Task Clustering to Optimize the Reliability of Distributed Systems based on Failure Data Analysis

Sagar Gulati

Technology Education & Research Institute, Kurukshetra, India

Karamjit Bhatia

Department of Computer Science, Gurukul Kangri Vishwavidyalaya, Haridwar, India

P K Yadav

Central Building Research Institute, Roorkee, India

ABSTRACT

Distributed System is the study of geographically separated processors that communicate with one another through message passing. Reliability is the one of the most important factor to be considered in such an environment. The present paper aims at studying how the reliability of a distributed system gets affected with the introduction of probability of failure in the execution and communication process. In the present work, 'm' different tasks of different sizes, are to be executed on 'n' different processors. Every processor has a different execution rate with different failure probability. In this scenario, the reliability of the distributed system is calculated for different task combinations and thus the most reliable solution is determined from set of possible solutions.

KEYWORDS

Task Clustering, Distributed System, Performance Optimization, Reliability Analysis

INTRODUCTION

A distributed system consists of a collection of autonomous computers linked by a computer network and equipped with distributed system software [1]. Distributed systems are implemented on hardware platforms that vary in size from a few workstations interconnected by a single local area network to thousands of computers connected via multiple wide area networks. Distributed processing involves cooperation among several loosely coupled computers communicating over a network. Distributed Processing System provide cost- effective ways for improving computer system's resource sharing, performance, throughput, fault- tolerance, and reliability [2]-[8]. A very common research problem for distributed computing systems is the allocation problem, in which system reliability is to be maximized. These problems are studied by various researchers such as, [9]-[16]. Kumar [15] discussed a task allocation problem for optimizing the execution and communication reliability of a computer communication network. He considered the unreliability matrices for the execution and communication for the purpose of the allocation. Raghavendra et. al. [17] described that the reliability of the distributed computing system depends not only on reliability of a communication network but also on the reliability of the processing nodes and distribution of the resources in the network.

While Shatz et. al. [18] explained when the system hardware configuration is fixed the system reliability mainly depends on the allocation of resources. In the present work, we have introduced a probability of failure in the communication and execution process and simulated the model to see its effect on reliability of the system.

DEFINITIONS

Execution Time

Each task t_i has an Execution Time when executed on j_{th} processor ET_{ij} ($1 \leq i \leq m$ and $1 \leq j \leq n$),

$$ET = \sum_{i=1}^{m} \left\{ \sum_{j=1}^{n} ET_{ij} x_{ij} \right\}$$

Failure Rate

The Failure rate [FR] that task ti shall not get executed per unit time interval on the processor pj is the probability FRij ($1 \leq i \leq m$ and $1 \leq j \leq n$), that task ti will not be successfully executed on processor pj, per unit time interval

$$FR = \prod_{i=1}^{m} \left\{ \sum_{j=1}^{n} FR_{ij} x_{ij} \right\}$$

Inter-Processor Communication Time

The IPC time IPCTij ($1 \leq i \leq m$ and $1 \leq j \leq m$) of the interacting tasks ti and tj is incurred due to the data units exchanged between them during the process of execution.

$$IPCT = \sum_{j=1}^{m} \left\{ \sum_{j=1}^{m} CT_{ij} y_{ij} \right\}$$

Task Communication Rate

The task communication rate is per unit time that a task t_i takes when communicates with task t_j.

$$TCR = \prod_{i=1}^{m} \left\{ \sum_{j=1}^{m} TC_{ij} y_{ij} \right\}$$

Communication Reliability

The Communication Reliability CRij ($1 \leq i \leq m$ and $1 \leq j \leq m$), is the probability of successfully data units exchanged between the task ti and tj under the given conditions.

$$CR = \prod_{i=1}^{m} \left\{ \sum_{j=1}^{m} CR_{ij} y_{ij} \right\}$$

Execution Reliability

The Execution Reliability [ER] of a task ti on the processor pj is the probability ERij ($1 \leq i \leq m$ and $1 \leq j \leq n$), that task ti will be successfully executed on processor pj, within specified conditions.

$$ER = \prod_{i=1}^{m} \left\{ \sum_{j=1}^{n} ER_{ij} x_{ij} \right\}$$

Index

The index represents the ratio of $T_{Reliability}$ to the total cost T_{time}.

Index = $T_{Reliability}$ / T_{Cost}.

$$\text{where } x_{ij} = \begin{cases} 1, \text{if } i^{th} \textit{ task is assigned } \text{to } j^{th} \text{ processor, and} \\ 0, \text{otherwise} \end{cases}$$

$$y_{ij} = \begin{cases} 1, \text{if the } \textit{task assigned } \text{to processor } i \text{ communicates with the task assigned to processor } j \\ 0, \text{otherwise} \end{cases}$$

PROBLEM STATEMENT

Let the given system consists of a set of n processors P = {p_1, p_2, ..., p_n}, with specific processing rate and interconnected by communication links, and a set of m tasks T = {t_1, t_2,.....,t_m}of different size(s) to be executed on these processors, with the possibility of failure in communication and execution process. The proposed model relies upon:

(i) Developing the method for fusing (m-n) excess tasks.

(ii) Formulating Execution Survival Matrix and Communication Survival Matrix.

(iii) Formulating Execution Reliability Matrix and Communication Reliability Matrix

(iv) Formulating the Index based on the time and reliability for the system.

(v) Developing an algorithm to obtain the best Index value for the system.

