ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

Performance Analysis of SCADY Grid for High Performance Parallel Applications

Rakesh Bhatnagar[1], Dr. Jayesh Patel[2]

Asst. Prof., MCA Dept, S. K. Patel Institute of Management & Computer Studies, Gandhinagar, Gujarat, India[1]

Associate Prof., Acharya Motibhai Patel Institute of Computer Studies, Ganpat University, Kherva, Gujarat, India [2]

ABSTRACT: Considering the need of a better, GUI based and effective high performance computing solution, we created the Grid called SCADY (Scalable & Dynamic). Lot of grids are available with different specifications like multithreading, dynamic resource allocation, scheduling etc. But we found that in all the available grids, whenever a resource fails, a new resource is allocated through dynamic allocation but the job restarts. In Scady, we implemented dynamic resource allocation through caching. To obtain the best performance, both grid system and application-specific information can be considered in implementing high performance problems that meets the desired Quality of Service (QoS) requirements. Authors have implemented grid scenarios to solve parallel application problems like Monte Carlo Simulation etc. Authors also implemented the same in Alchemi.net, Unicore and Globus grid environments and have obtained favorable experiment results for Scady. Scenarios were created in such a manner that dynamic resource allocation can be tested.

This paper gives the performance as well as the comparative analysis of Scady with other grid environments like Alchemi.net, Unicore, and Globus in executing high performance parallel applications.

KEYWORDS: Scady, Monte Carlo simulation, Performance Analysis, Scenarios

I. INTRODUCTION

Scady [4] is a grid framework designed for serving high performance computations. It provides a window based GUI for execution. Scady is based on Alchemi.Net [9, 10] grid framework. Alchemi.Net grid framework is gaining popularity in high performance computations as it is window based. Alchemi .Net is an open source software framework that uses the power of computing of machines present in a network and helps to develop and run applications on the grid environment.

Scady grid framework [4, 5] uses the same Manager-Executor model as used by Alchemi.Net framework. But there is no owner node in Scady. There can be more than one Manager Node in Scady. These Manager nodes can perform task by distributing it to the executor nodes.

Dynamic resource allocation [3] is used based on the CPU usage and load using which high performance applications can be catered. According to the CPU usage, nodes are selected dynamically and tasks are delegated to them. Even if an executor node or manager node fails, the task is allocated dynamically to the next available node. To save time and effort, when the task is allocated to the new node, execution starts from the point where it stopped [2].

In this paper, authors have given the comparative analysis of Scady, Alchemi.net, Unicore [14, 15] and Globus [16, 17]. The organization of the Paper is as follows – Section 2 describes the literature review; Section 3 presents the experiment taken to implement Monte Carlo simulation; Section 4 describes the implementation and results; Section 5 contains the result analysis; Section 6 describes the conclusion, and Last Section contains the references.

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

II. RELATED WORK

We studied various Grid systems, their security mechanisms and task management. The most popular system GLOBUS [17, 18] which uses three security mechanisms - GSI (Grid Security Infrastructure)[24], MyProxy & GSI-OpenSSH [24]. GSI helps in providing APIs and tools for certificate management and authentication. Globus is very popular among non-GUI grid environments. The Globus Toolkit is an open source toolkit for grid computing developed and provided by the Globus Alliance that follows the standards from the Open Grid Forum (OGF), Open Grid Services Architecture - OGSA, Open Grid Services Infrastructure - OGSI, Web Services Resource Framework - WSRF, Job Submission Description Language - JSDL, Distributed Resource Management Application API - DRMAA, SOAP and Grid Security Infrastructure - GSI etc. As discussed in [18, 19], the Grid framework is used to execute high performance parallel applications.

UNICORE [14, 15, 23] is open source grid middleware system that follows the standards from the Open Grid Forum (OGF), W3C, OASIS, and IETF, in particular the Web Services Resource Framework (WS-RF 1.2), JSDL etc. It is simple and easy to install and configure. It uses the SOA architecture and RESTful Api and is implemented in Java and Python. We have used the UNICORE Testgrid. This Testgrid offers a very easy access to a full featured UNICORE demo site for new and interested users. The Testgrid allows users to create, submit and monitor grid jobs and workflows. Unicore is used to execute high performance parallel applications.

