International Journal of Computer Theory and Engineering, Vol. 1, No. 5, December, 2009
1793-8201

Scavenging Idle CPU Cycles for Creation of Inexpensive Supercomputing Power

Madhuri D Bhavsar and Shrikant N Pradhan

Abstract—- **Grid computing is often regarded as a means for creating inexpensive "super-computing." With grid computing organizations can optimize computing and data resources, pool such resources to support massive, compute intensive loads. This paper aims at demonstrating experiences gained during aggregation of large computational power acquired from idle workstations connected on the campus. The cycle harvesting is attained by integrating and configuring a suite of grid and middleware software technologies anchored by the Globus toolkit, further optimized to extract the large computational capacity in the form of MIPS and GFLOPS. NirmaGrid produced is coordination of available computational efforts by Scavenging of idle CPU cycles, which allows the sharing of a computational task among multiple processors.**

Index Terms— **Campus grid, Computational grid, Grid Computing, Supercomputing**

I. Introduction

Grid computing enables organizations to transparently integrate, streamline and share dispersed, heterogeneous pool of hosts, servers, workstations and networks into one synergistic system in order to deliver agreed upon service at specific level of application efficiency and processing performance [3]. Scavenging grid, which is one of the popular computing paradigm commonly used to locate and exploit machine cycles on idle servers and desktop computers for use in resource, compute intensive task. One of the Grid promises is to make it possible to share and effectively use distributed resources on an unprecedented scale. Specifically, this includes harnessing the unused capacity of idle desktop PCs. Much research has been done to Grid enable idle resources, yet no widely accepted system to effectively scavenge idle cycles in window, in particular idle Windows cycles, has been found. This paper demonstrates the methodology implemented on the campus for scavenging idle CPU clock cycles to generate large number of computational power. It discovers how to effectively exploit the massive amount of idle CPU cycles from the desktop machined connected on the campus for grid computing, demanded by e-science or compute intensive applications.

Manuscript received June 19,2009.
Madhuri D Bhavsar, is with Computer Science and Engineering Department, Institute of Technology, Nirma University of Science and Technology, Ahemadabad, (+91-02717-241911-262; e-mail: madhuri.bhavsar@nirmauni.ac.in)
Dr Shrikant N Pradhan, is with Computer Science and Engineering Department, Institute of Technology, Nirma University of Science and Technology, Ahemadabad(e-mail: snpradhan@nirmauni.ac.in)

II. Related Work

TeraGrid[2] is a well-known collaboration between leading organizations including Argonne National laboratory. Boinc [4] is a software platform that allows many different distributed computing projects to utilize idle computing resources. [1] Demonstrates the functioning of newly developed batch scheduling algorithm which concentrates on utilization of idle CPU cycles in distributed computing systems. [6] Provides an approach adopted for usage of virtual machine to provide sandbox that completely allows cycle scavenging with security and protection. In Seti@home idle cycles are used for dedicated scientific applications developed. [7] Presents the design and implementation of Dodo, an efficient user-level system for harvesting idle memory in off-the-shelf clusters of workstations.

In this paper we depicts the formation of NirmaGrid using desktop machine in linux clustering with windows spreads in two different building blocks interconnected with 100 mbps fibre optic cable , provides computational grid around 165 GFlops along with 545218 MIPS. Extensibility allows to have seamless integration of machines in an enterprise.

III. Motivations To Computaional Grid

Grids are geographically distributed platforms for computations accessible to their users via a single interface [5]. Computational grid represents the natural extensions of large parallel and distributed systems and exits to provide high performance computing, offering properties like – *large* in terms of potentially available resources, distributed, dynamic, heterogeneous, and multiorganisational boundaries. It is obvious from these properties that providing a single abstraction hiding complexities is a substantial challenge. Still grid users and researchers accept these challenges because they need to go beyond a single parallel system with cutting financial burdens. It has been identified that most of the time desktop, personal computers remains unutilized. Aggregating and scavenging these idle cycles plays a key role for resolving complexities of scientific applications. E-science is scientific investigation performed during global collaborations between scientists and their resources, and the computing infrastructure that enables deployment of jobs comfortably.

