# Optimum Service Rate and Service Time in Distributed Computing Environment

**Vanita Ben Dhagat[1], Afshan Butt[2] and Bhanu Pratap Tripathi[3]**
1 Jai Narayan College of Technology, Bhopal
2 Bhilai Institute of Technology ,Kendari, Raipur
3 Govt. N.P.G. Science College, Raipur
E mail : vanita1_dhagat@yahoo.co.in

***Abstract*—Mathematical programming plays a vital role to solve the problem related to science and engineering. Mathematical programming techniques have been used for designing the pattern for *Systematic Allocation* of tasks for evaluation of performance of the Distributed Computing Systems (DCS). In a DCS, a task is allocated to a processor in such a way that extensive Inter Processor Communication (IPC) is avoided and the capabilities of the processor suit to the execution requirements of the task. The Algorithm discussed in this paper provide an optimal solution for assigning a set of "m" tasks of a program to a set of "n" processors (where, m >> n) with the goal to systematic utilization of processors capacity through maximize the overall throughput of the system. The Execution Cost (EC) and Data Transfer Rate (DTR) have been considered while preparing the Algorithm. The present allocation policy involves stepwise modification of Execution Cost Matrix [ECM] and Data Transfer Rate Matrix [DTRM] by making the clusters of tasks, some of the tasks may not involve in any cluster treated as independent tasks. The several sets of input data are considered to test the complexity and efficiency of the algorithm. It is found that the algorithm is suitable for arbitrary number of processor with the random program structure and workable in all the cases and optimize the service rate and service time.**

***Keyword—Distributed Computing System, Task Allocation, Execution Cost, Inter Task Communication Cost, Data Transfer Rate***

## INTORDUCTION

The term "Distributed Computing System (DCS]" is used to describe whenever there are several computers interconnected in some fashion so that a program or procedure running on system terms with multiple processors. However, the term has different meanings with regard to different systems, because processors can be interconnected in many ways for various reasons. In the most general form, the word distribution implies that the processors are in geographically separate locations. Occasionally, the term is also applied to an operation using multiple mini-computers, which are not hardware, connected with each other and are connected through satellite. A user-oriented definition [1, 2] of distributed computing is "Multiple Computers, utilized cooperatively to solve problems".

Distributed computing system has attracted several researchers by posing several challenging problems. In a DCS, the execution of a program may be distributed among several processing elements to reduce the overall cost of execution by taking advantage of inhomogeneous computational capabilities and other resources within the system. The task allocation in a DCS finds extensive applications in the faculties, where large amount of data is to be processed in relatively short period of time, or where real-time computations are required.

The main incentives for choosing DCS are higher throughput, improved availability, and better access to a widely communicated web of information. The increased commercialization of communication system means that ensuring system reliability is of critical importance. Inherently, distributed system is more complex; therefore, it is very difficult to predict the performance of DCS. Mathematical modeling is the tool which can plays an important role to predict the performance of DCS. Therefore, there is an urgent need to develop a method for it. Allocation of tasks in a DCS may be done in verity of ways **(i) Static Allocation** In static allocation when a task is assigned to processor, it remains there while the characteristic of the computation change, and a new assignment must be computed. The phrase "characteristics of the computation" means the ratios of the times that a program spends in different parts of the program. Thus in a static allocation, one is interested in finding the assignment pattern that holds for the life time of a program, and result in the optimum value of the measure of effectiveness. These problems may be categorized in static [3 -13], **(ii) Dynamic Allocation** In order to make the best use of resources in a distributed system, it is essential to reassign modules or tasks dynamically during program execution, so as to the advantage of changes in the local reference patterns of the program [14-18]. Although the dynamic allocation has potential performance advantages, Static allocation is easier to realize and less complex to operate.

Several other methods have been reported in the literature, such as, Integer programming [19,21], Branch and bound technique [22-23], Matrix reduction technique [7,11,13], reliability evaluation to deal with various design and allocation issues in a DCS by [24-34].

The main objective of this paper is to minimize the total program execution cost. The model utilized the mathematical programming technique for execution of the tasks considering when a task is assigned to processor, it remains there while the characteristic of the computation change, and a new assignment must be computed.

