A Reliable Distributed Grid Scheduler for Independent Tasks

Kovvur Ram Mohan Rao[1], Ramachandram S[2], Vijaya Kumar Kadappa[3] and Govardhan A[4]

**[1] Department of Computer Science and Engineering, Vasavi College of Engineering, Osmania University
Hyderabad, Andhra Pradesh, 500031, India**

**[2] Department of Computer Science and Engineering, Osmania University
Hyderabad, Andhra Pradesh, India**

**[3] Department of Computer Applications, Vasavi College of Engineering, Osmania University
Hyderabad, Andhra Pradesh, 500031, India**

**[4] Department of Computer Science and Engineering, JNTUH
Karimnagar, Andhra Pradesh, India**

Abstract

Scheduling of jobs is one of the crucial tasks in grid environment. We consider non-preemptive scheduling of independent tasks in a computational grid. Recently, a general distributed scalable grid scheduler (GDS) was proposed, which prioritizes mission-critical tasks while maximizing the number of tasks meeting deadlines. However, the GDS scheduler did not consider the reliability factor, which may result in low successful schedule rates. In this paper, we propose a novel distributed grid scheduler which takes reliability factor (RDGS) into consideration with respect to the failure of grid nodes. The proposed scheduler invokes the tasks allocated to deficient grid nodes and maintains them in a queue. Further the queued tasks are rescheduled to the other nodes of the grid. It is observed that RDGS scheduler shows a significant improvement in terms of successfully scheduled tasks as compared to a variation of GDS without priority and deadlines (GDS-PD). The results of our exhaustive simulation experiments demonstrate the superiority of RDGS over the GDS-PD scheduler.

Keywords: *Grid Computing, Scheduling, Re-Scheduling, Distributed Scheduler, Reliability*

1. Introduction

Grid computing system is a collection of distributed computing resources available over a local or wide area network that appears to an end user or application as one large virtual computing system. The aim of grid system is to create virtual dynamic organizations through secure, coordinated resource-sharing among individuals, institutions, and resources. Grid computing is to provide an unlimited power, collaboration, and information access to everyone connected to grid [1] [2] [3].

Grid scheduling is a process of mapping grid tasks to grid resources over multiple administrative domains. The grid scheduler has four phases, which consists of resource discovery, resource selection, job selection and job execution. The responsibility of a scheduler is selecting resources and scheduling tasks in such a way that the user and application constraints are satisfied, in terms of overall execution time and cost of the resources utilized [5].

Quality-of-Service (QOS) support in resource management and scheduling has been the focus of many research studies in the computational studies. Ali Afzal et al. [6] bring out a scheduling algorithm that minimizes the cost of execution of workflows while ensuring that their associated QOS constraints are satisfied. Cesar A.F.De Rose et al. [8] present an explicit allocation strategy, in which an adaptor automatically fits grid requests to the resource in order to decrease the turn-around time of application. Mustafizar et al. [7] propose an approach for decentralized and cooperative workflow scheduling in a dynamic and distributed grid resource-sharing environment. The participants in the system such as the workflow brokers, resources and users who belong to multiple control domains, work together to enable a single cooperative resource sharing environment. Peijie Huang et al. [9] propose a method, which combines of an off-line static strategy using time series prediction and an on-line dynamic adjustment using reinforcement learning. The superiority of this scheduling algorithm is that it shows better load balancing of the whole hierarchical grid and achieves higher success rate of the grid service request. Ruay-Shiung Chang et al. [10] propose a balanced ant colony optimization (BACO) algorithm for job scheduling in the grid environment. The BACO algorithm balances the entire system load while trying to minimize the makespan of a given set of jobs. In contrast to these methods, Cong Liu et al. [11] developed a general distributed scalable grid scheduler (GDS) for independent tasks with different priorities and deadlines. GDS comprises of three phases: a multiple attribute ranking phase, a shuffling phase, and peer-to-peer dispatching phase.

However, the aforementioned methods do not consider the reliability factor, which is vital in the context of grid environment. There is no guarantee that the task will be scheduled successfully if the system is not reliable. In general, reliability is an ability of a system to perform and continue its functions in routine circumstances, as well as hostile or unexpected circumstances [13]. The reliability of a grid scheduling scheme depends upon the following three important factors:

- Task execution time: The time taken by the task to complete its execution.
- Communication time: The time consumed in communication in order to obtain the required resources from the various nodes of the grid.
- Rate of failure: The rate of failure of elements of grid computing system such as grid nodes, communication channels.