IV. Configuring Steps for Generation of Computational Grid

NirmaGrid is configured and integrated using suite of grid and middleware software technologies like PBS and Condor anchored by the Globus toolkit with a design objective towards an open, heterogeneous and extensible architecture that can be further expanded to promote interoperability amongst different academic organizations. Fig 1 shows an architecture of NirmaGrid, where currently only 162 machines are integrated for experimentations. To achieve interoperability large number of workstations will be integrated in near future.

V. Helpful Hints

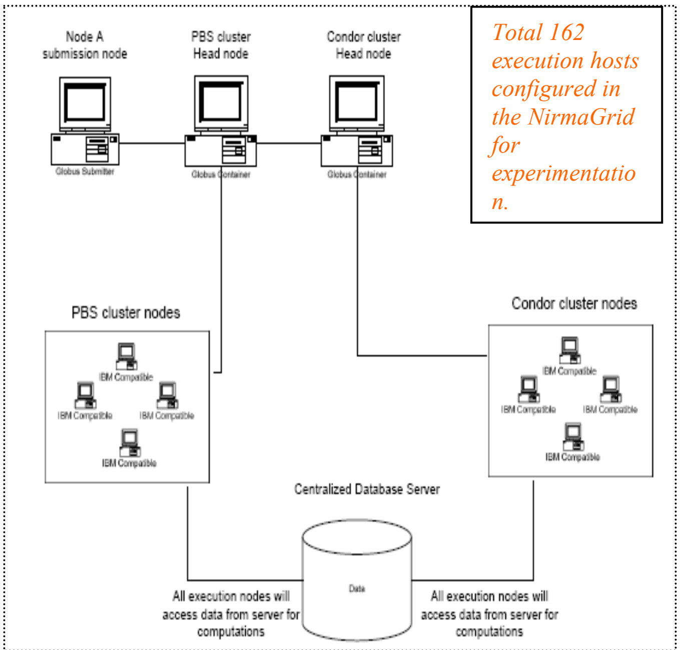
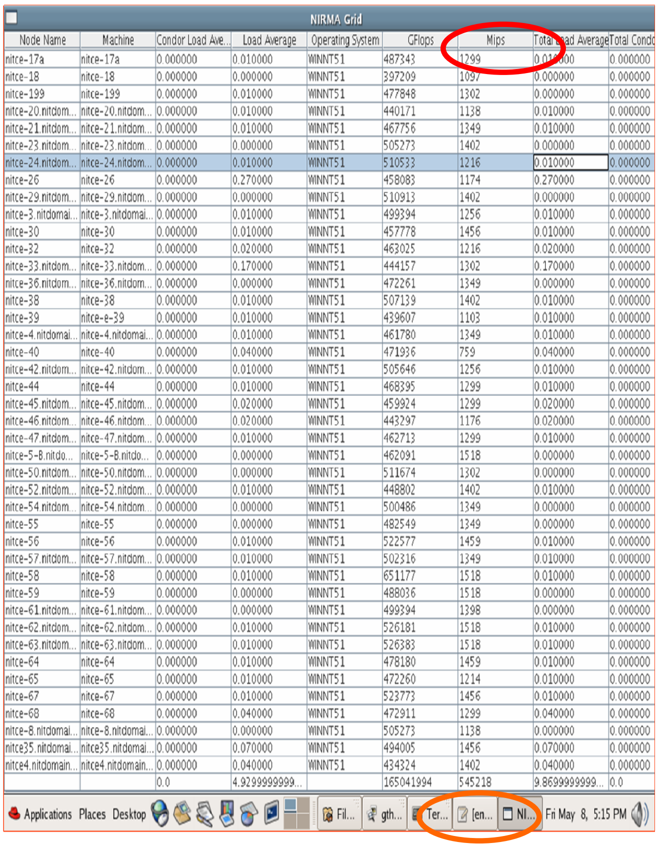


Fig. 1 Architecture of NirmaGrid

A. Steps To Be Configured

1. As per the installation and implementation steps given for Globus, node A, B and C are configured providing basic grid infrastructure.
2. PBS and Condor clusters are formed with pooling other resources together, integrated with central database server.

