INTELLIGENT JOB ROUTER

by

Kartik Vedalaveni

A THESIS

Presented to the Faculty of

The Graduate College at the University Of Nebraska

In Partial Fulfilment of Requirements

For the Degree of Master Of Science

Major: Computer Science

Under the Supervision of Dr. David Swanson

Lincoln, Nebraska

May, 2013

INTELLIGENT JOB ROUTER

Kartik Vedalaveni, M.S

University Of Nebraska, 2013

Adviser: Dr. David Swanson

Schedulers come with plethora of features and options for customization that fulfills myriad goals of clusters and data centers . Often there is a need to extend these schedulers to solve situations arising from new use cases. One such case is when any resource like I/O, RAM or Network is throttled and the degradation that occurs as a result of it . With increase in number of entities concurrently using the resource, there is a need to monitor and schedule concurrent and unscrupulous access to any given resource to prevent degradation.These issues that we encounter in real life at Holland Computing Center (CITE) are the basis and motivation for tackling this problem and develop an adaptive approach for scheduling that is aware of multi-resource throttling, load-balancing across multiple sites and degradation with a goal to run clusters at high efficiency and share resources fluidly.

DEDICATION

Dedicated to

ACKNOWLEDGMENTS

Thanks

Contents

Chapter 1

Introduction

Grid Computing refers to collectively and efficiently utilizing the resources that exist across multiple clusters. The resources come in the form of hardware and software that allows us to submit jobs, run the jobs and monitor the jobs on the grid. Universities usually have multiple clusters across their campus and these are owned by different departments but stand united under the banner of university. Derek Weitzel has described about campus grids in much detail in his thesis[CITE] .Campus grids can be thought of as mini grids in which jobs are spanned across multiple clusters based on the need of the user and available computing infrastructure across multiple clusters within the computing platforms across university.

Modern schedulers used in clusters provide innumerable features, the problem of cluster performance degradation that occurs due to improper load balancing is a problem that hasn't been addressed. The problem of performance degradation when many jobs are scheduled on single system either based on processor equivalence or based on number of processor slots. Some of these schedulers like maui (CITE) are smart enough to take into account contention of other resources like RAM but ultimately convert the 2D vector values of CPU and RAM into a single scalar value which equals to hard-coding the value

or presenting these resources in some kind of ratio which makes us question effectiveness of such scheduling mechanism.

Existing schedulers depend on the availability of resource and frequent polling of it to determine the slots which of scheduling. It should be noted that state of the art schedulers like maui/torque, slurm, condor take into account only CPU as a resource and the resources like RAM, I/O or Network are either ignored or converted into a scalar values which isn't an effective way of tackling the multiple resource scheduling problem as their true resource load measure is converted to processor equivalent or ignored that. We take a turnaround time approach to measure degradation, we define a resource to be degraded if and when we submit jobs that results in increasing the turnaround time by say 25% we conclude that the system is degraded. It must be noted that there isn't explicit measurement of individual resources of RAM, I/O, Network and CPU, there are no resource managers keeping track of these resource. The Co-Scheduler is driven by the fact that turnaround time increases when concurrent jobs accessing resources reaches a threshold value and this is the basis for scheduling.

One aspect to Co-Scheduler is degradation detection and management by adapting to the throttling of resource another aspect is to efficiently distribute jobs across multiple sites. The Co-Scheduler adapts to degradation and efficiently distributes the jobs and load-balances across multiple sites submitting more jobs to a site with lesser turnaround time and also ensuring to submit lesser jobs to the sites with larger turnaround time thus efficiently load-balancing across multiple sites on a grid.

Grid environment is a heterogeneous environment, some sites might have bare minimal OS installation, some with additional packages. Care must be taken as a Co-Scheduler, it does not consume additional packages that are not available on another cluster. To implement such a Co-Scheduler we limit ourselves to condor based clusters and utilize libcondorapi. The result is a threaded Co-Scheduler that submits to multiple sites

concurrently and is aware of resource

1.1 Co-Scheduler

DHTC environment

Chapter 2

Background

2.1 High-Throughput Computing

High Throughput Computing, HTC is defined as a computing environment that that delivers large amounts of computational power over a long period of time. The important factor being over a long period of time which differentiates HTC from HPC which focuses on getting large amount of work done in small amount of time. The workloads that run on condor system doesn't have an objective of how fast the job can be completed but how many times can the job be run in the next few months.In another definition of HTC, European Grid Infrastructure defines HTC as a computing paradigm that focuses on the efficient execution of large number of loosely coupled tasks.

2.2 HTCondor

HTCondor is a distributed system developed by HTCondor team at the University of Wisconsin-Madison. It provides High-Throughput Computing environment to sites that foster research computing and enables sites to share computing resources when computers

are idle at a given site. HTCondor system includes a batch queuing system for a pool of computers mainly used for compute-intensive jobs, HTCondor runs on both UNIX and windows based workstations that are all connected by a network. HTCondor serves the research community by providing them a queuing mechanism, scheduling policy, priority scheme and resource classification. Although there are other batch schedulers out there for dedicated machines. The power of condor comes from the fact that the amount of compute power represented by sum total of all the non-dedicated desktop workstations sitting on people's desks is sometimes far greater than the compute power of dedicated central resource. There are many unique tools and capabilities in HTCondor which make utilizing resources from non-dedicated systems effective. These capabilities include process checkpoint and migration, remote system calls and ClassAds.

2.3 Open Science Grid

Open Science Grid(OSG), provides service and support for resource providers and scientific institutions using a distributed fabric of high throughout computational services. OSG was created to facilitate data analysis from the Large Hadron Collider . OSG doesn't own resources but provides software and services to users and enables opportunistic usage and sharing of resources among resource providers. The main goal of OSG is to advance science through open distributed computing. The OSG provides multi-disciplinary partnership to federate local, regional, community and national cyber-infrastructures to meet the needs of research and academic communities at all scales.

OSG provides resources and directions to Virtual Organizations(VO's) for the purposes of LHC experiments and HTC in general.

Building a OSG site requires listing background and careful planning. The major

components of a OSG site includes a Storage Element and Compute Element.

Storage elements (SE) manage physical systems, disk caches and hierarchical mass storage systems, its an interface for grid jobs to underlying storage Storage Resource Management protocol and Globus Grid FTP protocol and others, A storage element requires an underlying storage system like hadoop, xrootd and a GridFTP server and an SRM interface.

A Compute Element(CE) allows grid users to run jobs on your site. It provides a bunch of services when run on the gatekeeper. The basic components include the GRAM and GridFTP on the same CE host to successfully enable file transfer mechanisms of Condor-G.

2.4 SPC

2.5 Job Router

Condor Job Router as defined in condor manual[1] transforms jobs from vanilla universe to grid universe according to a configurable policy. Condor Job Router helps to balance jobs across clusters by transferring excess jobs from one cluster to another. The rate of job submissions equals the rate at which site starts running the job. The other mechanisms including glidein and condor flocking does not provide this kind of balancing mechanism.

High throughput work flows benefits a lot from Job routing as its easy to reach the goal of distributing the workload and keeping as many computers as busy as possible. The Job Router does not know which site will run the jobs faster but it can decide whether to send more jobs to a site based on whether the already submitted Jobs are sitting idle

or not or whether the site has experienced recent job failures.

Bibliography

[1] Condor manual, section 5.6. 2.5

[2] J. Frey, Todd Tannenbaum, M. Livny, I. Foster, and S. Tuecke. Condor-g: a computation management agent for multi-institutional grids. In *High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on*, pages 55–63, 2001.