The Setup and Installation of the Dixon Hall Supercomputing Pool

Brian Campbell Senior Elizabeth City State University
Unquiea B. Wade, Junior Elizabeth City State University
Bryce Carmichael, Senior Elizabeth City State University

Abstract- **The international polar year was designed to study and better understand the current state of the climatic changes to the world's ice sheets. For the last few decades, there have been automated weather stations and satellites in geo-synchronous orbit that created data sets. Today, numerous amounts of data are unexplored due to insufficient funding and the scarcity of resources. For this reason, the polar grid concept was proposed to delegate the analysis of the existing data sets.**

The goal of the Elizabeth City State University's Polar Grid Team was to construct a model network to serve as a base for a super computing pool. The super computing pool will be constructed on the university's campus and linked to the overall polar grid system. Numerous Software and protocols were researched that are currently in use at other institutions around the nation. From the possible protocols, the condor software was chosen. Condor was created and developed at the University of Wisconsin because of easier usage and its willingness for expansion.

An eighteen node computing pool was constructed and tested within Dixon Hall's second floor lab using Condor. This pool was comprised of seventeen desk-tops running on a Windows NT platform, with the pool's mater housed in Lane hall acting as a Linux based server.

I. INTRODUCTION

The Polar Grid Team is comprised of, Brian Campbell, Unquiea Wade and Bryce Carmichael. Under the supervision of Dr. Eric Akers, work on the project began in the fall of 2007, as part of the overall research effort at Elizabeth City State University. The task of the Polar Grid Team was to configure the first of numerous computers to be added to the ECSU computing pool.

The first task performed was the collection of data pertaining to high throughput networking, including software options and the analysis of networks currently in operation. This step was the most time consuming due to the teams varying levels of experience in the field of networking and computer science.

The second phase included the installation and configuration of the virtual network linking the computers in the Dixon hall laboratory to the master server in lane hall this in itself was a problem due to name and access protocol. The final stage of the project is still currently being performed which is an orientation to the operation and monitoring of the condor system. In this stage we sought to familiarize ourselves with the protocol for the submission of a task to the system and to understand the various outputs and diagnostic tools available through the condor software.

A. Super Computing

In the beginning of our research it was important to understand what is meant by the term supercomputing. The term was originally coined in reference to computer systems which were capable of running at speeds and efficiency greater than what was readily available to the public, by several orders of magnitude. For the majority of its short history this was done by building systems to meet the requirement out of extreme high-end materials and technologies.

B. Parallel vs. Distributed

As supercomputing progressed, the limits of what was possible using these expensive material was quickly exceeded, limited by space, temperature and cost. The short coming of the single system approach was quickly acknowledged, leading to the development of

protocol for extending the processing power of a network without the need to purchase the expensive high end components. This was accomplished by linking several independent processors on to a single logic sharing network.

Parallel processing as this was called links all of the networks processor to work as a single unit, taking on a single task with the efficiency afforded by shear number. This method provided the increases to work load capability that was desired, but developed several draw backs as the needs of the computing network changed. It was troublesome to reconfigure, when adding new nodes the entire system was required to change. Since the processor all acted as a single unit if one was busy all were busy, this lead to an abundance of idle time within the system on smaller jobs. Second as the networks began to expand it became necessary to increase the amount of communication between the individual processors and the master server. Communication traffic began to affect the efficiency of the system significantly after 50 to 60 nodes were added. This has been compensated for by increasing the complexity of communication and networking algorithms, but the underlying problem persisted.

In response to these deficiencies a second form of multi-node processing arose to compensate. In a method termed distributed processing a single job is broken in to several parts with each part assigned to an individual node (processor) the processors the run separately communicating only when task were completed or when data had to be transferred effectively minimizing communication across the network. This innovation allowed single networks to run multiple jobs simultaneously effectively eliminating idle time, and could be applied to treat entire parallel sets as a single node allowing the easy expansion of both types of computing.