## TASK ASSIGNMENT PROBLEM

The specific problem being addressed is as follows: Given application software that consists of “M” communicating tasks, $T = \{t_1, t_2, \ldots.t_m\}$, and a heterogeneous distributed computing system with “N” processors, $P = \{p_1, p_2, \ldots.p_n\}$, where it is assumed that M>>N, assign (allocate) each of the “M” tasks to one of the “N” processors in such a manner that the IPC time is minimized and the processing load is balanced. The processing cost of these tasks on all the processors is given in the form of Execution Cost Matrix [ECM (,)] of order m x n, the ITCC is taken in the form of a symmetric matrix named as Inter Task Communication Cost Matrix [ITCCM (,)], and Data Transfer Rate Matrix [DTRM ( )] which is of order m. It is also assumed that a task can communicate with other tasks by using single channel of more the one channel the

The load balancing, which involves sending load from overloaded processors under loaded processors, should be carried out with due regard for excessive data transfer overhead so that it is accomplished as quickly as possible. It becomes essential to optimize the overall throughput of the processors by allocating the tasks in such a way that the allocated load on all the processors shall have to be balanced. The proposed methodology includes:

- Identification of average load on each Processor
- Identification of allocation
- Identification of total data transfer between the tasks on each processor
- Calculate total Processing Cost of each Processor
- Calculate the throughput of the processors
- Identification of Optimal response time of the system
- Identification of optimal busy time of the system

## DEFINITIONS:

**Execution Cost:** The execution cost $e_{ij}$ Where $1 \leq i \leq m$, $1 \leq j \leq n$ of each task $t_i$ depends on the processor $p_j$ to which it is assigned and the work to be performed by each of tasks of that processor $p_j$. The processing delay cost EC of the tasks on all the processors is given in the form of Execution Cost Matrix (ECM) of order m x n. The Execution Cost of a given assignment on each processors are calculated by Equation (1)

$$PEC(j) = \sum_{j=1}^{n} e_{ij} x_{ij}, i = 1,2,\ldots m \qquad (1)$$

$$\text{Where } x_{ij} = \begin{cases} 1, \text{ if task } t_i \text{ is assigned to procerssor } p_j \\ 0, \text{ otherwise} \end{cases}$$

**Data Transfer Rate [DTR]:** Data Transfer Rate $d_{ik}$ is the per unit time data exchanged between tasks $t_i$ and $t_k$ during the program execution.

$$DTR(P_j) = \min(et_{ij}) * d_{ij} \qquad (2)$$

$$TDT(P_j) = \sum DTR(P_j) \qquad (3)$$

where i =1,2,….m and j=1,2,…n

**Inter Processors Communication Time:**

The Inter Processor Communication time $ct_{ik}$ of the interacting tasks $t_i$ and $t_k$ is the minimum time required for the exchange of data units between the processors during the process of execution.

$$IPC(j) = \sum \min(et_{ij}) \text{ where } i = 1,2,\ldots m \text{ and } j = 1,2,\ldots.n$$

**Response Time (RT) of the System:**

Response time of the system is a function of amount computation to be performed by each processor and the computation time. This function is defined by considering the processor with the heaviest aggregate computation and communication load. Response time of the system for a given assignment is defined by

RT(Aalloc)=

$$\max_{1 \leq j \leq n} \{PEC(Aalloc)_j + IPC\ Aalloc)_j\}$$

## ASSUMPTIONS:

To keep the algorithm reasonable in size several assumptions have been made while designing the algorithm. A program is assumed to be collection of “m” tasks to be executed on a set of “n” processors, which have different processing capabilities. A task may be portion of an executable code or a data file. The number of tasks to be allocated is more than the number of processors (m>>n), as normally is the case in the real life. It is assumed that the execution cost of a task on each processor is known, if a task is not executable on any of the processor due to absence of some resources. The execution cost of that task on that processor is taken to be (∞) infinite. We assume that once a task has completed is execution on a processor, the processor stores the output data of the task in its local memory, if the data is needed by some another task being computed on the same processor, it reads the data from the local memory. Using this fact, the algorithm tries to assign heavily communicating tasks to the same processor. Whenever groups of tasks or cluster are assigned to the same processor, the Inter tasks communication cost between them is zero. Completion of a program from computational point of view means that all related tasks have got executed.