As given by Min Xie et al. [4], failure rate function $\lambda(t)$ is defined as the probability that a device of age t will fail in the small interval from t to $t+dt$ and is given by

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{R(t) - R(t+\Delta t)}{\Delta t R(t)}$$

The quantity *R (t)* represents the probability that system will be successfully operating without failure in the interval from time 0 to t.

In this work, we propose a distributed grid scheduler with reliability factor with respect to failure of grid nodes. The proposed scheduler also considers Communication to Computing Ratio (CCR) [11], which is useful to decide the appropriate grid site for scheduling tasks.

The rest of the paper is organized as follows. In section 2, we outline the grid model used in this work. Section 3 describes the proposed scheduling algorithm. Our experimental results are presented in section 4. Finally we conclude in section 5

2. Grid Model

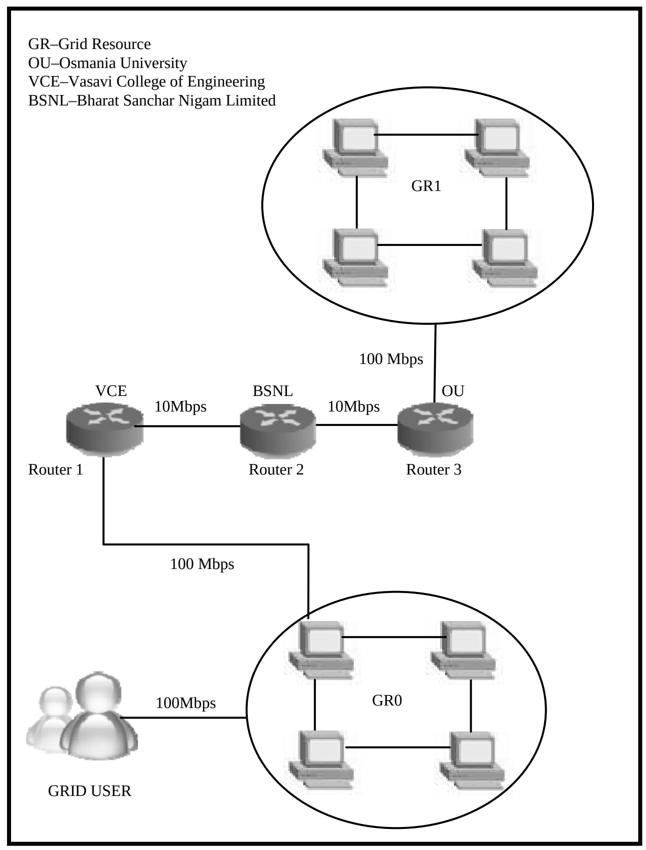


Fig. 1 Grid Model

We consider the grid model as shown in Fig.1, for our investigation. The grid model consists of geographically distributed sites which are interconnected through WAN. At each site, there is a Grid Resource (GR) consisting of several machines of different processing capabilities and a grid user have many tasks to be scheduled by the grid scheduler. The communication within the site (intra-site) is fast Ethernet (100Mbps); where as the communication across the sites (inter-site) is 10Mbps. Here we show a live model with two well-known educational institutions in India (VCE-Vasavi College of Engineering, OU-Osmania University, Hyderabad) and BSNL, an Internet Service Provider.

3. Reliable Distributed Grid Scheduler (RDGS)

In this section, we propose our scheduling algorithm (RDGS), which meets the following objectives:

- RDGS exploits reliability factor with respect to failure of nodes.
- RDGS is based on Communication to Computing Ratio (CCR), which is used to decide local or remote site for task scheduling.
- RDGS maximizes the total number of tasks completing execution.
- RDGS makes use of re-scheduling concept.

3.1 Notation

The following notation is used in this paper.

T_i : i th Task
Q : Task Queue
U : Queue of tasks assigned to a failed node
S_i : i th site with a number of machines
CCR_i : communication to computing ratio for task T_i
N_i : i th grid node

Now, we present our proposed RDGS algorithm.