II. METHODOLOGIES

A. Condor

Both of these types of computing networks make use of cheap, easily acquired components and software to amplify the power of the entire network. We choose to implement a distributed computing model as the base for the project because of its ease expansion and its non-centralized design. With the model chosen we began research in to the different software and protocols that could be use to establish such a system. Our searches lead us to several organizations through out America each applying the super computing model. Of the networks studied the most highly recommended was a software program developed by the University of Wisconsin known as Condor. Condor had the benefits of being able to configure for both parallel and distributed systems. It also has the ability to operate on multiple OS platforms simultaneously; Although Condor comes with several administrative tools such as the checkpoint program, which can only be accessed on a Linux or UNIX platform.

The software accessibility was a big draw. In order to further the development of the science of super computing, the University of Wisconsin has given open source rights to the condor source code. They have allowed easy download from the project condor home page at http://www.cs.wisc.edu/condor/. Along with the program downloads this page also offers an installation guide, user manual and example programs that can be run to test configuration. This site became one of our primary sources in the installation and trouble shooting of the network.

The first step in the installation of the system was the establishment of the systems primary master the software was loaded onto the subject device and installed using OS based installation software. The software was initially installed on a windows machine known as CERSER-1 located on site within the Dixon hall computing lab. But was relocated due to a naming redundancy and the benefits of running the master out of a Linux platform, such as access to the standard universe which contains condors proprietary code and administrative tools. It was also necessary to isolate the master from the system so that user traffic could not

interfere with its administrative operation.

Due to the need for reliable communication the master node had to be installed on a machine with a static IP address so that the identity of the master could be insured to be constant in case of blackouts or the unforeseen. To further communication some changes had to be made to the configuration of all the machines altering the read and write privileges to include the qualifier 10.*.*.*, this allowed files to be accessed remotely and for any requested out puts to be copied to the task's source node.

The remaining seventeen nodes within the pool were configured on Windows NT platforms this was for no other reason then it was the current OS platform for these machines. The same additions were made to the read and write access line of the configuration allowing communication. The IP address of the master was set to 10.40.20.37 as host name for the master node. After these initial changes were made within the first machines attached to the group the others came into the system without impedance.

In a condor network task can be submitted from any node to the master. This is done through an ssh shell environment by entering the condor_submit command followed by the name of the executable file. This ease of submission was in the forefront of reasons for selecting the condor platform for our project. Jobs could be written in any executable format for submission to what is known as the vanilla universe. In this universe the majority of condor special features are nonfunctional to simplify the administration of the job.

Within the standard condor universe jobs can be submitted to the master using a Linux/ Unix based converter to access condor proprietary languages code allowing access to more in depth control over the execution of the jobs. Using this specialized code a skilled programmer can designate the type of platform to run each task or the number of nodes to be used, and put into place “checkpoints” at which the process will be halted in order to allow an examination of the output at different stages within the processing.

The output from any task must be specified within the code of the executable and will be written to the console where the job was originally submitted within the condor bin by default or to the specified location. However Condor offer two file types which are automatically placed within the task files directory known as file.error and file.log, relatively self explanatory file.error houses any error reports that may occur during the processing and file.log will record time reliant data throughout the duration of the task, including time of submission time, task time, cpu usage time and downtime. Such information can also be accessed during the processing using the condor_q and condor_status commands within the bin directory of condor.

The condor_q is a list of job submission statuses, this command will show the number of task awaiting processing the task currently being run in conjunction with information of the CPU currently running and will show duration, start and stop times for tasks performed recently within the system. This file is useful in tracing the progress of a task through the system and has aided us significantly in troubleshooting the network. Though it has a draw back, unless emptied on a regular basis this file can become sizable rather quickly making it difficult to locate the proper information in a timely manner.