ASSUMPTIONS

To cope up with the real life problems, and to keep the algorithms reasonable in size, following assumptions have been made:

- The no. of tasks to be allocated is more than the no. of processors, as is the normal case in Distributed Systems.
- Whenever two or more tasks have been assigned to the same processor, Inter-processor communication time between them is assumed to be zero.
- If a task is not executable on a certain processor, due to absence of some resources, the ET of the task on that processor is taken to be infinite.
- The completion of a program from computational point of view means that all related tasks have got executed.
- Reassignment of the task is not possible i.e. allocation policy is static.
- Number of tasks in a cluster is obtained using Ceil (m/n), say ‘R’. Combination of tasks for clusters is obtained as per the order of tasks in the task set. As maximum as possible clusters of tasks are formed. Remaining tasks are considered for making smaller clusters based on the same approach.
- Random number generation method has been used to create PRM(,) and TSM(,), EFM(,), IPCTM(,), TCRM(,) and CFM(,).
- The thrust in this paper has been on obtaining best run time complexity for the algorithm, hence, load balancing on processors has not been considered.

PROPOSED METHOD

A task is allocated to a processor in such a way that reliability of tasks is optimized and the capabilities of the processor suit to the execution requirements of the task. We begin the work with a processor rate matrix PRM (,), task size matrix TSM (,) and task communication rate matrix TCRM (,) along with randomly generated execution failure matrix EFM (,) and communication failure matrix CFM (,). First of all, with the help of PRM (,) and TSM (,), the execution time matrix ETM (,) is calculated followed by the generation of execution survival matrix ESM (,) and communication survival matrix CSM (,). Then corresponding execution reliability matrix ERM (,) as well as communication reliability matrix CRM (,) is also evaluated, using the method suggested by [19].

ETM(,) = PRM(,) * TSM(,)

ESM(,) = ETM(,) – EFM(,)

ERM(,) = ESM(,) / ETM(,)

TSM(,) = TCRM(,) – CFM(,)

CRM(,) = TSM(,) / TCRM(,)

Here, by C(,) = A(,) / B(,) notation, we mean that A, B and C are matrices of same order and the division operation indicates that each element of A is divided by the corresponding element of B and stored at the corresponding position in C.

Thereafter, the proposed allocation policy involves stepwise refinement of ETM(,), ERM(,), and CRM(,) for fusing ‘m’ tasks that are in excess of the number of ‘n’ processors. The process of refinement is continued till the number of tasks become equal to the number of available processors. Now these tasks are assigned to the processors in such a way that their respective ER and CR are maximized.

To begin, we obtain the total number to task combinations required for testing to optimize the performance of distributed systems as [(n * ${}^{m}C_{m-n}$) / Ceil (m/n)]. Let us call it ‘nl’. Number of tasks ‘R’ in a cluster is obtained using Ceil (m/n). Combination of tasks for clusters is obtained as per the order of tasks in the task set. As maximum as possible clusters of ‘R’ tasks are formed, remaining tasks are considered for making smaller clusters based on the same approach. Let tasks ti and tk be the candidate tasks for fusion. To proceed, i^{th} row and k^{th} row of ETM(,) are summed up, and accordingly for ERM(,) are multiplied [19]. If any of the entry of the summed-up/multiplied row is finite, then these tasks are accepted for fusion, otherwise, a new task-pair (again call it (t_i, t_k)) in the order is selected. This process is repeated until the tasks are available for fusion.

Once the task t_i and t_k are selected for fusion, the corresponding row and column entries are replaced by one in CRM(,) and then k^{th} row of CRM(,) is multiplied with its i^{th} row, and k^{th} column of CRM(,) is multiplied with its i^{th} column and then k^{th} row and k^{th} column are deleted from CRM(,). The above process is repeated till all the (m-n) excess tasks are fused. The above stepwise modifications of ETM(,), ERM(,) and CRM(,) consequently reduce these matrices to n x n order. Now the problem remains to determine the optimal task allocation strategy by considering the processing efficiency of individual processor(s).

To allocate the tasks to processors, the minimum values of each row and each column of ETM(,) are obtained. Let min{r_{ij}} represents the minimum row value corresponding to the task t_i lying in j^{th} column and min{c_{ij}} represents the minimum column value for processor p_j lying in i^{th} row. These values are replaced by zero in ETM(,). For allocation, a variant of Hungarian assignment method is employed which allocates a task to a processor where it has minimum ET (i.e. minimum e_{ij} value). Similarly, optimal values of ER is also obtained by applying the same process to ERM(,). Thus, total reliability [Treliability] is evaluated.

Finally an Index, which is based on the reliability along with the execution time of the tasks to the processors, is obtained. The maximum value of the index gives the optimal result.

METHODOLOGY

(i) Input no. of tasks and no. of processors, PRM(,), TSM(,) and EFM(,), IPCTM(,), TCRM(,) and CFM(,), nar : = 0; Tass: = { } ; NETM (,) : = ETM(,);
NERM(,) : = ERM(,) NIPCTM(,) = IPCTM(,);
(ii) Evaluate ETM(,), ESM(,), CSM(,), ERM(,) and CRM(,).
(iii) Calculate the maximum size 'R' of cluster in a combination as :
R = Ceil (m/n)
(iv) Calculate total no. of combinations 'nl' for set of task clusters.
nl = [(n * mCm-n) / Ceil (m/n)].
(v) Get 'n' clusters of tasks. For this, as maximum as possible clusters of tasks of size 'R' are formed in such a way that total number of clusters equals 'n' and minimum size of cluster is 1.
(vi) Get revised of ETM(,), IPCTM(,) and CRM(,) after fusion of tasks.
(vii) Get the allocation of tasks to the processors and find out corresponding optimal values of ET, IPC Time, ER and CR and obtain TTime and Treliability.
(viii) Repeat the process till all the task clusters for a combination are obtained.
(ix) Repeat the process for all task combinations.
(ix) Select the combination for which Index is maximum.