3.2 Proposed RDGS Algorithm

The proposed algorithm (RDGS) consists of two phases: In the first phase all incoming tasks at each site are classified based on CCR value. Next in the second phase, scheduler assigns tasks to a specific resource on a site. Those tasks that are unable to execute due to machine failure are placed in a queue for resubmission.

First phase (classification of tasks based on CCR value): At each site, the users may submit a number of tasks with CCR values of 'low' and 'high'. The scheduler at each site puts all the incoming into task queue *Q*. If the task CCR value is high (communication intensive), these tasks are to be executed locally. If the task CCR value is low (computational intensive), these tasks are executed remotely.

Second phase (scheduling of tasks on a Grid Node with rescheduling): To schedule a task T_i on a site S_i, the scheduler selects a node randomly to balance the load. If the status of the selected node is 'working', the task T_i is executed on the selected node. If the status of the selected node is 'failed', the grid scheduler makes a provision for Task T_i to put up in a queue *U*. Further the tasks in the queue, *U* are simultaneously re-scheduled to other available resources.

We present the algorithm in a more formal way as given below. A user submits tasks to be executed, which are maintained in a Task Queue, *Q*. For each task T_i in Queue, *Q* we use RDGS () algorithm for scheduling.

Algorithm RDGS (Ti)
begin
1. If (CCR is 'low') then
1.1 T_i is assigned to Remote Grid Site, S_i
1.2 Call RDGS-Execute (T_i , S_i) for execution of T_i
2 Else If (CCR is 'high') then
2.1 T_i is assigned to Local Grid Site, S_j
2.2 Call RDGS-Execute (T_i , S_j) for execution
End

Algorithm RDGS-Execute (T_i, S_k)
begin
1. Select a node, N_i randomly at Grid Site, S_k
2. Check the status of the node, N_i.
3. If (Status of N_i is 'Failed')
3.1 Insert T_i in Queue U.
3.2 Re-schedule T_i by calling, once RDGS-Execute (T_i , S_k)
4. Else (Status of N_i is 'Working') then
4.1 T_i is scheduled to Node N_i
End

4. Experimental Results and Analysis

In this section, we present our experimental results and compare RDGS and GDS-PD (GDS without priority and deadlines) schedulers.

4.1 Experimental Setup

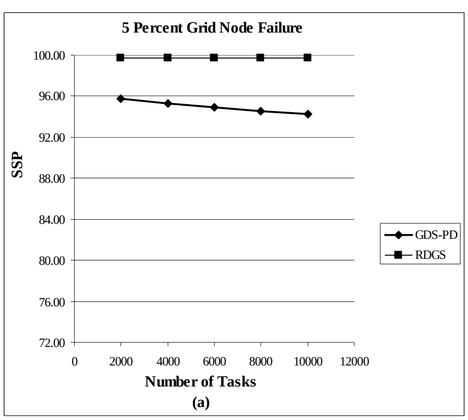
We used the following parameters in our experimental study: Task ID, Task length, Task file size, and Task output size, Communication to computational Ratio (CCR). We considered 'low', 'high' values for CCR.

We assumed the number grid of nodes as 10% of the tasks under consideration in our experiments. We varied number of failed nodes as 5%, 8%, 10%, 16%, 20% of nodes under consideration and obtained results. We computed Successful Schedule Percentage (SSP) using number of tasks successfully scheduled and total number of tasks.

We used GridSim [12] simulator for simulating Grid environment and the experimental results are shown in Figs. (2)-(3). We used Pentium-4 based system with CPU clock speed of 3GHz, 2.99 GB RAM running with Windows XP operating system.

4.2 Discussion of Results

Experiment 1 (Varying Number of tasks and fixed number of failed nodes): We plotted Figs. (2)(a)-(e) by computing Successful Schedule Percentage (SSP) with varying number of tasks. For each of these cases, we assumed fixed number of failed nodes (5%, 8%, 10%, 16%, and 20%) as shown in Figs. (2)(a)-(2)(e). From the Figs. 2(a)-2(e), we observed that RDGS scheduler shows improved SSP as compared to GDS-PD scheduler with varying number of tasks. With minimum node failure (i.e. 5%) RDGS shows higher SSP i.e. 99% (hence higher reliability) against 96% with GDS-PD method. With maximum node failure (i.e. 20%), RDGS shows significantly better SSP (95%) as compared GDS-PD method (75%). As the node failure rate increases RDGS is able to achieve much better SSP as compared to GDS-PD scheduler, thus showing high reliability. Also note that GDS-PD scheduler's reliability is worsened with increased node failure. In other words, RDGS is able to cope-up well with failed grid nodes, where as GDS-PD is lagging.