Alchemi [9, 10, 11, 13] is an open source software framework that allows you to painlessly aggregate the computing power of networked machines into a virtual supercomputer (desktop grid) and to develop applications to run on the grid. It has been designed with the primary goal of being easy to use without sacrificing power and flexibility. Alchemi includes:

- The runtime machinery (Windows executables) to construct computational grids.
- A .NET API and tools to develop .NET grid applications and grid-enable legacy applications.

This GUI grid is used to execute high performance parallel applications.

III. EXPERIMENT

We have used Monte Carlo method [21, 22] to calculate the value of pi. Monte Carlo (MC) methods are stochastic techniques which mean that they use the concept of random numbers with statistical probability to investigate problems [20, 21]. Monte Carlo methods use statistical simulation by using collection of random number generation to perform the simulation. To get accurate result, more iterations and their average is taken.

Consider a circle embedded within a square, like a dart board, as shown in Fig-I.

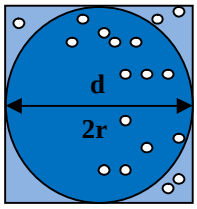

FIG - I: EMBEDDED CIRCLE IN A SQUARE

Here the width of the circle i.e. diameter d is equal to the width of the square. The square therefore has length and width equal to the diameter of the circle i.e. twice the radius, 2r.

Consider the area of the circle

$$\text{Area(c)} = \text{pi} \times r^2$$

And the area of the surrounding square

$$\text{Area(s)} = l \times b = d \times d = d^2 = (2r)^2 = 4r^2$$

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

If we compare the area of the circle with the area of the square, we can form a ratio -

$$\frac{\text{Area(c)}}{\text{Area(s)}} = \frac{pi \times r^2}{4r^2} \quad \text{which will reduce to} \quad \frac{\text{Area(c)}}{\text{Area(s)}} = \frac{\text{pi}}{4}$$

And rearrange to solve for pi:

$$\text{Pi} = 4 \times \frac{\text{Area(c)}}{\text{Area(s)}}$$

We can conclude that pi is equal to four times the ratio of the area of an embedded circle to the area of its outer square. To estimate pi we will use the above equation.

Now, assume that darts are thrown at random locations on Fig-I. After substantial trials, the percentage of random darts striking the circle will be proportional to the area of the circle as a percentage of the total area of square. In other words, if we throw twenty darts and sixteen of them go within the circle then the area of circle can be calculated as 16/20 or 80% of the area available.

If we throw twenty darts and ten go within the circle, then we would estimate the area of the circle to be 10/20 or 50% of the total area.

Using this assumption, we can determine through Monte Carlo simulation a rough value for the ratio of the area of the circle to the total area:

$$\frac{\text{Area(c)}}{\text{Area(s)}} = \frac{\text{number of darts landing inside circle}}{\text{total number of darts thrown}} = \text{hit \%}$$

and from this ratio, a value for pi:

$$\text{pi} = 4 \times \frac{\text{Area(c)}}{\text{Area(s)}} = 4 \times \text{hit \%}$$

To get a decent and accurate value of pi, it takes a very large number of throws, say over 100000. So, we distribute the number of trials the user wants to attempt among the available nodes in the grid.

IV. IMPLEMENTATION & RESULT

To implement Monte Carlo method as discussed above, we created scenarios to test. The parameters taken were –

- Number of Iterations
- Number of nodes (processors/executors) required
- Time taken to produce the result in mili seconds
- Dynamic node allocation if a node fails during the execution.

When the manager is started, it asks for how many executors i.e. processors or nodes to be used. We created different scenarios, by using different values of the above parameters. And we selected the Monte Carlo method to calculate the value of pi. This method is based on random and large number of iterations and trials and takes the average of them. The working is shown in Fig-II.