DESCRIPTION OF METHOD WITH PROBLEM INSTANCE

Input:

$$PRM(,) = \begin{bmatrix} P1 & 0.279 \\ P2 & 0.254 \\ P3 & 0.291 \end{bmatrix}$$

$$TSM(,) = \begin{bmatrix} t1 & t2 & t3 & t4 & t5 & t6 & t7 \\ 12 & 9 & 8 & 11 & 7 & 14 & 13 \end{bmatrix}$$

$$EFM(,) = \begin{bmatrix} & P1 & P2 & P3 \\ t1 & 0.0019 & 0.0018 & 0.0072 \\ t2 & 0.0013 & 0.0033 & 0.0061 \\ t3 & 0.003 & 0.004 & 0.008 \\ t4 & 0.0041 & 0.007 & 0.0029 \\ t5 & 0.0011 & 0.007 & 0.0039 \\ t6 & 0.009 & 0.0091 & 0.0018 \\ t7 & 0.0033 & 0.0036 & 0.0072 \end{bmatrix}$$

$$\text{IPCTM }(,) = \left[\begin{array}{cccccccc} & t1 & t2 & t3 & t4 & t5 & t6 & t7 \\ t1 & 0.000 & 2.998 & 3.196 & 3.556 & 2.012 & 3.014 & 1.156 \\ t2 & 2.998 & 0.000 & 3.130 & 2.253 & 3.657 & 1.011 & 2.601 \\ t3 & 3.196 & 3.130 & 0.000 & 3.256 & 2.360 & 3.213 & 2.474 \\ t4 & 3.556 & 2.253 & 3.256 & 0.000 & 1.295 & 3.112 & 4.011 \\ t5 & 2.012 & 3.657 & 2.360 & 1.295 & 0.000 & 1.157 & 2.216 \\ t6 & 3.014 & 1.011 & 3.213 & 3.112 & 1.157 & 0.000 & 1.992 \\ t7 & 1.156 & 2.601 & 2.474 & 4.011 & 2.216 & 1.992 & 0.000 \end{array}\right]$$

$$\text{CFM}(,) = \left[\begin{array}{cccccccc} & t1 & t2 & t3 & t4 & t5 & t6 & t7 \\ t1 & 0.00 & 0.0023 & 0.0018 & 0.0055 & 0.0037 & 0.00 & 0.0041 \\ t2 & 0.0023 & 0.00 & 0.00 & 0.0048 & 0.0033 & 0.0021 & 0.00 \\ t3 & 0.0018 & 0.00 & 0.00 & 0.0018 & 0.0039 & 0.0027 & 0.0044 \\ t4 & 0.0055 & 0.0048 & 0.0018 & 0.00 & 0.0011 & 0.0087 & 0.0090 \\ t5 & 0.0037 & 0.0033 & 0.0039 & 0.0011 & 0.00 & 0.0018 & 0.0073 \\ t6 & 0.00 & 0.0021 & 0.0027 & 0.0087 & 0.0018 & 0.00 & 0.0071 \\ t7 & 0.0041 & 0.00 & 0.0044 & 0.009 & 0.0073 & 0.0071 & 0.00 \end{array}\right]$$

$$\text{TCRM}(,) = \left[\begin{array}{cccccccc} & t1 & t2 & t3 & t4 & t5 & t6 & t7 \\ t1 & 1.00 & 0.97 & 0.96 & 0.94 & 0.93 & 1.00 & 0.94 \\ t2 & 0.97 & 1.00 & 1.00 & 0.97 & 0.96 & 0.93 & 1.00 \\ t3 & 0.96 & 1.00 & 1.00 & 0.97 & 0.98 & 0.96 & 0.99 \\ t4 & 0.94 & 0.97 & 0.97 & 1.00 & 0.96 & 0.94 & 0.92 \\ t5 & 0.93 & 0.96 & 0.98 & 0.96 & 1.00 & 0.92 & 0.93 \\ t6 & 1.00 & 0.93 & 0.96 & 0.94 & 0.92 & 1.00 & 0.91 \\ t7 & 0.94 & 1.00 & 0.99 & 0.92 & 0.93 & 0.91 & 1.00 \end{array}\right]$$

Initialize nar : = 0; far : = { }; T_{ass}: = { } ;