## RESULTS & DISCUSSIONS:

Algorithm has been formulated to model the execution process of tasks in the distributed computing environment. The Algorithm can be used for efficient execution of allocate the tasks in distributed computing system without overloading the processors. To justify the application and usefulness of the present method an example of a distributed computing system is considered with the following Inputs

Input , m= 8, n= 3 ECM(,), DTRM(,), CDM(,)

$$\text{ECM}(,) = \begin{array}{c|ccc|} & p_1 & p_2 & p_3 \\ t_1 & 6 & 3 & 5 \\ t_2 & 4 & 2 & 3 \\ t_3 & 3 & 1 & 2 \\ t_4 & 5 & 2 & \infty \\ t_5 & 3 & 4 & 2 \\ t_6 & 6 & \infty & 6 \\ t_7 & 5 & 6 & 7 \\ t_8 & \infty & 2 & 5 \end{array}$$

$$\text{CDM}(,) \begin{array}{c|cccccccc|} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 \\ t_1 & 0 & 2 & 1 & 1 & 2 & 2 & 1 & 1 \\ t_2 & 2 & 0 & 1 & 1 & 1 & 1 & 1 & 2 \\ t_3 & 1 & 1 & 0 & 2 & 1 & 2 & 1 & 1 \\ t_4 & 1 & 1 & 2 & 0 & 1 & 2 & 1 & 2 \\ t_5 & 2 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ t_6 & 2 & 1 & 2 & 2 & 1 & 0 & 2 & 2 \\ t_7 & 1 & 1 & 1 & 1 & 1 & 2 & 0 & 2 \\ t_8 & 1 & 2 & 1 & 2 & 1 & 2 & 2 & 0 \end{array}$$

$$\text{DTRM}(,) \begin{array}{c|cccccccc|} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 \\ t_1 & 0.000 & 0.333 & 0.250 & 0.500 & 0.167 & 0.125 & 1.000 & 0.000 \\ t_2 & 0.333 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.200 \\ t_3 & 0.250 & 0.000 & 0.000 & 0.250 & 0.333 & 0.500 & 0.000 & 0.000 \\ t_4 & 0.500 & 0.000 & 0.250 & 0.000 & 0.200 & 0.333 & 0.500 & 0.200 \\ t_5 & 0.167 & 0.000 & 0.333 & 0.200 & 0.000 & 0.000 & 0.000 & 0.000 \\ t_6 & 0.125 & 0.000 & 0.500 & 0.333 & 0.000 & 0.000 & 0.167 & 0.125 \\ t_7 & 1.000 & 0.000 & 0.000 & 0.500 & 0.000 & 0.167 & 0.000 & 0.200 \\ t_8 & 0.000 & 0.200 & 0.000 & 0.200 & 0.000 & 0.125 & 0.200 & 0.000 \end{array}$$

The present paper deals with a simple yet efficient mathematical and computational algorithm to identify the *Systematic Allocation* of tasks for evaluation of performance of the Distributed Computing Systems. A simple procedure has been developed to determine the following:

- Systematic Allocation of tasks in DCS
- Mean service rate,
- Mean service time
- Throughput of the processors

The optimization results from the algorithm ensure overall system cost as well as load on the processors are optimally balanced. Table-1 shows that 3 tasks are executing on processor $p_1$ , 3 tasks are executing on $p_2$ and 2 tasks are executing on $p_3$.

| Tasks | Processor |
|---|---|
| $t_3$ | $p_1$ |
| $t_5$ | $p_2$ |
| $t_7$ | $p_1$ |
| $t_2$ | $p_2$ |
| $t_4$ | $p_2$ |
| $t_8$ | $p_2$ |
| $t_1$ | $p_3$ |
| $t_6$ | $p_3$ |

**Table –1:** Assignment obtained by the Algorithm

Table 2 shows that results of the algorithm form the table it is concluded that maximum busy time of the systems as 61.989 which is related to processor p1. Therefore, the optimal time of the system is 61.989. Throughput of the processors is 0.273, 0.500 and 0.182. The average throughput of the DPS is 0.318.

**Table – 2** : Results obtained by the Algorithm

| Processors | EC | IPC | Mean service rate | Throughput of the processors | Mean service time | Total |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | (2+3+6) |
| $p_1$ | 11 | 40 | 0.091 | 0.273 | 10.989 | 61.989 |
| $p_2$ | 6 | 30 | 0.167 | 0.500 | 5.988 | 35.988 |
| $p_3$ | 11 | 34 | 0.091 | 0.182 | 10.989 | 55.989 |

Figure-1 shows there is a direct relation between the mean service rate and throughput of the processors.

