can be shown graphically as in figure 4-

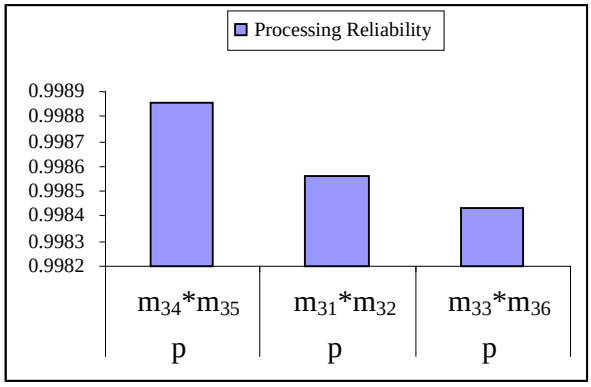

Figure 4: Processing Reliability in Completing Tasks

7. CONCLUSION

In the present paper conclusively we design an algorithm for achieving better performance to enhance the efficiency on Distributed Computing System (DCS) in terms of optimized time, cost and reliability. We have taken a problem to assign the task to the processors where the numbers of tasks are greater then the number of processors in a DCS and the presented algorithms on different inputs

Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

show better performance while comparing the results with some other algorithms. We get this conclusion by taking three tasks namely t_1, t_2 and t_3 with many modules, for process the task t_1 in minimum time, task t_2 in minimum cost and task t_3 with maximum reliability. This algorithm shows the optimized results as shown in the table given below-

	Processors			Optimal	Optimal	Optimal
Tasks	p_1	p_2	p_3	Etime	Ecost	Ereliablity
t_1	m_{13} *m_{15}	m_{11} *m_{12}	m_{14}	640	…	…
t_2	m_{22} *m_{23}	m_{21}	m_{24}	…	10700	…
t_3	m_{34} *m_{35}	m_{31} *m_{32}	m_{33} *m_{36}	…	…	0.995852

The analysis of any algorithm specifically emphasis on time complexity. The time complexity is a measure of the amount of time required to execute an algorithm. Time complexity expresses the relationship between the size of the input and the run time for the algorithm. It is a function of input size 'n'.The time complexity of the above mentioned algorithm is O (mn). By taking several input examples, the above algorithm gives the results mentioned below-

No.of Processors(n)	No. of task(m)	Optimal Results
3	4	12
3	5	15
4	6	24
4	7	28
5	8	40
5	9	45
6	10	60
6	11	66
7	12	84
7	13	91

The graphical representation of the above results are shown by figure 5, 6, 7, 8 and 9 as given below-