3. To add execution hosts in the Condor pool, in addition to configuration of condor, hostname and ipaddress of central manger are configured in Hosts file of execution hosts.
4. Integrate these resources with Globus in the grid. Configure Globus to submit job to a heterogeneous Condor pool
5. After complete integration, get the list of workstations in grid.
6. Fetch various parameters like flops, load, mips etc.
7. Calculate aggregated GFlops, Average load, Mips available in the grid.
8. Obtain aggregate power in the computational grid and ready to accept the job[fig 2] .Marked fields shows available GFLOPS with other details of the host. Fig 3 shows the dynamic generation of resources repository connected in the grid. This information periodically updates the status of resources available in the grid. This figure is captured in the peak hours where very few resources were idle.



Node Name	Machine	Condor Load Ave...	Load Average	Operating System	GFlops	Mips	Total Load Average	Total Cond
nitce-17a	nitce-17a	0.000000	0.010000	WINNT51	487343	1299	0.010000	0.000000
nitce-18	nitce-18	0.000000	0.000000	WINNT51	397209	1097	0.000000	0.000000
nitce-199	nitce-199	0.000000	0.010000	WINNT51	477848	1302	0.000000	0.000000
nitce-20.nitdom...	nitce-20.nitdom...	0.000000	0.010000	WINNT51	440171	1138	0.010000	0.000000
nitce-21.nitdom...	nitce-21.nitdom...	0.000000	0.010000	WINNT51	467756	1349	0.010000	0.000000
nitce-23.nitdom...	nitce-23.nitdom...	0.000000	0.000000	WINNT51	505273	1402	0.000000	0.000000
nitce-24.nitdom...	nitce-24.nitdom...	0.000000	0.010000	WINNT51	510533	1216	0.010000	0.000000
nitce-26	nitce-26	0.000000	0.270000	WINNT51	458083	1174	0.270000	0.000000
nitce-29.nitdom...	nitce-29.nitdom...	0.000000	0.000000	WINNT51	510913	1402	0.000000	0.000000
nitce-3.nitdomai...	nitce-3.nitdomai...	0.000000	0.010000	WINNT51	499394	1256	0.010000	0.000000
nitce-30	nitce-30	0.000000	0.010000	WINNT51	457778	1456	0.010000	0.000000
nitce-32	nitce-32	0.000000	0.020000	WINNT51	463025	1216	0.020000	0.000000
nitce-33.nitdom...	nitce-33.nitdom...	0.000000	0.170000	WINNT51	444157	1302	0.170000	0.000000
nitce-36.nitdom...	nitce-36.nitdom...	0.000000	0.000000	WINNT51	472261	1349	0.000000	0.000000
nitce-38	nitce-38	0.000000	0.010000	WINNT51	507139	1402	0.010000	0.000000
nitce-39	nitce-e-39	0.000000	0.010000	WINNT51	439607	1103	0.010000	0.000000
nitce-4.nitdomai...	nitce-4.nitdomai...	0.000000	0.010000	WINNT51	461780	1349	0.010000	0.000000
nitce-40	nitce-40	0.000000	0.040000	WINNT51	471936	759	0.040000	0.000000
nitce-42.nitdom...	nitce-42.nitdom...	0.000000	0.010000	WINNT51	505646	1256	0.010000	0.000000
nitce-44	nitce-44	0.000000	0.010000	WINNT51	468395	1299	0.010000	0.000000
nitce-45.nitdom...	nitce-45.nitdom...	0.000000	0.020000	WINNT51	459924	1299	0.020000	0.000000