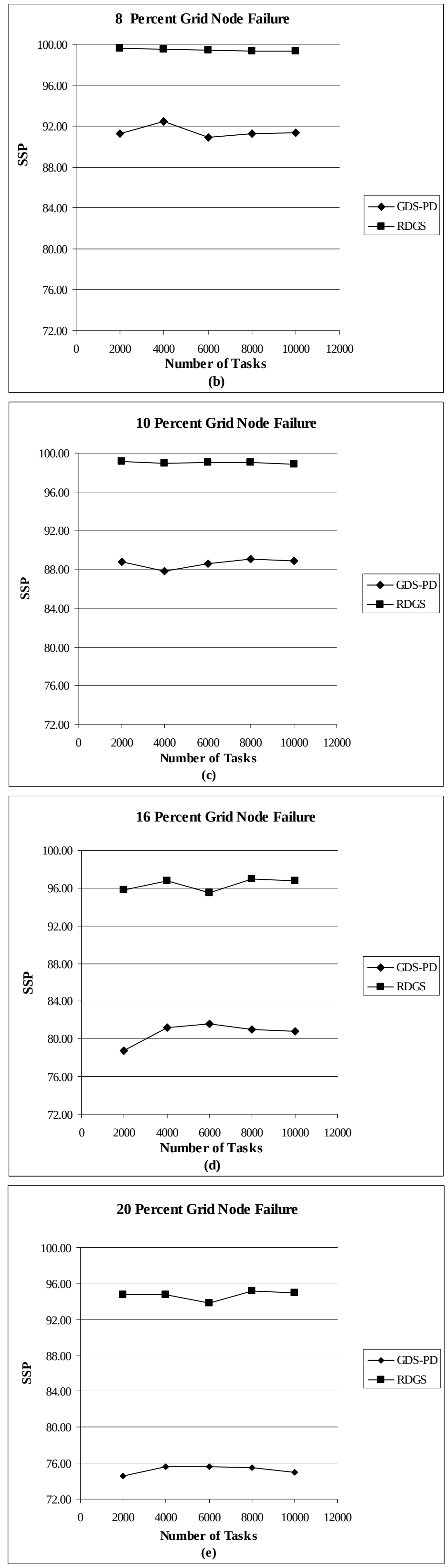
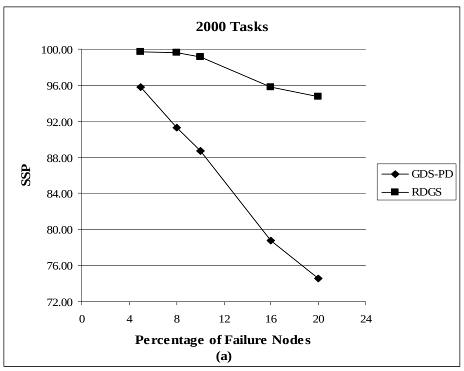
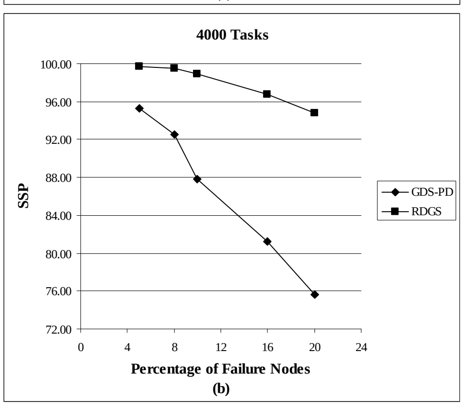
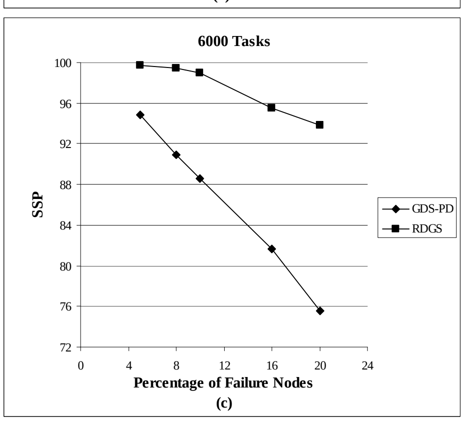