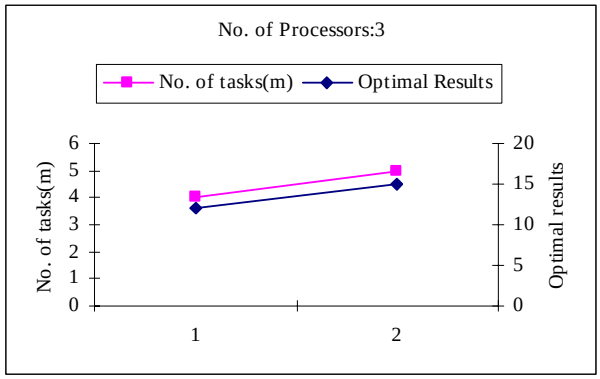

Figure 5: Processor Wise Complexity

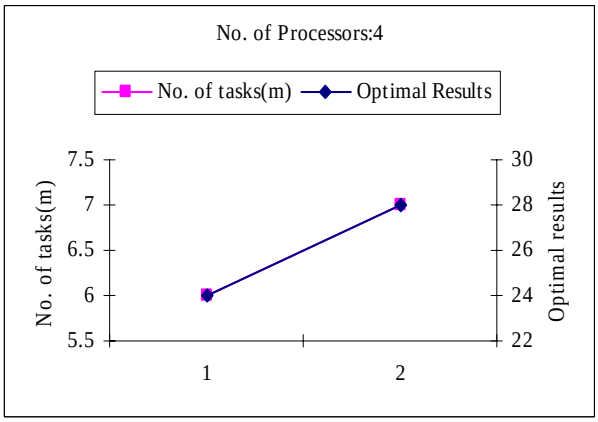

Figure 6: Processor Wise Complexity

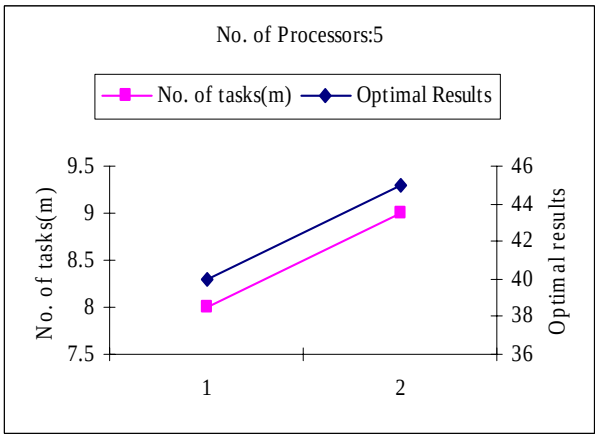

Figure 7: Processor Wise Complexity

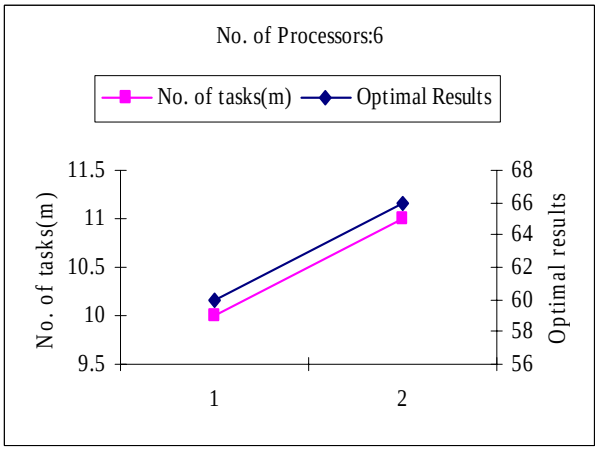

Figure 8: Processor Wise Complexity

Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

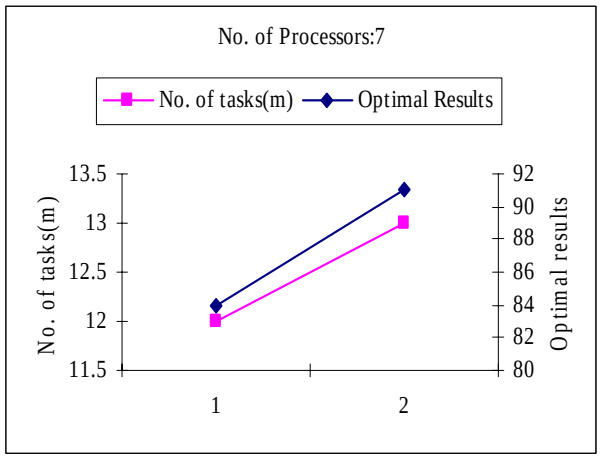

Figure 9: Processor Wise Complexity

The performance of the suggested algorithm is compared with the algorithm suggested by H.kumar et al [13] as the given table shows the comparison of time complexity between algorithm [13] and our algorithm-

Processors(n)	Tasks(m)	Time complexity of algorithm[13] $O(m^2 + mn)$	Time complexity of Present algorithm O(mn)
3	4	28	12
3	5	40	15
4	6	60	24
4	7	77	28
5	8	104	40
5	9	126	45
6	10	160	60
6	11	187	66
7	12	228	84
7	13	260	91

From the above table it is clear that algorithm proposed by us is much better for the optimal allocation of tasks for enhancing the performance of distributed computing system. The graphical representation is given below between algorithm [13] and present algorithm by figure 10, 11, 12, 13 and 14.