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

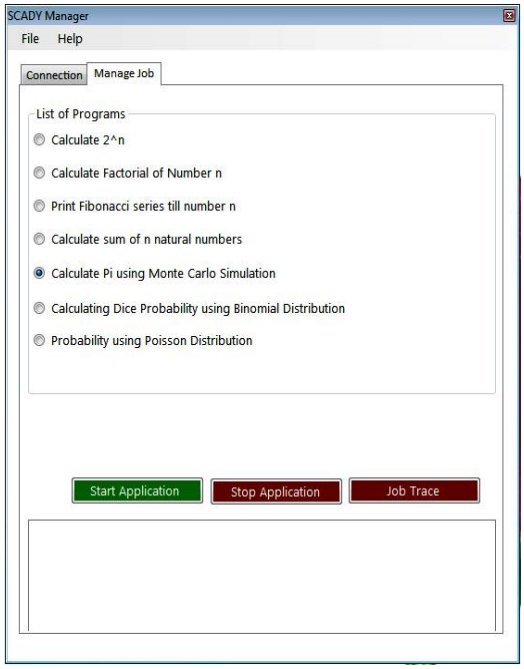

FIG - II: SCADY MANAGER (APPLICATION MANAGER)

When the application is started, we have to give input parameters – number of executors and number of iterations as in Fig-III. These are required for implementing the concepts of multithreading and Monte Carlo Simulation.

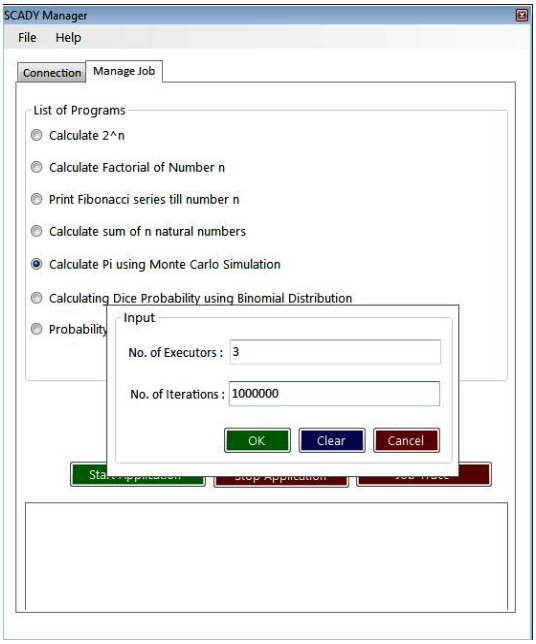

FIG - III: INPUT PARAMETERS (SCENARIO 1)

We used different scenarios by altering these parameters. When we gave number of executors as 3 and number of iterations as 1000000, we calculated the value of pi as shown in Fig - IV.

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

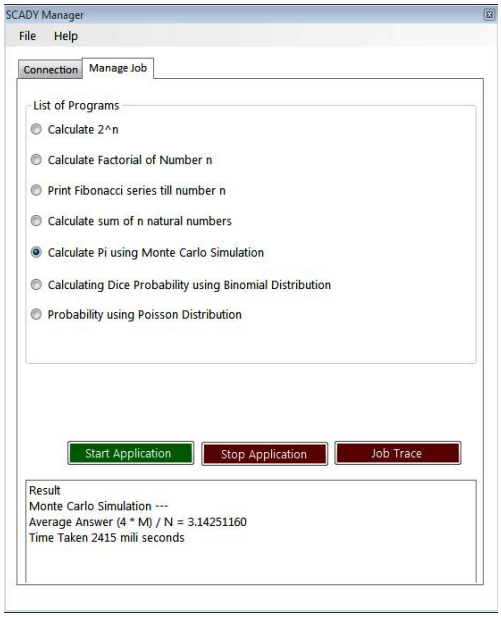

FIG - IV: RESULT (SCENARIO 1)

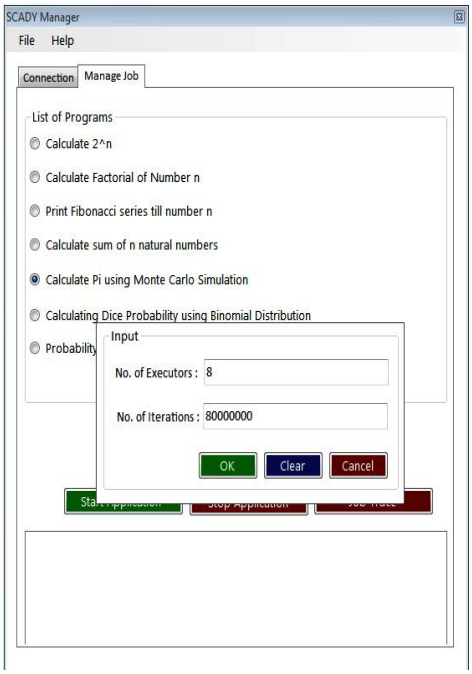

FIG - V: INPUT PARAMETERS (SCENARIO 2)

Then we used a different scenario and increased the number of executors as 8 and number of iterations as 80000000 and calculated the value of pi as shown in Fig – V and Fig - VI.