Evaluate the matrices using the formulas mentioned in the proposed method

$$\text{ETM}(,) = \left[\begin{array}{cccc} & P1 & P2 & P3 \\ t1 & 3.348 & 3.048 & 3.492 \\ t2 & 2.511 & 2.286 & 2.619 \\ t3 & 2.232 & 2.032 & 2.328 \\ t4 & 3.069 & 2.794 & 3.210 \\ t5 & 1.953 & 1.178 & 2.037 \\ t6 & 3.906 & 3.556 & 4.074 \\ t7 & 3.627 & 3.302 & 3.783 \end{array}\right]$$

$$\text{ESM}(,) = \left[\begin{array}{cccc} & P1 & P2 & P3 \\ t1 & 3.346 & 3.046 & 3.484 \\ t2 & 2.509 & 2.282 & 2.613 \\ t3 & 2.229 & 2.028 & 2.32 \\ t4 & 3.065 & 2.787 & 3.198 \\ t5 & 1.951 & 1.171 & 2.033 \\ t6 & 3.897 & 3.547 & 4.072 \\ t7 & 3.624 & 3.298 & 3.776 \end{array}\right]$$

$$\text{ERM}(,) = \left[\begin{array}{cccc} & P1 & P2 & P3 \\ t1 & 0.999 & 0.999 & 0.997 \\ t2 & 0.999 & 0.998 & 0.997 \\ t3 & 0.998 & 0.998 & 0.996 \\ t4 & 0.996 & 0.997 & 0.999 \\ t5 & 0.999 & 0.994 & 0.998 \\ t6 & 0.997 & 0.997 & 0.999 \\ t7 & 0.999 & 0.998 & 0.998 \end{array}\right]$$

$$\text{CSM}(,) = \left[\begin{array}{cccccccc} & t1 & t2 & t3 & t4 & t5 & t6 & t7 \\ t1 & 1.000 & 0.9677 & 0.9582 & 0.9345 & 0.9263 & 1.000 & 0.9359 \\ t2 & 0.9677 & 1.000 & 1.000 & 0.9652 & 0.9567 & 0.9279 & 1.000 \\ t3 & 0.9582 & 1.000 & 1.000 & 0.9682 & 0.9761 & 0.9573 & 0.9856 \\ t4 & 0.9345 & 0.9652 & 0.9682 & 1.000 & 0.9589 & 0.9313 & 0.911 \\ t5 & 0.9263 & 0.9567 & 0.9761 & 0.9589 & 1.000 & 0.9182 & 0.9277 \\ t6 & 1.000 & 0.9279 & 0.9573 & 0.9313 & 0.9182 & 1.000 & 0.9029 \\ t7 & 0.9359 & 1.000 & 0.9856 & 0.911 & 0.9277 & 0.9029 & 1.000 \end{array}\right]$$

$$\text{CRM}(,) = \left[\begin{array}{cccccccc} & t1 & t2 & t3 & t4 & t5 & t6 & t7 \\ t1 & 1.000 & 0.998 & 0.998 & 0.994 & 0.996 & 1.000 & 0.996 \\ t2 & 0.998 & 1.000 & 1.000 & 0.995 & 0.997 & 0.998 & 1.000 \\ t3 & 0.998 & 1.000 & 1.000 & 0.998 & 0.996 & 0.997 & 0.996 \\ t4 & 0.994 & 0.995 & 0.998 & 1.000 & 0.999 & 0.991 & 0.990 \\ t5 & 0.996 & 0.997 & 0.996 & 0.999 & 1.000 & 0.998 & 0.992 \\ t6 & 1.000 & 0.998 & 0.997 & 0.991 & 0.998 & 1.000 & 0.992 \\ t7 & 0.996 & 1.000 & 0.996 & 0.990 & 0.992 & 0.992 & 1.000 \end{array}\right]$$

Compute the total combinations and store them in TCOMB ().

TCOMB = [(n * ${}^{m}C_{m-n}$) / Ceil (m/n)] = 35

TCOMB (1) = (123, 456, 7)

$$NETM\,(,) = \begin{array}{cccc} & p_1 & p_2 & p_3 \\ t1*t2*t3 & 8.091 & 7.366 & 8.439 \\ t4*t5*t6 & 8.928 & 7.528 & 9.312 \\ t7 & 3.627 & 3.302 & 3.783 \end{array}$$

$$NIPCTM\,(,) = \begin{array}{cccc} & t1*t2*t3 & t4*t5*t6 & t7 \\ t1*t2*t3 & 0.000 & 24.332 & 6.231 \\ t4*t5*t6 & 24.332 & 0.000 & 8.219 \\ t7 & 6.231 & 8.219 & 0.000 \end{array}$$

$$NCRM(,) = \begin{array}{cccc} & t1*t2*t3 & t4*t5*t6 & t7 \\ t1*t2*t3 & 1.000 & 0.971 & 0.992 \\ t4*t5*t6 & 0.971 & 1.000 & 0.974 \\ t7 & 0.992 & 0.974 & 1.000 \end{array}$$

On applying modified Hungarian method devised by Yada et al [21] to assign the task, min $\{r_i\}$ from NETM(,) for every i, $r_{12} = 7.366$, $r_{22} = 7.528$, $r_{32} = 3.302$. Making $r_{13} = r_{23} = r_{31} = 0$. Again min $\{c_j\}$ from NETM(,) for every j, are $c_{31} = 3.627$, $c_{23} = 0$, $c_{33} = 3.783$. Making $c_{31} = c_{33} = 0$, so that, we get,

$$NETM(,) = \begin{array}{cccc} & p_1 & p_2 & p_3 \\ t1*t2*t3 & 8.091 & 0 & 8.439 \\ t4*t5*t6 & 8.928 & 0 & 9.312 \\ t7 & 0 & 0 & 0 \end{array}$$

Further, NETM(,) is reduced to following:

$$NETM(,) = \begin{array}{cccc} & p_1 & p_2 & p_3 \\ t1*t2*t3 & 0 & 0 & 0.348 \\ t4*t5*t6 & 0.837 & 0 & 1.221 \\ t7 & 0 & 8.091 & 0 \end{array}$$

After implementing assignment process, the first set of the allocation is thus obtained.