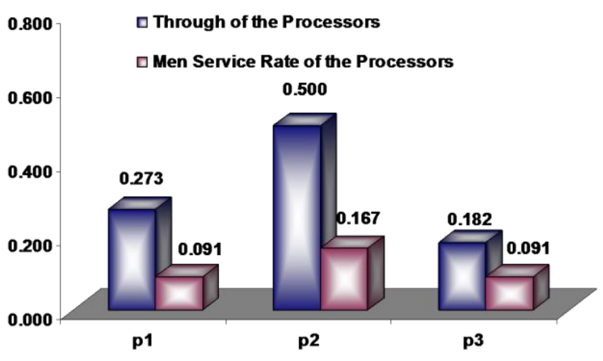
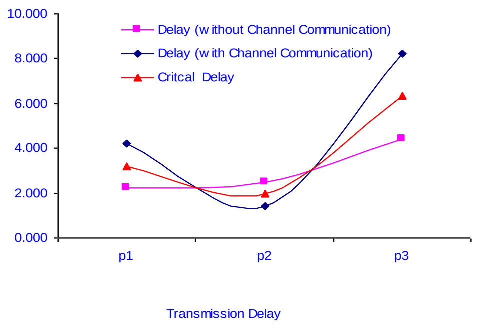


**Figure-1** Relation between Mean service rate & Throughput of the processors

Figure-2 shows the transmission delay due to data exchange with or without channel between the tasks executing on different processors.



The Performance of the algorithm is compared Sagar, G., and Sarje, A.K. (13). The algorithm suggested in Sagar, G., and Sarje, A.K. (13) is not considered the criteria of load balancing and proper utilization of each processor whereas our model considered both the issues. The run time complexity of the algorithm suggested by Richard R.Y., Lee, E.Y.S. and Tsuchiya, M. (22) is o($n^m$) which to high and the show the problem is NP-Hard and the run complexity of the algorithm presented in this paper is o(5m2+2mn), which is much less then that the proposed by Richard R.Y., Lee, E.Y.S. and Tsuchiya, M. (22). This optimize the service rate ad service time.

## REFERENCES

1. K.K. Bhutani, "Distributed Computing", The Indian Journal of Telecommunication, pp. 41-44, 1994.
2. B.R. Sitaram, "Distributed Computing – A User's View Point", CSI Communications, Vol.-18 No. 10, pp.26,28, 1965.
3. Baca, D.F. (1989), Allocation Modules to Processor in a Distributed System, IEEE Transactions on Software Engineering, vol. SE-15, 1427-1436.
4. Coit, D.W. and Smith, A.E. (1996), Reliability Optimization of Series Parallel Systems using a Genetic Algorithm, IEEE Transactions on Reliability, vol. R-45, 254-260.
5. Kumar, V. Singh, M.P. and Yadav, P.K. (1995), A Fast Algorithm for Allocating Tasks in Distributed Processing System, Proc. of the 30th Annual Convention of CSI, held at Hyderabad, India 347-358.
6. Kumar, V. Singh, M. P. and Yadav, P.K. (1995), An Efficient Algorithm for Allocating Tasks to Processors in a Distributed System, Proc. of the 19th National system conference, SSI, held at Coimbatore, India 82-87.
7. Kumar, V. Yadav, P.K. and Bhatia, K. (1998), Optimal Task Allocation in Distributed Systems owing to Inter Tasks Communication Effects, Proc. of the 33rd Annual convention of system society of India, held at New Delhi, India 369-378.
8. Singh, M.P., Kumar, V. and Kumar, A. (1999), An Efficient Algorithm for Optimizing Reliability Index in Tasks-Allocation, Acta Ciencia Indica, vol. xxv m, 437-444.
9. Srinivasan, Santhanam and Jha. K. Niraj (1999), Safety and Reliability Driven Task Allocation in Distributed System, IEEE Transactions on Parallel and Distributed Systems, vol. 10, 238-250.
10. Tillman, F. A., Hwang, C. L. and Kuo, W. (1977), Determining Component Reliability and Redundancy for Optimum System Reliability, IEEE Transactions on Reliability, vol. R-26, 162-165.
11. Yadav, P. K. and Kumar, Avanish (2002), An Efficient Static Approach for Allocation through Reliability Optimization in Distributed Systems", presented at the International conference on Operations Research for Development (ICORD2002) held at Chennai.
12. Zahedi, E., and Ashrafi, N. (1991), Software Reliability Allocation based on Structure, Utility, Price and Cost, IEEE Transactions on Software Engineering, vol. -17, 345-356.
13. Sagar, G., and Sarje, A.K. (1991), Task Allocation Model for Distributed System, Int. J. System Science, vol. 22, 1671-1678.
14. Bokhari, S.H. (1979), Dual Processor Scheduling with Dynamic Re-Assignment, IEEE Transactions on Software Engineering, vol. SE-5, 341-349.
15. Casavent, T.L. and Kuhl, J. G. (1988), A Taxonomy of Scheduling in General Purpose Distributed Computing System, IEEE Transactions on Software Engineering, vol. SE-14, 141-154.
16. Kumar, Avanish (1999), Optimizing for the Dynamic Task Allocation", published to the proceedings of the III Conference of the International Academy of Physical Sciences held at Allahabad, 281-294.