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

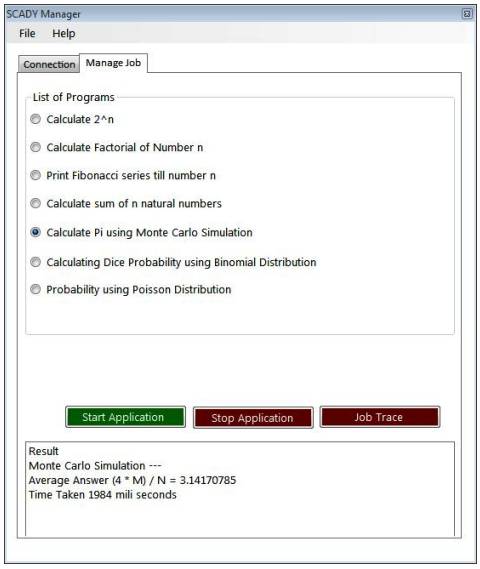

FIG - VI: RESULT (SCENARIO 2)

In the similar manner, other scenarios were implemented and tested. Initially, we implemented the scenarios in Scady and Alchemi .Net. The results are given in Table - I.

TABLE - I: EXECUTION TIME OF PI CALCULATION

Number of Executors	Number of Iterations	Time (in mili seconds)	
		Scady Toolkit	Alchemi Toolkit
3	1000000	2551	2509
8	80000000	1984	2016
8	100000000	1852	1979
8	800000000	1619	1846
20	800000000	1852	1979
50	800000000	1619	1846

Then we also implemented the scenarios in Unicore as well as Globus grid environments. The comparative results are given in Table – II.

Then we also implemented the scenarios in Unicore as well as Globus grid environments. The comparative results are given in Table – II.

TABLE - II: COMPARISON OF EXECUTION TIMES USING OTHER GRID ENVIRONMENTS

Number of Executors /Processors	Number of Iterations	Time (in Secs)			
		Unicore Toolkit	Globus Toolkit	Alchemi Toolkit	Scady Toolkit
3	1000000	3016	2515	2509	2551
8	80000000	2566	1998	2016	1984
8	100000000	2025	1945	1979	1852

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

Special scenarios were also tested for dynamic resource allocation. Deliberately a node that was executing the task was shutdown and the application was tested. Manager dynamically reallocates the task to another executor. The cache write-through method [2] is used in which at the same time the data is written into the cache block and the corresponding main memory location. The data cached can be retrieved very fast on demand, while it is ensured that nothing will get lost if a crash, power failure, or other system disruption occurs as the same data is in main memory. Write through method is the most preferred method for storing the data for the applications where data is critical and data loss cannot be tolerated. When the task is allocated to the new node, result from cache is read and the execution starts from the point where it stopped.

TABLE - III: EXECUTION TIME WHEN NODE FAILS

Number of Executors	Number of Iterations	Time (in mili seconds)	
		Scady Toolkit	Alchemi Toolkit
3	1000000	3205	4376
8	80000000	2882	3780
8	800000000	2729	3225
20	800000000	2170	2636
50	800000000	1724	2108

V. RESULT ANALYSIS

The results of the experiment of calculating value of pi by using different scenarios and the reallocation of task to another node in case of node failure are shown in the comparison chart of Fig – VII & Fig – VIII.

