HORT: Hadoop Online Ray Tracing with MapReduce

Lesley Northam*
Rob Smits†
University of Waterloo

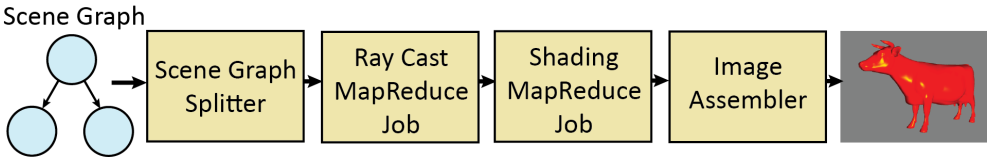

Figure 1: *The HORT rendering pipeline.*

CR Categories: I.3.2 [Computer Graphics]: Graphics Systems—Distributed/network graphics;

Keywords: ray tracer, mapreduce, high-performance rendering

1 Introduction

High quality computer-generated imagery (CGI) rendering is a demanding computational task [Hearn and Baker 1997]. To generate high-quality CGI, film studios rely on expensive, specialized, rendering clusters (render farms) [Rath 2009]. Render farms cost a premium over more general infrastructure such as infrastructure as a service (IaaS) offerings. Furthermore, render farms often provide limited selection of commercial rendering applications and have significant overhead with respect to requesting a job. This contrasts general IaaS offerings such as Amazon's Elastic Compute Cloud (EC2), which allows for straight-forward, automated provisioning of many virtualized machine instances.

To evaluate the effectiveness of rendering in a general IaaS environment, we have built and tested HORT, a MapReduce ray tracer which runs as a Hadoop ([Dean and Ghemawat 2008]) task.

2 Background

MapReduce is a programming paradigm and framework for batch processing large sets of data distributed over a cluster of commodity systems [Dean and Ghemawat 2008]. MapReduce is successful because it allows different types distributed batch jobs to be written in a similar way, and it allows these jobs to be executed, managed and monitored on the same infrastructure consistently.

The MapReduce implementation as described by Dean et al. is a proprietary system. For our experiments, we use Hadoop, a popular open source Java implementation of MapReduce [Bialecki et al. 2005].

Amazon offers several cloud computing products branded under the banner Amazon Web Services (AWS). AWS's IaaS offering is branded as Elastice Cloud Computing (EC2). To simplify the creation of running Hadoop jobs in EC2, Amazon released a service called Elastic MapReduce (EMR) which runs Hadoop. Our HORT experiments use EMR, although HORT could run on any Hadoop deployment.

*e-mail: lanortha@cs.uwaterloo.ca
†e-mail: rdfsmits@cs.uwaterloo.ca

3 Approach

To utilize the MapReduce programming model, HORT is comprised of several C++ programs which split the ray tracing pipeline into stages. Specifically, ray casting and shading operations are distinct MapReduce jobs, allowing for greater user control over job distribution. Additionally, we identified that sending large scene graphs (hundreds of megabytes to gigabytes) over the network to each MapReduce worker introduces large delays (transfer time) and unnecessary data replication. To solve this issue we split the scene graph into equal-size blocks, and each MapReduce worker receives a subset of blocks instead of the entire graph.

The scene graph is split by decomposing meshes into single-intersection components (i.e., individual triangles) and then components are divided into k user-specified blocks with an equal number of components. Map workers perform computation (e.g., ray-object intersection tests) on their blocks, and Reduce workers combine results to produce the final answer. For example, Map workers from the ray casting stage produce a list of intersection points, and the corresponding Reduce workers find the intersection nearest the camera.

We executed several experiments with HORT using EMR to determine the feasibility of using IaaS services for rendering. We found that HORT continues to show improved performance as more machine instances are made available. To demonstrate fault tolerance, we intentionally terminated an EC2 node used by our Hadoop cluster during a rendering job and observed its successful completion. Finally, we noted that adjusting the number of instances and MapReduce workers is flexible and user-friendly.

References

BIALECKI, A., ET AL., 2005. Hadoop: a framework for running applications on large clusters built of commodity hardware. `http://lucene.apache.org/hadoop`.

DEAN, J., AND GHEMAWAT, S. 2008. MapReduce: Simplified data processing on large clusters. *Communications of the ACM 51*, 1, 107–113.

HEARN, D., AND BAKER, M. P. 1997. *Computer graphics (2nd ed.): C version*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

RATH, J., 2009. The data-crunching powerhouse behind 'avatar'. `http://www.datacenterknowledge.com/archives/2009/12/22/the-data-crunching-power`, December.

SIGGRAPH 2011, Vancouver, British Columbia, Canada, August 7 – 11, 2011.
ISBN 978-1-4503-0921-9/11/0008