nitce-46.nitdom...	nitce-46.nitdom...	0.000000	0.020000	WINNT51	443297	1176	0.020000	0.000000
nitce-47.nitdom...	nitce-47.nitdom...	0.000000	0.010000	WINNT51	462713	1299	0.010000	0.000000
nitce-5-B.nitdo...	nitce-5-B.nitdo...	0.000000	0.000000	WINNT51	462091	1518	0.000000	0.000000
nitce-50.nitdom...	nitce-50.nitdom...	0.000000	0.000000	WINNT51	511674	1302	0.000000	0.000000
nitce-52.nitdom...	nitce-52.nitdom...	0.000000	0.010000	WINNT51	448802	1402	0.010000	0.000000
nitce-54.nitdom...	nitce-54.nitdom...	0.000000	0.000000	WINNT51	500486	1349	0.000000	0.000000
nitce-55	nitce-55	0.000000	0.000000	WINNT51	482549	1349	0.000000	0.000000
nitce-56	nitce-56	0.000000	0.010000	WINNT51	522577	1459	0.010000	0.000000
nitce-57.nitdom...	nitce-57.nitdom...	0.000000	0.010000	WINNT51	502316	1349	0.010000	0.000000
nitce-58	nitce-58	0.000000	0.010000	WINNT51	651177	1518	0.010000	0.000000
nitce-59	nitce-59	0.000000	0.000000	WINNT51	488036	1518	0.000000	0.000000
nitce-61.nitdom...	nitce-61.nitdom...	0.000000	0.000000	WINNT51	499394	1398	0.000000	0.000000
nitce-62.nitdom...	nitce-62.nitdom...	0.000000	0.010000	WINNT51	526181	1518	0.010000	0.000000
nitce-63.nitdom...	nitce-63.nitdom...	0.000000	0.010000	WINNT51	526383	1518	0.010000	0.000000
nitce-64	nitce-64	0.000000	0.010000	WINNT51	478180	1459	0.010000	0.000000
nitce-65	nitce-65	0.000000	0.010000	WINNT51	472260	1214	0.010000	0.000000
nitce-67	nitce-67	0.000000	0.010000	WINNT51	523773	1456	0.010000	0.000000
nitce-68	nitce-68	0.000000	0.040000	WINNT51	472911	1299	0.040000	0.000000
nitce-8.nitdomai...	nitce-8.nitdomai...	0.000000	0.000000	WINNT51	505273	1138	0.000000	0.000000
nitce35.nitdomai...	nitce35.nitdomai...	0.000000	0.070000	WINNT51	494005	1456	0.070000	0.000000
nitce4.nitdomain...	nitce4.nitdomain...	0.000000	0.040000	WINNT51	434324	1402	0.040000	0.000000
		0.0	4.9299999999...		165041994	545218	9.8699999999...	0.0

Fig. 2 Parameters aggregated in NirmaGrid

This setup configured is dynamic and provides a repository of available execution hosts in the grid environment; however priority is assigned to the owner of the host. Fig 4 shows the screenshot during job submission carried out on the computational grid.

V. Conclusion

Grids can be built ranging from just few processors to large groups of processors, either homogeneous or heterogeneous, organized as a hierarchy that spans a continent. Our attempt was to exploit an unused computational cycles from the campus grid and avail these for solving the complexities of scientific application. This paper provides an optimized approach for generation of inexpensive supercomputing power. Extensibility of this NirmaGrid allows integrating more number of machines to increase the computational power.