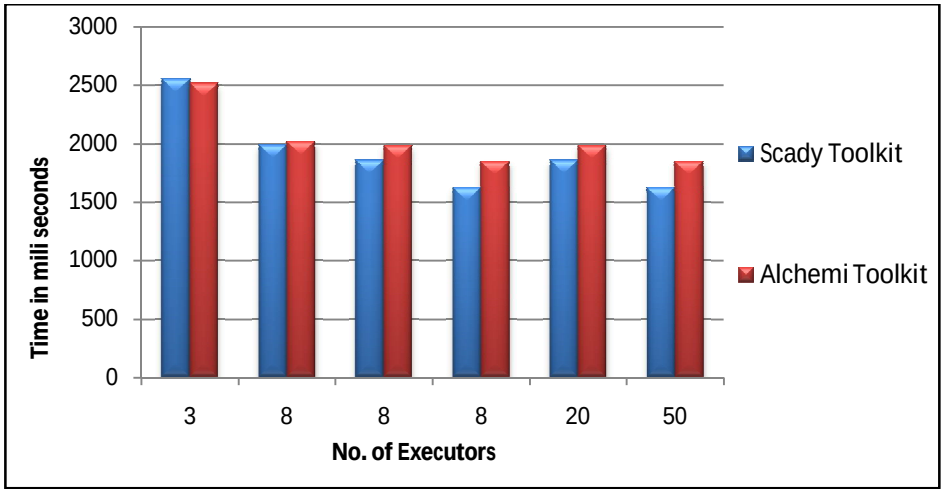

FIG - VII: COMPARISON CHART OF SCADY & ALCHEMI

Above comparison chart gives the comparison of Scady with Alchemi .Net toolkit. It indicates that as the numbers of processors/executors are increased, the calculation time is decreases. Also the time taken by Scady is less as compared to other grid environments.

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

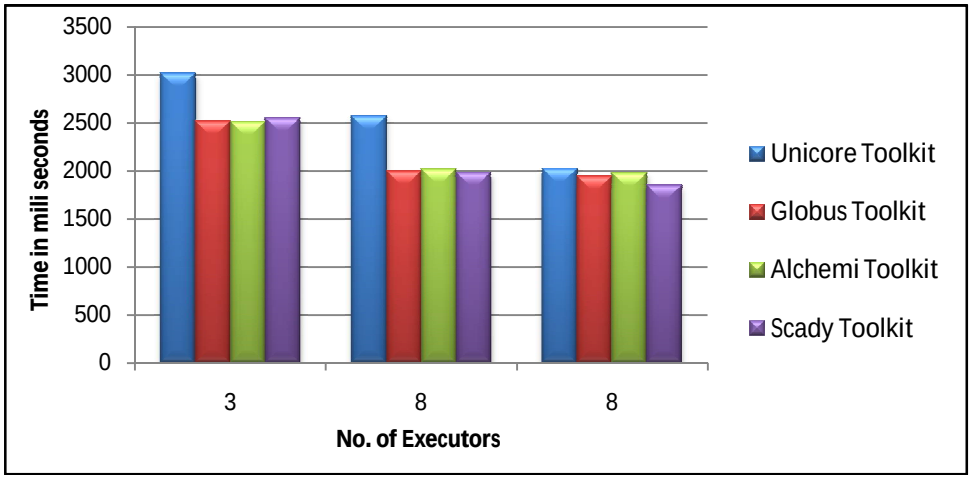

FIG - VIII: COMPARISON CHART OF SCADY, ALCHEMI, UNICORE & GLOBUS

Above comparison chart gives comparison of Scady with Unicore, Globus, and Alchemi .Net grids. It indicates that as the numbers of processors/executors are increased, the calculation time is decreases. Also the time taken by Scady is less as compared to other grid environments. Scady, Globus and Alchemi .net gives almost the same performance.