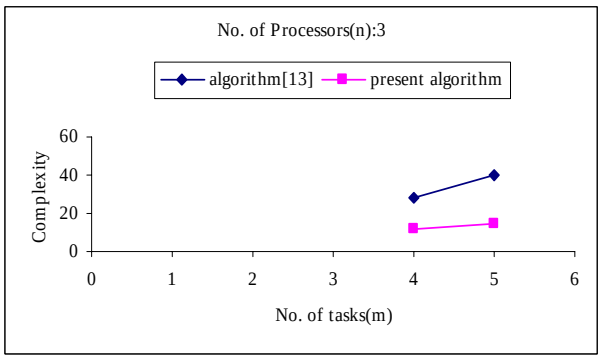

FIGURE 10: Complexity Comparisons

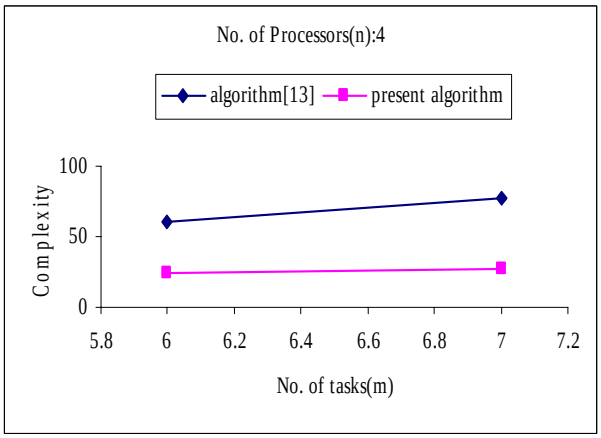

FIGURE 11: Complexity Comparisons

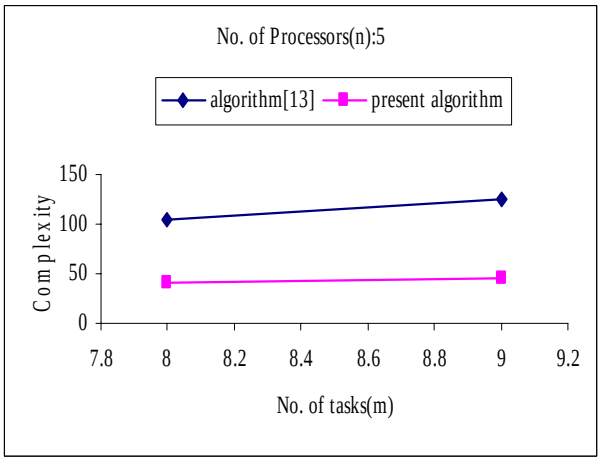

FIGURE 12: Complexity Comparison

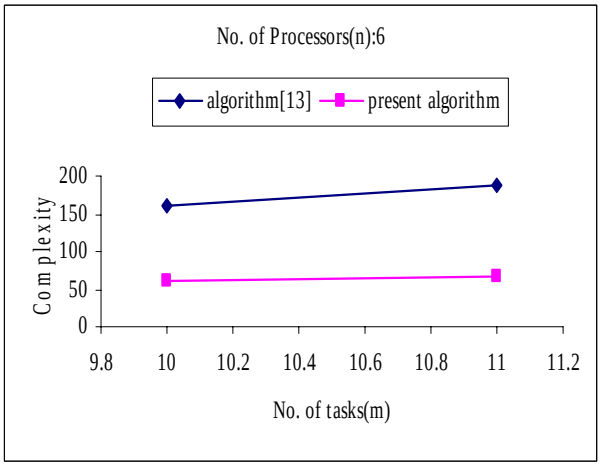

FIGURE: 13 Complexity Comparisons

Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

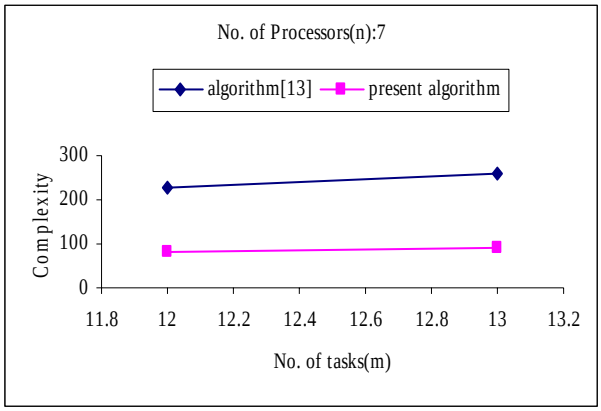

FIGURE: 14 Complexity Comparisons

8. REFERENCES

[1] Anju Khandelwal, Optimal Execution Cost of Distributed System through Clustering, *International Journal of Engineering Science and Technology*, *3*(3), 2011, 2320-2328.

[2] Ahmed Younes, and Hamed, Task Allocation for Minimizing Cost of Distributed Computing Systems Using Genetic Algorithms, *International Journal of Advanced Research in Computer Science and Software Engineering*, *2*(9), 2012, 1202-1209.

[3] Ajit Kimar Verma, and Mangesh Trimbak Tamhankar, Reliability Based optimal Task Allocation in Distributed Database Management Systems, *IEEE Transactions on Reliability*, *46*(4), 1997, 453-459.

[4] Anurag Raii, and Vikram Kapoor, Reliable Clustering Model for Enhancing Processors Throughput in Distributed Computing System, *International Journal of Computer Applications*, *38*(8), 2012, 47-50.

[5] Boeres, Cristina1, Rebello, and Vinod E.F., A versatile cost modeling approach for multicomputer task scheduling, *Journal of* Parallel Computing, *25*(1), 1999, 63-86.

[6] Braun Tracy D, Siegel Howard Jay, Maciejewski Anthony A, and Hong Ye, Static resource allocation for heterogeneous computing environments with tasks having dependencies, priorities, deadlines, and multiple versions, *Journal of Parallel and Distributed Computing*, *68*(11), 2008, 1504-1516.

[7] Bo yang, Huajun Hu, and Suchang Guo**,** Cost-oriented task allocation and hardware redundancy policies in heterogeneous distributed computing systems considering software reliability**,** *Journal Computers and Industrial Engineering*, *56*(4), 2009, 1687-1696.