The condor status command another which we made extensive use of in the troubleshooting the network, outputs a file containing a list of host names and IP addresses associated with members of the pool. It also gives information as to the current availability of node on the system. The availability is shown through a group of one word qualifiers in the forth column of the display. The possible qualifiers are; claimed, meaning the nodes is currently active in the processing of a job. The Unclaimed qualifier signifies that the node is free to take on a job, but may not meet the requirements. Matched indicates that the node meets the qualifications necessary to perform the submitted task. There have been two busy qualifiers that have come up through

submission of the task, unavailable with means that for what ever reason the computer has declined or is otherwise unable to perform the task and owner, which indicates that there is someone working as a user on the node in question.

III. RESULTS

To date we have added only 17 nodes to the condor pool excluding the setup of the master in lane hall making the total 18. This has opened the gate for any console on the ECSU grid to be added using the above configuration. No attempt has been made to attach the pool to the polar grid system but it is expected that it will be accomplished with relative ease.

Simple counting and outputting programs have been submitted to the pool to test the network, but the full capability of the pool has not been tested. During the trouble shooting of the Dixon hall system it was found that there were two naming redundancies there were with two consoles answering to Cerser-1 and another pair answering to cerser-12. The redundancies were easily fixed by changing the nodes host name, but it should be noted that nodes within the pool can only hold record of one IP address per host name. Duplication of a host name is not allowed on the network; the master will choose randomly between the nodes sharing the name and ignore the others. This caused a profound amount of trouble since CERSER-1 was selected to be the original master for the pool; because of the redundancy no communication was possible.

IV. CONCLUSION

The future expansion of the project should be easily accomplished and with the right oversight and planning can progress without limit. To preserve simplicity the number of nodes on any single pool should be limited and multiple pools should be constructed around local masters to be connected to the main system as a parallel node. This will allow a simple repetitive naming convention and the partitioning will allow the isolation of a local pool if it should become necessary.

A down side to the distribution of the network across the campuses labs is that the computing functions will be subject to interrupt by the day to day user traffic. It may be wise to continue to isolate a cpu as a dedicated master for each pool saving it from the random interruptions. It is my suggestion that the construction of at least one dedicated pool should be considered for use by priority tasks, and that the configuration is reexamined after the completion of the pool as it is now a task must be restarted at any interruptions and this could lead to a lot of wasted time.

V. FUTURE WORK

The continuation of this project would immediately install the software through the Ubuntu portal in Dixon to accentuate the windows pool. Then the Lane hall computer lab should be added to the pool we suggest that these also be introduced through a UNIX platform. The condor software is available for the MAC, but has many of the same limitations as the windows machines. After the assimilation of the lane hall lab the network can be spread to other areas of the campus at the cost of the dedicated master. We see no reason not to continue the use of the schools existing network for communication.

Attachment to the polar grid system may have to wait as the system grows but groups should be encourage to submit task to our pool to foster the relations and to familiarize the CERSER teams with condor operations. Workshops and training should be conducted to allow students to make full use of the system. We have an opportunity to draw some real attention to ECSU and we should not let his simple progressive technology to pass us by.

VI. REFERENCES

1. Andrew S. Tanenbaum, Maarten Van Steen (2002): Distributed Systems Principles and Paradigms. New Jersey: Prentice- Hall Inc.

2. Amza C., A.L. Cox, S. Dwarkadas, P. Keleher, R. Rajamony H. Lu, W. Yu, and

W.Zwaenepoel. ThreadMarks: Shared memory computing on networks of workstations, to appear in IEEE Computer,(draft copy): www.cs.rice.edu/willy/TreadMarks/papers.html
3. A.J. van der Steen, An evaluation of some Beowulf clusters, Technical Report WFI-00-07, Utrecht University, Dept. of Computational Physics, December 2000. (Also available through www.euroben.nl, directory reports/.)
4. A.J. van der Steen, Overview of recent supercomputers high-end servers, June 2005, www.euroben.nl, directory reports/.
5. University of Wisconsin,(2007) condor user manual and installation guide retrieved form
http://www.cs.wisc.edu/condor/manual/v7.0/
on the date of 10/31/07
6. University of California, (2007 Overview of Boinc, downloaded from
http://boinc.berkeley.edu/trac/wiki/BoincIntro
on the date of 01/21/08
7.