Tasks	**Processors**	**ET**	**IPCTime**	**TCRate**	**ER**	**CRel**
t1t2t3	P1	8.091	30.563	0.643	0.990	0.963
t4t5t6	P2	7.528	32.351	0.538	0.992	0.945
t7	P3	3.783	14.45	0.725	0.998	0.966

ET (1) = 19.402

ER (1) = 0.980

IPC Time(1) = 38.782 (Since IPCTM(,) is a symmetric metrix)

TCRate (1) = 0.251

CRel (1) = 0.880

Therefore, Total Reliability TRel = 0.862

Repeating the above process, suggested in the algorithm the corresponding values of ET, TCRate, IPC Time, ER, CRel, TRel and some derived values are obtained and shown in the table 1.

It is concluded that the maximum value of reliability (0.877) as well as Index (0.0135) for the system is for TCOMB (20). Thus, the optimal result is as below:

$$\begin{array}{cccccccc} Tasks & \rightarrow & \Pr ocessors & ER & CR & Treliability & Index \\ t1*t2*t4 & \rightarrow & p1 & & & & \\ t3*t5*t6 & \rightarrow & p2 & 0.985 & 0.891 & 0.877 & 0.0135 \\ t7 & \rightarrow & p3 & & & & \end{array}$$

CONCLUSION

The present paper addresses the problem of allocation of tasks to the processors in distributed computing system to minimize the execution time and optimize the overall reliability. Further, a performance evaluation parameter ‘Index’ based on reliability and time is also evaluated. In this paper, we have chosen the problem in which numbers of tasks ‘m’ are more than the number of processors ‘n’. An efficient algorithm has been proposed to obtain appropriate solution of the problem. The method is presented in computational algorithmic form and implemented on the several sets of input data to test the performance and effectiveness of the algorithm, satisfactory results have been observed.

Table 1 Reliability Evaluation Table

S. No.	TCOMB	ET	IPCTime	TCRate	Total Data Transferd	Time consumed T_{time}	ER	CRel	TRel	Index
1	123,456,7	19.402	38.782	0.251	11.511	65.264	0.980	0.880	0.862	0.0132
2	124,356,7	19.192	35.731	0.256	11.707	64.923	0.985	0.891	0.877	0.0135
3	125,346,7	19.502	40.291	0.261	12.004	65.493	0.927	0.892	0.826	0.0126
4	126,345,7	19.927	45.877	0.221	10.139	65.804	0.974	0.870	0.847	0.0129
5	127,234,6	20.704	46.193	0.216	9.978	66.897	0.962	0.875	0.841	0.0126
6	134,256,7	19.497	45.894	0.290	13.309	65.391	0.965	0.880	0.849	0.0130
7	135,246,7	19.972	45.990	0.272	12.509	65.962	0.925	0.897	0.829	0.0126
8	136,245,7	19.527	35.192	0.220	10.074	65.319	0.938	0.876	0.821	0.0126
9	137,245,6	20.620	46.113	0.235	10.837	66.733	0.948	0.885	0.838	0.0126
10	145,236,7	19.452	45.916	0.267	12.260	65.368	0.976	0.880	0.858	0.0131
11	146,235,7	19.877	45.810	0.217	9.941	65.687	0.974	0.890	0.866	0.0132
12	147,235,6	20.728	46.094	0.256	11.800	66.822	0.955	0.899	0.858	0.0128
13	156,234,7	20.102	46.001	0.231	10.626	66.103	0.934	0.874	0.816	0.0123
14	157,234,6	19.121	45.985	0.256	11.772	65.106	0.931	0.891	0.829	0.0127
15	167,234,5	20.693	46.753	0.231	10.800	67.446	0.929	0.885	0.822	0.0122
16	234,156,7	20.102	46.873	0.231	10.828	66.975	0.929	0.874	0.811	0.0121
17	235,146,7	20.113	46.723	0.217	10.139	66.836	0.973	0.890	0.865	0.0129
18	236,145,7	20.030	46.673	0.266	12.415	66.703	0.946	0.880	0.832	0.0125
19	237,145,6	20.656	46.963	0.251	11.788	67.619	0.973	0.878	0.854	0.0126
20	245,136,7	19.527	46.823	0.220	10.301	66.350	0.938	0.876	0.821	0.0124
21	246,135,7	19.952	45.809	0.272	12.460	65.761	0.965	0.897	0.865	0.0132
22	247,135,6	20.692	46.762	0.246	11.503	67.454	0.953	0.896	0.853	0.0126
23	256,134,7	19.016	45.983	0.290	13.335	64.999	0.909	0.881	0.800	0.0123
24	257,134,6	20.644	46.898	0.245	11.490	67.542	0.944	0.888	0.838	0.0124
25	267,134,5	20.303	46.673	0.273	12.742	66.976	0.927	0.888	0.823	0.0123
26	345,126,7	19.927	45.987	0.221	10.163	65.914	0.964	0.870	0.838	0.0127
27	346,125,7	19.502	45.901	0.265	12.164	65.403	0.937	0.892	0.835	0.0128