[8] Dr. Kapil Govil, Processing Reliability based a Clever Task Allocation Algorithm to Enhance the Performance of Distributed Computing Environment, *International Journal of Advanced Networking and Applications*, *3*(1), 2011, 1025-1030.

[9] D. Coit, and A. Smith, Reliability optimization of series-parallel systems using genetic algorithm, *IEEE Transaction on Reliability*, *45*(2), 1996, 254-266.

[10] G.sagar, Anil K, and Sarj E, Task allocation model for distributed systems, *International Journal of Systems Science*, *22*(9), 1991, 1671-1678.

[11] Gamal Attiya, and Yskandar Hama, Task allocation for maximizing reliability of distributed systems: A simulated annealing approach, Journal *of Parallel and Distributed Computing*, *66*(10), 2006, 1259-1266.

[12] Hsieh, Chung-Chi, Hsieh, and Yi-Che, Reliability and cost optimization in distributed computing systems, *Journal of Computers & Operations Research*, *30*(8), 2003, 1103-1119.

[13] H.Kumar, A task allocation model for distributed data network, Journal of Mathematical Sciences, *1*(4), 2006, 379-392.

[14] Keren A, and Barak A, Opportunity cost algorithms for reduction of I/O and inter process communication overhead in a computing cluster, *IEEE Transaction on Parallel and Distributed Systems*, *14*(1), 2003, 39-50.

[15] Marwa Shouman, Gamal Attiya, and Ibrahim Z. Morsi, Static Workload Distribution of Parallel Applications in Heterogeneous Distributed Computing Systems with Memory and Communication Capacity Constraints, *International Journal of Computer Applications*, *34*(6), 2011, 18-24.

[16] Manoj B.S, Sekhar Archana, and Siva Ram Murthy C, A state-space search approach for optimizing reliability and cost of execution in distributed sensor networks, *Journal of Parallel and Distributed Computing*, *69*(1), 2009, 12-19.

[17] Manisha Sharma, Harendra Kumar, and Deepak Garg, An Optimal Task Allocation Model through Clustering with Inter-Processor Distances in Heterogeneous Distributed Computing Systems, *International Journal of Soft Computing and Engineering*, *2*(1), 2012, 50-55.

[18] Mostapha Zbakh, and Said El Hajji, Task allocation problem as a non cooperative game, Journal *of Theoretical and Applied Information Technology*, *16*(2), 2010, 110-115.

[19] Najjar Faïza, Slimani, and Yahya, Extension of the one-shot semi join strategy to minimize data transmission cost in distributed query processing, *Journal of Information Sciences*, *114*(1), 1999, 1-21.

Volume: 04 Issue: 05 Pages: 1710-1718 (2013) ISSN : 0975-0290

[20] Nirmeen A. Bahnasawy, Fatma Omara, Magdy A. Koutb, and Mervat Mosa, A new algorithm for static task scheduling for heterogeneous distributed computing system, *International Journal of Information and Communication Technology Research*, *1*(1), 2011, 10-19.

[21] Pradeep Kumar Yadav, M.P. Singh, and Kuldeep Sharma, Task Allocation Model for Reliability and Cost optimization in Distributed Computing System, *International Journal of Modeling, Simulation and Scientific Computations*, *2*(2), 2011, 1-19.

[22] P. K. Yadav, M. P. Singh, and Kuldeep Sharma, An Optimal Task Allocation Model for System Cost Analysis in Heterogeneous Distributed Computing Systems: A Heuristic Approach, *International Journal of Computer Applications*, *28*(4), 2011, 30-37.

[23] Peng-Yeng Yin, Shiuh-Sheng Yu, Pei-Pei Wang, and Yi-Te Wang, Task allocation for maximizing reliability of a distributed system using hybrid particle swarm optimization, *Journal of Systems and Software*, *8*(5), 2007, 724-735.

[24] Qin-Ma Kang, Hong He, Hui-Min Song, and Rong Deng, Task allocation for maximizing reliability of distributed computing systems using honeybee mating optimization, *Journal of Systems and Software*, *83*(11), 2010, 2165–2174.

[25] Shu Wanneng, Wang Jiangqing, and Min-Min Chromosome, Genetic Algorithm for Load Balancing in Grid Computing, *International Journal of Distributed Sensor Network*, *5*(1), 2009, 62-63.

[26] Sagar G, Sarje, Anil K. Ahmed, and Kamal U., Task allocation techniques for distributed computing systems: a review, *Journal of Microcomputer Applications*, *12*(2), 1989, 97-105.

[27] Vidyarthi D.P., and Tripathi A.K., Maximizing reliability of distributed computing system with task allocation using simple genetic algorithm, *Journal of System Architecture*, *47*(6), 2001, 549-554.