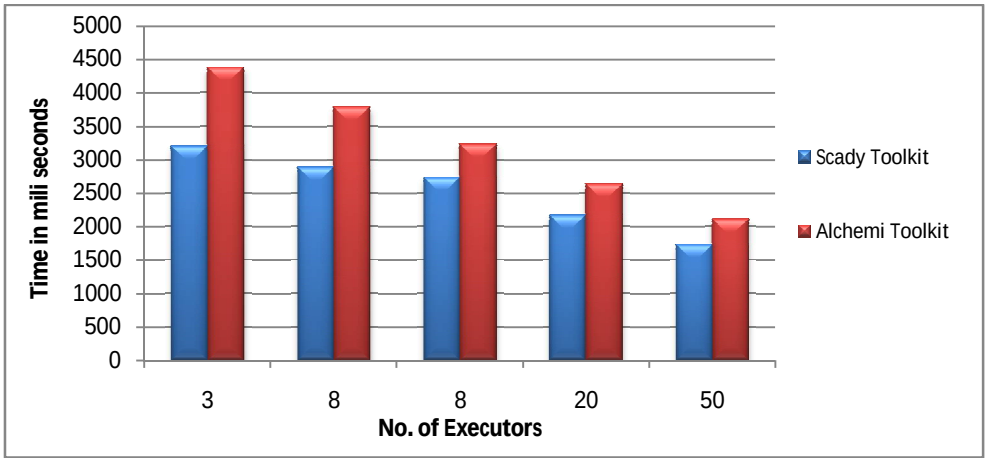

FIG - IX: COMPARISON CHART OF SCADY, ALCHEMI, UNICORE & GLOBUS

This chart indicates the time taken by Scady and Alchemi .Net grids when a node fails and the job is reallocated to a new node. It can be seen that Scady gives better performance as compared to Alchemi .Net as the task reallocated does not start from the beginning but is resumed from where it stopped.

VI. CONCLUSION

In this paper, we tested the performance of Scady for high performance parallel applications and compared it with Alchemi.Net Grid Framework, Unicore and Globus. We found that even though these frameworks provide good performance, Scady gives better results in node failure as the intermediate results are written into cache and as well as the main memory. And the contents of cache are transferred to manager nodes at specific intervals.