28	347,125,6	20.632	46.786	0.256	11.977	67.418	0.939	0.896	0.841	0.0125
29	356,124,7	19.192	46.076	0.256	11.795	65.268	0.978	0.890	0.870	0.0133
30	357,124,6	20.680	46.987	0.235	11.042	67.667	0.946	0.903	0.854	0.0126
31	367,124,5	20.255	46.354	0.256	11.867	66.609	0.919	0.898	0.825	0.0124
32	456,123,7	19.402	45.940	0.251	11.531	65.342	0.929	0.880	0.817	0.0125
33	457,123,6	20.668	45.942	0.256	11.761	66.610	0.957	0.892	0.853	0.0128
34	467,123,5	20.327	46.671	0.279	13.021	66.998	0.939	0.906	0.850	0.0127
35	567,123,4	19.328	45.900	0.285	13.082	65.228	0.926	0.890	0.824	0.0126

Relationships between Trel & Index, IPC time & Total Data Transferred and total time consumed & Index are shown in Fig. 1, Fig. 2 and Fig. 3 respectively.

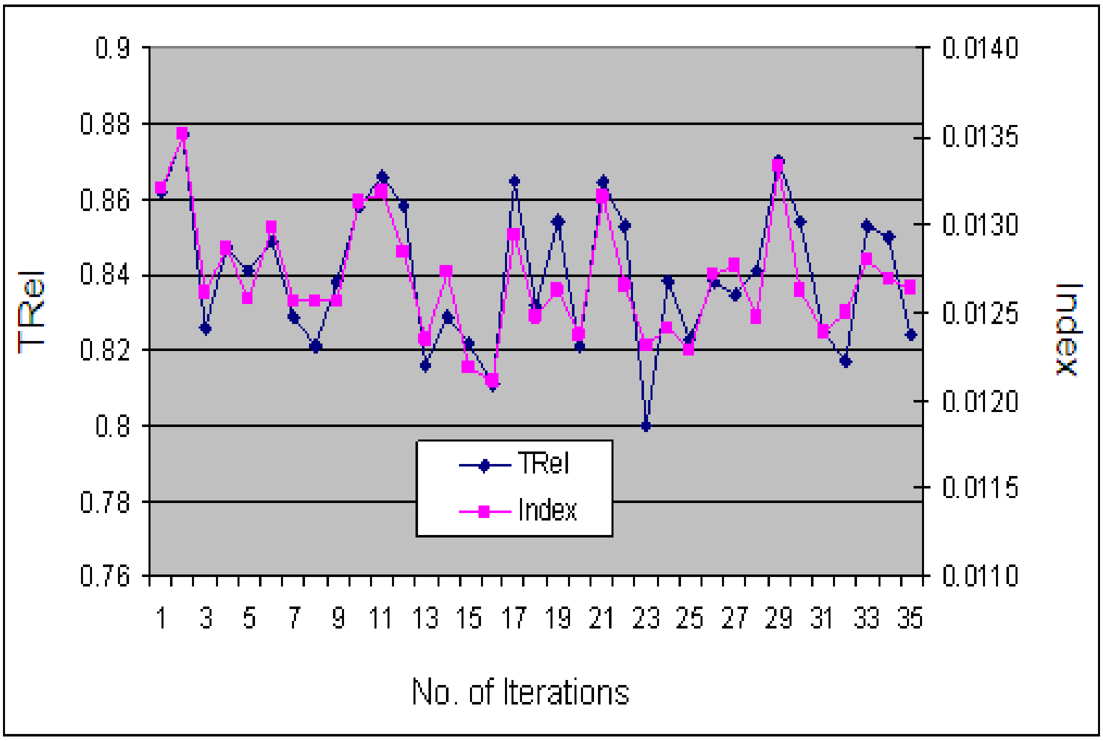


Fig. 1 Reliability vs. Index

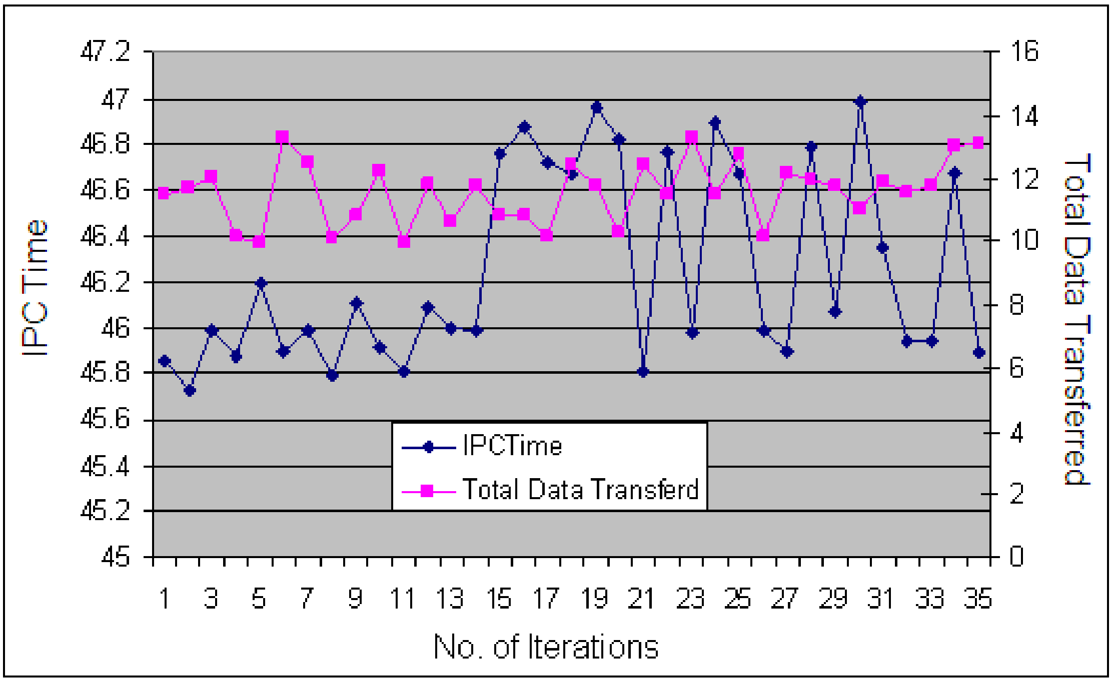


Fig. 2 IPC Time vs. Total data Transferred

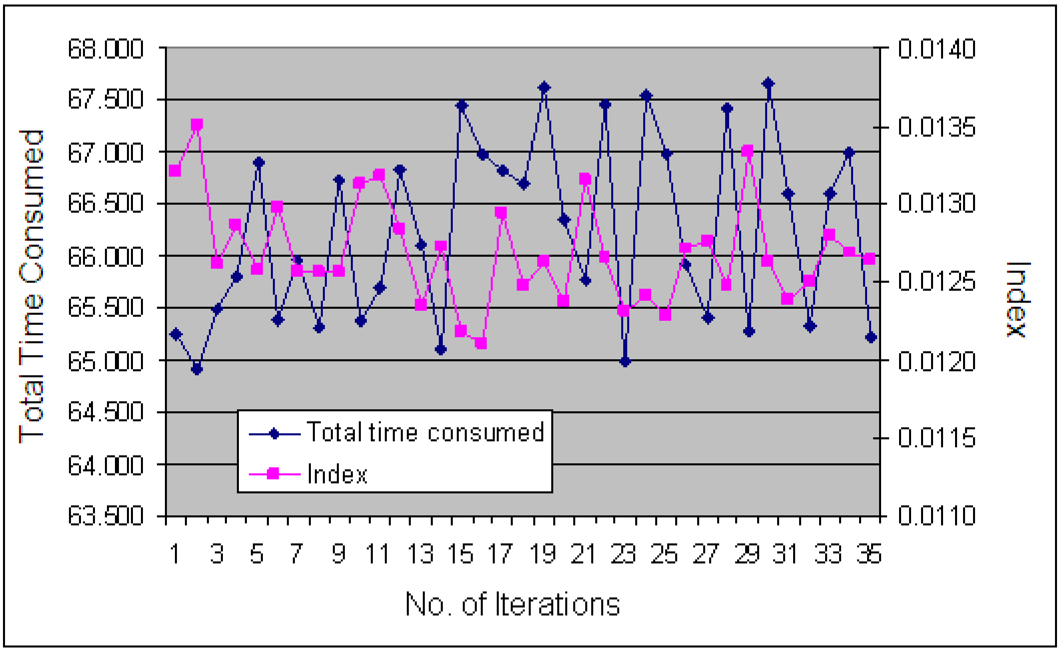
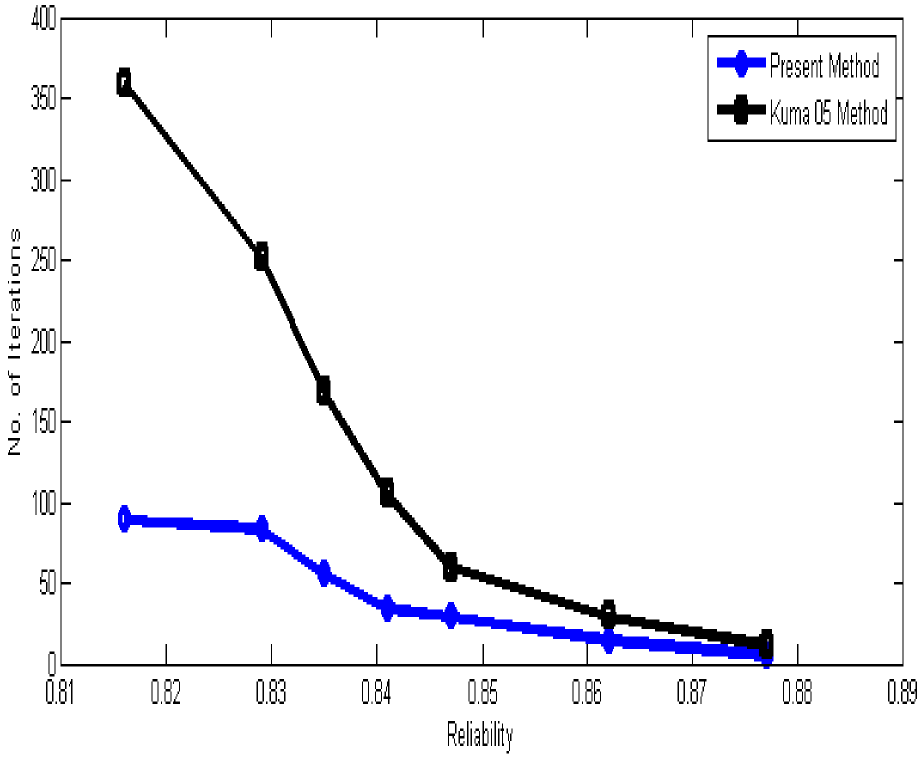


Fig. 3 Time consumed vs. Index

To justify the efficiency of the algorithm, the runtime complexity is evaluated that comes out to be O(mn^2) [23], which is compared with that of [22], i.e. O (m^2n) and the comparison is shown in the Fig. 4 that clarify that the present method's run time complexity is better than that of [22].