17. Kumar, V. Singh, M.P. and Yadav, P.K. (1996), An Efficient Algorithm for Multi-processor Scheduling with Dynamic Reassignment, Proc. of the 6th National seminar on theoretical Computer Science, held at Banasthally Vidyapeeth, India 105-118.

18. Rotithor, H.G. (1994), Taxonomy of Dynamic Task Scheduling in Distributed Computing Systems, IEEE Proc. Computer Digit Tech., vol. 14, 1-10.

19. Misra, K. B. and Sharma, U., (1991) An Efficient Algorithm to solve Integer Programming Problem arising in System Reliability Design, IEEE Transactions on Reliability, vol. R-40, 81-91.

20. Bulfin, R. L. and Liu, C. Y. (1985), Optimal Allocation of Redundant Components for large Systems, IEEE Transactions on Reliability, vol. R-34, 241-247.

21. Chu, W.W. (1969), Optimal File Allocation in a Multiple Computing System, IEEE Transactions on Computer, vol. C-18, 885-889.

22. Richard R.Y., Lee, E.Y.S. and Tsuchiya, M. (1982), A Task Allocation Model for Distributed Computer System, IEEE Transactions on Computer, vol. C-31, 41-47.

23. Dessoukiu-El, O.I. and Huna, W. H., (1980), Distributed Enumeration on Network Computers, IEEE Transactions on Computer, vol. C-29. 818-825.

24. Fitzgerald, Kent, Latifi, Shahram and Srimani, Pradip K. (2002), Reliability Modeling and Assessment of the Star-Graph Networks, IEEE Transactions on Reliability, vol. R-51, 49-57.

25. Fyffe, D.E., Hines, W. W. and Lee, N. K. (1968), System Reliability Allocation and Computational Algorithm, IEEE Transactions on Reliability, vol. R-17, 64-69.

26. Ghare, P. M. and Taylor, R. E. (1969), Optimal Redundancy for Reliability in Series Systems, Operational Res., vol. 17, 838-847.

27. Kumar, Avanish (2001), An Algorithm for Optimal Index to Tasks Allocation Based on Reliability and cost", published to the proceedings of International Conference on Mathematical Modeling held at Roorkee, 150-155.

28. Lin, Min-Sheng (2002), A Linear-time Algorithm for Computing K-terminal Reliability on Proper Interval Graphs, IEEE Transactions Reliability, vol. R-51, 58-62.

29. Lyu, Michael R., Rangarajan, Sampath and Moorsel, Aad P. A. Van (2002), Optimal Allocation of test Resources for Software Reliability growth modeling in Software Development, IEEE Transactions on Reliability, vol. R-51, 183-192.

30. Nakagawa, Y. and Miyazaki, S. (1981), Surrogate Constraints Algorithm for Reliability Optimization Problems with two Constraints, IEEE Transactions on Reliability, vol. R-30, 175-181.

31. Ormon, Stephen W., Cassady, C. Richard and Greenwood, Allen G. (2002), Reliability Prediction model to Support Conceptual Design, IEEE Transactions on Reliability, vol. R-51, 151-157.

32. Painton, L. and Campbell, J. (1992), Genetic Algorithm in Optimization of System Reliability, IEEE Transactions on Reliability, vol. R-44, 172-178.

33. Peng, Dar-Tezen, Shin, K. G. and Abdel, Zoher T. F. (1997), Assignment Scheduling Communication Periodic Tasks in Distributed real time System, IEEE Transactions on software Engg. vol. SE-13, 745-757.

34. Ramesh, Anapathur V., Twigg, David W., Sandadi, Upender R. and Sharma, Tilak C. (2002), Reliability Analysis of System with Operation Time Management, IEEE Transactions on Reliability, vol. R-51, 39-48.

Yadav, P. K., Kumar, Avanish and Singh, M. P., "An Algorithm for Solving the Unbalanced Assignment Problems", International Journal of Mathematical Sciences, Vol. 12(2), pp. 447-461 200