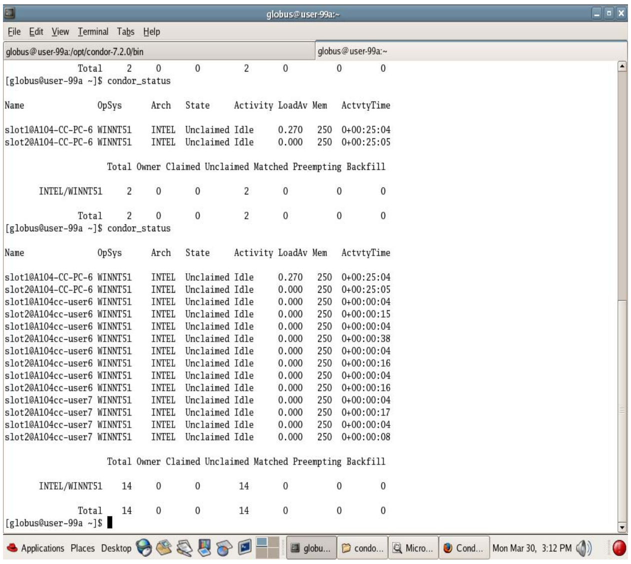


Fig 3 Snapshot of resource pool

ACKNOWLEDGMENT

Authors are thankful to Sneha Mehta, Hitesh , Harikrishna Chiatanya and Manan for their contribution in forming the NirmaGrid

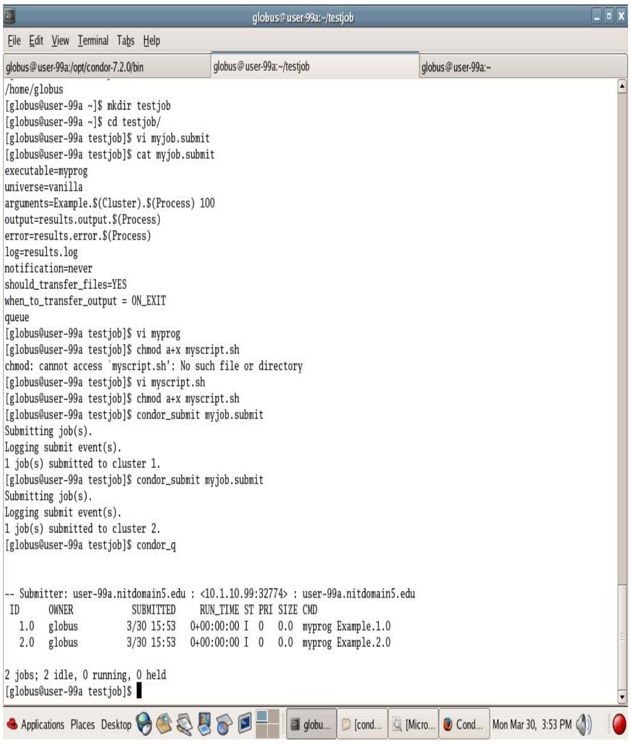


Fig 4 job status

REFERENCES

[1] Andreas Wespi, Ernst Rothauser, "Utilizing idle CPU cycles in a distributed computing system", Proceedings of IEEE NAECON 1997, Volume 1, July 1997

[2] R. Andersen and B. Vinter. Harvesting idle windows cpu cycles for grid computing. Proceedings of the 2006 International Conference on Grid Computing & Applications (GCA 2006), pages 121–126, Las Vegas, Nevada, USA, June 2006. CSREA Press

[3] Daniel Minoli, " A Networking Approach to Grid Computing,A John Wiley & Sons Inc. Publication.

[4] http://boinc.berkely.edu

[5] D B Skillicorn," Motivating computational grids", 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid ,CCGRid '02.21-24 May 2002.

[6] Charlie Carlett, "The philosophy of TeraGrid: building an open, extensible, distributed terascale facility", Proceedings of 2nd IEEE/ACM international symposium on cluster computing and grid '02.

[7] Samir Koussih Anurag Acharya Sanjeev Setia , "Dodo: A User-level system for exploiting iIdle memory in Workstation Clusters", The Eighth IEEE international symposium on High Performance Distributed Computing, Aug 1999 Proceedings