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

REFERENCES

1. Rakesh Bhatnagar, Dr. Jayesh Patel, "Security Model for Scady Grid Toolkit – Analysis and Implementation", International Journal of Innovative Research in Computer & Communication Engineering, ISSN 2320–9801, Vol. 3, Issue 12, December 2015, Pages 362-365
2. Rakesh Bhatnagar, Dr. Jayesh Patel, Nirav Vasoya, Snehal Rindani, "Implementation of Caching Algorithm in Scady Grid Framework", 6th IEEE International Conference on Computing Communications and Networking Technologies ICCCNT - 2015, July 13-15, 2015, Texas, USA.
3. Rakesh Bhatnagar, Dr. Jayesh Patel, Nirav Vasoya, "Dynamic Resource Allocation in SCADY Grid Toolkit", IEEE International Conference on Computing, Communication and Automation (ICCCA-2015), 15-16 May, 2015.
4. Rakesh Bhatnagar, Dr. Jayesh Patel, "Scady: A Scalable & Dynamic Toolkit for Enhanced Performance in Grid Computing", IEEE International Conference on Pervasive Computing ICPC - 2015, Jan 8-10, 2015, IEEE Xplore, DOI:10.1109/PERVASIVE.2015.7087085 Publication Year: 2015
5. Rakesh Bhatnagar, Dr. Jayesh Patel, "API Specification for Small Grid Middleware - Scady", IEEE Xplore, India Conference (INDICON), 2014 Annual IEEE, DOI:10.1109/INDICON.2014.7030545 Publication Year: 2015
6. Rakesh Bhatnagar, Dr. Jayesh Patel, "An Empirical study of Secur- ity issues in Grid Middleware", International Journal of Emerging Technology & Advanced Engineering, ISSN 2250–2459**,** Volume 4 Issue 1, Jan 2014, Pages 470-474
7. Rakesh Bhatnagar, Dr. Jayesh Patel, "Performance Analysis of a Grid Monitoring System - Ganglia", International Journal of Emerging Technology & Advanced Engineering, ISSN 2250–2459**,** Volume 3 Issue 8, Aug 2013, Pages 362-365
8. Rakesh Bhatnagar, Dr. Jayesh Patel, "Performance Analysis of a Grid Monitoring System - Autopilot", Journal of Sci-Tech Research (KSV – JSTR) - Volume 4, Issue 2, July - Dec 2013, ISSN 0974 – 9780, Pages 15-20.
9. Rajkumar Buyya, Akshay Luther, Rajiv Ranjan, and Srikumar Venugopal, "Alchemi: A .NET-based Enterprise Grid Computing System", Internet Computing, ICOMP 2005 www.cloudbus.org/papers/alchemi_icomp05.pdf
10. Rajkumar Buyya, Akshay Luther, Rajiv Ranjan, and Srikumar Venugopal, "Alchemi: A .NET-based Grid Computing Framework and its integration into Global Grids", www.cloudbus.org/papers/alchemi.pdf
11. Manisha Bhardwaj, Sarbjeet Singh & Makhan Singh, "Implementation of Single Sign-on and Delegation mechanism in Alchemi.Net based Grid Computing Framework", International Journal of Information Technology and Knowledge Management January-June 2011, Volume 4, No. 1, pp. 289-292
12. Sabir Ismail, Abu Fazal Md Shumon, Md Ruhul Amin, "Distributed Memory Caching for the Fail Safe Computation to Improve the Grid Performance", Proceedings of 13th International Conference on Computer and Information Technology (ICCIT 2010), 23-25 December, 2010, Dhaka, Bangladesh
13. "Alchemi, .NET Grid Computing Framework", Accessed April 13, 2015 from http://www.cloudbus.org/~alchemi/
14. Dietmar W. Erwin, David F. Snelling, "UNICORE: A Grid Computing Environment", Euro-Par 2001 Parallel Processing, 7th International Euro-Par Conference Manchester, UK, August 28–31, 2001 Proceedings, Volume 2150, pp 825-834, Online ISBN 978-3-540-44681-1
15. A. Streit, D. Erwin, Th. Lippert, D. Mallmann, R. Menday, M. Rambadt, M. Riedel, M. Romberg, B. Schuller, Ph. Wieder**, "**Unicore — From project results to production grids", Advances in Parallel Computing, Volume 14, 2005, Pages 357–376\
16. F. Cappello, E. Caron, M. Dayde, F. Desprez, Y. Jegou, P. Primet, E. Jeannot, S. Lanteri, J. Leduc, N. Melab, G. Mornet, R. Namyst, B. Quetier, O. Richard, "Grid'5000: A Large Scale and Highly Reconfigurable Grid Experimental Testbed", Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing, Pages 99-106
17. Jiageng Li, David Cordes, "A scalable authorization approach for the Globus grid system", Future Generation Computer Systems, Elsevier, Volume 21, Issue 2, 1 February 2005, Pages 291–301
18. Rakesh Bhatnagar, Aditya Patel, Vipul Prajapati, Dr. J.G. Pandya, "Agent Based Architecture for Scalable Grid Resource Management and Scheduling", International Journal of Computer Applications in Engineering, Technology and Sciences (IJ-CA-ETS), Vol. 2, Issue 1, (ISSN 0974-3596), Oct-Mar, 2009, Paper Number – 72, Page(s): 395 – 401
19. Rakesh Bhatnagar, Digvijay Rathod, Vipul Prajapati, "Grid Framework – An Optimum Foundation for Service Oriented Architecture", International Journal of Computer Applications in Engineering, Technology and Sciences (IJ-CA-ETS), Vol. 2, Issue 1, (ISSN 0974-3596), Oct-Mar, 2009, Paper Number – 53, Page(s): 299 – 303.
20. Cameron B. Browne, Edward Powley, Daniel Whitehouse, Simon M. Lucas, "A Survey of Monte Carlo Tree Search Methods", IEEE Transactions on Computational Intelligence, Volume 4, Issue 1, Page(s): 1 – 43
21. G Fishman, "Monte Carlo: concepts, algorithms, and applications", books.google.com, accessed on January 19, 2016
22. Rick Wicklin, "Monte Carlo estimates of pi and an important statistical lesson", http://www.statsblogs.com/2016/03/14/monte-carlo-estimates-of-pi-and-an-important-statistical-lesson/ accessed on March 14, 2016
23. Unicore Client user manual retrieved Jan 10, 2014, from: *www.unicore.eu/documentation/manuals/unicore6/RichClient-.3.1.pdf*
24. GSI-OpenSSH Globus, https://dev.globus.org/wiki/GSI-OpenSSH

BIOGRAPHY

Rakesh Bhatnagar is Asst. Prof. in the MCA Department, S. K. Patel College of Management & Computer Studies, Kadi Sarva Vidyalaya University, Gandhinagar, Gujarat, India. He has 13 years of teaching experience. He received M.Tech.(Computer Science) degree in 2006 and M.C.A. degree in 2002. Presently pursuing his research (Ph.D.), his research area is in Grid Architecture, Middleware and Services.

Dr. Jayesh Patel is Associate Prof. in the MCA Department, Acharya Motibhai Patel Institute of Computer Studies, Ganpat University, Kherva, Gujarat, India. He has done MCA and received Doctorate degree in 2008. His research interest is in Grid Computing and ERP systems.