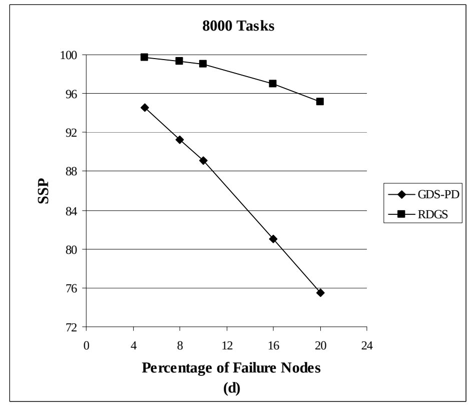
Fig. 2 Successful Schedule Percentage of RDGS & GDS–PD with varying number of tasks

Experiment 2 (Varying Percentage of Failure Nodes and Fixed number of tasks): We plotted Figs. (3)(a)-3(e) by computing SSP with varying percentage of failure rate and fixed number of tasks. For each these cases, we assumed fixed number of tasks as 2000, 4000, 6000, 8000, 10000 in Figs. (3)(a)-3(e) respectively. From the Figs. 3(a)-(e), we observed that RDGS scheduler shows improved and consistent SSP as compared to GDS-PD Scheduler. In other words, as the percentage of failed nodes increases (from 4% to 20%), fall in SSP of RDGS is not significant, where as GDS-PD shows wide variation in SSP. For RDGS, the variation in SSPs is 5% and the corresponding difference in SSPs for GDS-PD is 20%. In other words, RDGS is robust against failure in grid nodes.









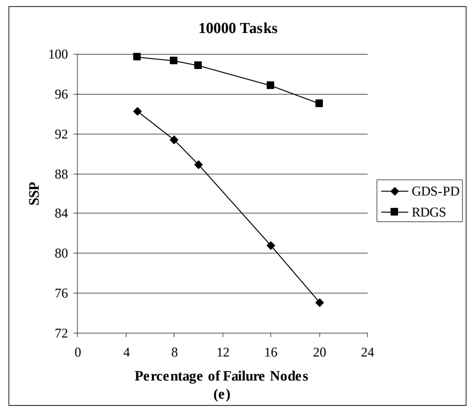


Fig. 3 Successful Schedule Percentage of RDGS and GDS-PD with varying number of failure nodes

Experiment 3 (Computational Requirements): We analyze here the computational requirements of RDGS and GDS-PD schedulers by varying number of tasks from 2000 to 10000 (in steps of 2000) with 10% fixed grid node failure rate. We computed additional computational requirements for RDGS to provide better reliability as compared to GDS-PD scheduler (Table.1). From the table (last column) it is evident that RDGS provides better reliability (better SSP) at the cost of an insignificant additional computational time (3.3% to 4.2%).

Table.1 Computational time requirements of RDGS & GDS-PD Schedulers

No.of tasks	No.of Nodes	SSP		Comp. Time		Addl. comp Time (2)-(1)
		GDS -PD	RDGS	GDS-PD (1)	RDGS (2)	
2000	200	88.75	99.15	14387	14862	3.3 %
4000	400	87.78	98.93	28817	29875	3.6%
6000	600	88.55	99.02	42992	44825	4.2%
8000	800	89.11	99.06	57386	59807	4.2%
10000	1000	88.90	98.88	71745	74718	4.1%

5. Conclusion

We proposed a reliable distributed scheduler, which promised an improved successful schedule rate in spite of grid node failures. The proposed scheduler shows superior successful schedule percentage at the cost of insignificant additional computational requirements. The proposed method is very useful in grid environment because there is a possibility for any node to get failed due to various factors. In future we improve the method by extending it different categories of tasks by taking parameters such as deadlines, priority, etc.

References

[1] L.J. zhang, J.Y.Chung, and Q. Zhou, Developing grid computing applications, part 1: Introduction of a grid architecture and toolkit for building grid solutions, IBM Corporation New York, October 2002.

[2] T. Hawk, Remarks at, Grid computing planet conference and expo, June 2002.

[3] D. Minoli, A Networking approach to grid computing. Hoboken, New Jersey: Wiley Interscience, A John Wiley & Sons, Inc.,, 2005.

