Deploying and Managing DSABR: A Distributed System for Automated Blender Rendering

Peter Bui, Grant Wuerker
Computer Science
{buipj,wuerkegr}@uwec.edu

1 Project Proposal

Two years ago, with the support of ORSP, we developed a collaborative animation rendering system called DSABR (Distributed System for Automated Blender Rendering). Due to the growing demand for increasingly complex digital 3D modeling and animation, we built a software application that harnesses the computing resources of multiple networked machines to speedup the rendering of digital animated videos. Given a file that contains a description of an animated scene, DSABR distributes the work required to generate an animation to multiple machines, gathers the results, and generates a final video file.

For instance, using DSABR and computing resources at both UWEC and the Center for High Throughput Computing at UW-Madison, we were able to reduce the rendering time for a video that took 72 minutes on a single machine down to 5 minutes using our system. Such dramatic speedups are vital for both reducing turnaround times for artists and for increasing the total number of videos that can be generated in a reasonable amount of time.

Last year we built off the initial success of DSABR by enhancing and extending the application by developing a web portal to allow users to interact with DSABR in a simplified manner. This web portal allowed users to simply login to a website, upload their Blender file, and hit a few buttons to automatically have their animation file rendered on multiple machines. When the process was complete, the website allowed users to preview or download their video directly from their web browser.

Unfortunately, the web portal is incomplete in a few important ways: First, it did not manage the workers required to perform the actual rendering. This was done manually in a haphazard manner and is not integrated into the web portal. Second, the system cannot handle more than one project at a time. This means that a user cannot queue up multiple jobs (likewise, multiple users cannot queue up jobs). This is obviously a serious defect and must be addressed before wider deployment. Finally, though the system served as a proof-of-concept it was not robust enough to present to Art faculty and students as a dependable service.

The goal of this project is to continue the development of DSABR and the web portal by integrating the management of worker processes into the web site and polishing the interface for use by Art faculty and students. In addition to these goals, we wish to explore a few interesting problems related to using the cloud to augment existing clusters and different scheduling algorithms for improving system throughput and efficiency.

2 Methodology

As noted in previous proposals in the various publications written about DSABR [2, 3], the rendering system utilizes the Python-WorkQueue [4] framework for distributing tasks to workers running on different machines in a HTCondor cluster [5]. With this framework, there is a master application which distributes tasks (in this case rendering jobs) to different workers who perform the computation and return the results to the master. Therefore, users must start both the master application and the worker processes. The current web portal only invokes the existing DSABR command line tool which serves as the master, but leaves it up to the user to manually run a set of workers.

For novice users, this is extremely confusing and goes against the purpose of the web portal: automate and simplify the use of DSABR. Our approach is to therefore develop the necessary software infrastructure to integrate the execution of remote workers into the web portal. This will involve updating the Python-WorkQueue bindings and writing some bridging software between the web portal and the HTCondor software. That is, in order for the web site to manage workers, it must be able to communicate with HTCondor. For this to happen, an intermediary application must translate commands from the other web site into a form HTCondor can understand.

Once these pieces are in place and functioning, we can then explore more interesting questions such as how should we allocate workers to multiple projects or multiple users. For instance, if we have 2 projects and N workers, is it better to give each project half of the workers? Or is it better to let one project go to completion before transferring workers from one project to another? Additionally, we will want to investigate ways of minimizing worker idle times and elastically growing and shrinking the number of workers based on the amount of resources available and the amount of work to be done.

3 Context and Significance

The continued development of the DSABR web portal allows us to further explore the growing practice of service-oriented science [6]. In this computing model, system developers create easy-to-use interfaces such as web portals to provide access to complex distributed applications. A prominent example of this is Nanohub [7], which provides web access to a variety of nanotechnology-related software. The goal of DSABR is to hide the complexities of distributed rendering of animation videos behind an accessible web portal for Art faculty and students at UW-Eau Claire.

Regarding the specific goals of this proposal, there is a large amount of interest in the distributed computing community in developing methods for measuring and determining the optimal scheduling of tasks and workers in a distributed system [8]. Likewise, building a portable middle-ware interface to HTCondor, as would be required by our project, would be a software project of interest to the wider HTCondor community who are looking at ways of integrating their system with different ecosystems.

4 History

This project has been funded by ORSP for the last two years. The first year yielded a working prototype of a command line tool that was presented at CERCA, HTCondor week, and in a paper at CLUSTER 2013 [2]. The second year yielded a working prototype of a web portal for utilizing the previous command line tool that was presented at MICS 2014 [3] and CERCA.

5 Dissemination

We hope to present the results of our research in the following venues:

1. A paper at the Midwest Computing and Instruction Symposium (MICS).
2. A poster or presentation at the Celebration of Excellence in Research and Creative Activity (CERCA).
3. A presentation at HTCondor Week at the University of Wisconsin - Madison.

If appropriate, we will also submit a poster or paper to other Computer Science workshops or conferences. Additionally, we plan on open sourcing our software so the other members of the Computer Science education community can utilize our tools. Finally, we plan on providing Art faculty and students access to this service.

References

[1] Python. https://www.python.org/.

[2] Peter Bui, Travis Boettcher, Nicholas Jaeger, Jeffrey Westphal. “Using Clusters in Undergraduate Research: Distributed Animation Rendering, Photo Processing, and Image Transcoding”. IEEE Cluster 2013 (CLUSTER2013). Indianapolis, IN. September, 2013.

[3] John Rankin, Travis Boettcher, and Peter Bui. “A Web Portal For An Animation Render Farm”. Midwest Instruction and Computing Symposium (MICS2014). Verona, WI. April, 2014.

[4] Peter Bui, Dinesh Rajan, Badi Abdul-Wahid, Jesus Izaguirre, Douglas Thain. “Work Queue + Python: A Framework For Scalable Scientific Ensemble Applications”. Workshop on Python for High Performance and Scientific Computing at SC 2011. November, 2011.

[5] Thain, Douglas, Todd Tannenbaum, and Miron Livny. “Distributed computing in practice: The Condor experience.” Concurrency and Computation: Practice and Experience 17.24 (2005): 323-356.

[6] Foster, Ian. “Service-oriented science.” Science 308.5723 (2005): 814-817.

[7] Klimeck, Gerhard, et al. “nanohub. org: Advancing education and research in nanotechnology.” Computing in Science and Engineering 10.5 (2008): 17-23.

[8] Armbrust, Michael, et al. “A view of cloud computing.” Communications of the ACM 53.4 (2010): 50-58.