,Fig. 4 Comparison of Run-time complexity

REFERENCES

[1] G. Coulouris, J. Dollimore and T. Kindberg, Distributed Systems - Concepts and Design, Second Edition, Addison-Wesley Publications, 2005.

[2] T. C. K. Chou and J. A. Abraham "Load redistribution under failure in distributed systems," IEEE Trans. Comput., vol. C-32, pp. 799-808, Sept. 1983.

[3] D. W. Davies, E. Holler, E. D. Jensen, S. R. Kimbleton, B. W. Lampson, G. Lelann, K. J. Thurber, and R. W. Watson, "Distributed systems architecture and implementation," in LectureNofes in Computer Science, vol. 105, Berlin, Germany: Springer-Verlag, 1981.

[4] P. Enslow, "What is a distributed data processing system," IEEE Comput. Mag., vol. 11, Jan. 1978.

[5] J. Garcia-Molina, "Reliability issues for fully replicated distributed database," IEEE Compuf. Mag., vol. 16, pp. 34-42, Sept. 1982.

[6] S. Hariri, C. S. Raghavendra, and V. K. Prasanna Kumar, "Reliability measures for distributed processing systems," in Proc. 6th Inf. Conf: Distributed Compuf. Syst., pp. 564-571, 1986.

[7] D. A. Rennels, "Distributed fault-tolerant computer systems," IEEE Cornput. Mag., vol. 13, pp. 5545, Mar. 1980.

[8] J. A. Stankovic, "A perspective on distributed computer systems," IEEE Trans. Cornput., vol. C-33, pp. 1102-1115, Dec. 1984.

[9] Sagar, G., and Sarje, A.K., "Task Allocation Model for Distributed system", Int. J. System Science, vol. 22, pp. 1671-1678, 1991.

[10] Kumar, A., Pathak, R. M., Gupta, Y. P., "Genetic Algorithm based Reliability Optimization for Computer Network Expansion", IEEE Transactions on Reliability, Vol. 44, No. 1, pp. 63-72, 1995.

[11] Kubat, Peter, "Assessing throughput and Reliability in Communication and Computer Networks", IEEE Transactions on Reliability, Vol. 37, No. 3, pp.308-311, 1988.

[12] P.Y.R.Ma. "A Task Allocation Model for Distributing Computing System" IEEE Trans. Computers, Vol. 31 no.1.pp. 41-47, 1982.

[13] Kartik, S., and Murthy, C. Siva Ram, "Improved Task Allocation Algorithms to Maximize Reliability of Redundant Distributed Computing Systems", IEEE Transactions on Reliability, Vol. 44, No. 4, pp.575-584, 1995.

[14] M.P.Singh, P.K.Yadav ,Harendra Kumar "A Tasks Allocation Model for Efficient Utilization Of Processor's Capacity in Heterogeneous Distributed Systems" presented in. 9th Conference of International Academy of Physical Sciences, held Agra, 2007

[15] Kumar, Avinash, Yadav P.K., & A. R. Gupta, "An exhaustive approach of distributed system based on cost assignment.", south Asian journal of mathematics and mathematical science, Vol.5 No 1, pp 29-44, 2006.

[16] Rotithor, H.G., “Taxonomy of Dynamic Task Scheduling in Distributed Computing Systems”, IEEE Proc. Computer Digit Tech., vol. 14, pp. 1-10, 1994.

[17] Raghavendra, C. S., and Hariri, S., “Reliability Optimization in the Design of Distributed Systems”, IEEE Transactions on Software Engineering, Vol. SE-11, pp. 1184-1193, 1985.

[18] Shatz, S. M., and Wang, Jai-Ping, “Models and Algorithms for Reliability-Oriented Task, Allocation in Redundant Distributed Computer Systems”, IEEE Transactions on Reliability, Vol. 38, No. 1, pp. 16-27, 1989.

[19] L. S. Srinath, “Reliability Engineering”, East-West Press, 2006.

[20] Bhatia K., Yadav P. K., & Sagar Gulati, “Performance Evaluation of distributed system through reliability analysis”, proceedings of International conference on Challenges and applications of mathematics in science and technology, NIT Rourkela, pp. 613 – 620, 2010.

[21] Yadav P.K., Kumar, Avinash, & M. P. Singh, “An algorithm for solving the unbalanced assignment problem”, “International journal of Mathematical Science”, Vol 12 (2), pp 447-461, 2004.

[22] Kumar, Avinash, Yadav P.K., & A. R. Gupta, “An exhaustive approach of distributed system based on cost assignment.”, south Asian journal of mathematics and mathematical science, Vol.5 No 1, pp 29-44, 2005.

[23] H. Ellis, Sahni S., & S. Rajasekaram, “Fundamentals of Computers Algorithm”, Galgotia Publications Pvt Ltd., 2005.