[4] Min Xie, Yuan Shun Dai, and Kim Leng Poh, Computing System Reliability Models and Analysis. Kluwer Academic/Plenum Publishers, New York, 2004.

[5] Maozhen Li, Mark Baker, The Grid Core Technologies, A John Wiley & Sons, Inc.,, 2005.

[6] Ali Afzal, A. Stephen McGough, John Darlington, Capacity planning and scheduling in grid computing environments, Future Generation Computer Systems 24(2008),404-414.

[7] Mustafizur Rahman, Rajiv Ranjan, Raj Kumar Buyya, Cooperative and decentralized workflow scheduling in global grids, Future Generation Computer Systems (2009).

[8] Cesar A.F.De Rose, Tiago Ferreto, Rodrigo N. Calheirros, Walfredo Cirne, Lauro B.Costa, Daniel Fireman, Allocation strategies for utilization of space-shared resources in Bag of Tasks grids, Future Generation Computer Systems 24(2008),331-341.

[9] Peijie Huang, Hong Peng, Piyuan Lin , Xuezhen Li , Static strategy and dynamic Adjustment: An efficient method for grids task scheduling, Future Generation Computer Systems 25(2009),884-892.

[10] Ruay-Shiung Chang, Jih-Sheng Chang, Po-Sheng Lin, An Ant algorithm for balanced job scheduling in grids, Future Generation Computer Systems 25(2009),20-27.

[11] Cong Liu, Sanjeev Baskiyar, A general distributed scalable grid scheduler for independent tasks, J. Parallel Distrib. Comput. 69(2009) 307-314.

[12] Buyya R K, Murshed M, Anthony S, Marcos D de A, Agustin C, GridSim Tool kit 4.1: A Grid simulation toolkit for resource modeling and application scheduling for parallel and distributed computing (2007).

[13] Wikipedia (visited Feb 2009) Reliability, [online], http://en.wikipedia.org/wiki/Reliability.

Kovvur Ram Mohan Rao is a part-time Ph.D student at Department of CSE, JNTUH, Hyderabad, India and also full time member of faculty at Department of Computer Science and Engineering, Vasavi College of Engineering, Hyderabad, India. His research interests are in Computer Networks, Distributed Systems and Grid Computing. He received his Bachelors degree in engineering from Nagpur University, Nagpur, India in 1993 and Masters Degree in Computer Science and engineering from Osmania University, India in 2000. He has 17 years of teaching experience. Mr. Rao is an annual member of IEEE and life member Computer Society of India. Mr. Rao published his research papers in several International conferences.

Dr.S. Ramachandram received Ph.D. degree in Computer Science and Engineering from Osmania University in 2005, M.Tech. from Osmania University in 1985 and B.E. in from Osmania University in 1983. He is working as a Professor of Osmania University, Hyderabad. He has published around 45 papers in various national and international Journals/conferences. Dr. Chandram is an annual member of IEEE, senior member of CSI, Fellow IETE and member of ACM. His research of interest includes Computer Networks, Distributed systems, Mobile computing and grid computing.

Dr.Vijaya Kumar Kadappa is presently working as Associate professor in the Dept. of Computer Applications, Vasavi College of Engineering, Hyderabad, India. Dr. Kumar obtained his Ph.D. from University of Hyderabad, India. He received his Bachelors degree in Science from Sri Krishna Devaraya University, Anantapur, India in 1995 and Masters Degree in Computer Applications from University of Mysore, India in 1998. He has 11 of teaching experience. His research interests are Pattern Recognition, Data Mining and Clustering. Dr. Kumar is a life member of IEEE, Indian Unit of International Association of Pattern Recognition, ISTE and Computer Society of India. Dr. Kumar has published his research papers in several International journals and conferences.

Dr.A.Govardhan received Ph.D. degree in Computer Science and Engineering from Jawaharlal Nehru Technological University in 2003, M.Tech. from Jawaharlal Nehru University in 1994 and B.E. in from Osmania University in 1992. He is working as a Principal of Jawaharlal Nehru Technological University, Jagitial. He has published around 50 papers in various national and international Journals/conferences. He is also a member in various professional bodies including CSI, ISTE, IAENG, FSF and WASET. His areas of interest include Databases, Data Warehousing & Mining, Information Retrieval, Computer Networks, Image Processing and Object Oriented Technologies.