Work Queue: A Master/Worker Framework for PDC Education and Research

Peter Bui, Department of Computer Science, University of Wisconsin - Eau Claire

Work Queue

Work Queue [1] is a framework for building large master-worker applications that span many computers including clusters, clouds, and grids.

Work Queue applications are written in a variety of programming languages using a simple API that allows users to define tasks, submit them to the queue, and wait for completion.

Tasks are executed by a standard worker process that can run on any available machine. Each worker calls home to the master process, arranges for data transfer, and executes the tasks. The system handles a wide variety of failures, allowing for dynamically scalable and robust applications.

Education

Work Queue allows students to explore parallel and distributed computing in a variety of courses including electives, core classes, and introductory courses.

Cloud Computing

Explore MapReduce by analyzing Linux Source code

Using custom MapReduce implementation written in Python and Work Queue perform the following on the Linux source code repository:

1. Generate word frequencies
2. Extract email addresses and curse words
3. Create inverted index of functions to source files
4. Determine page rank of each source file

Computer Organization and Design

Implement brute-force password cracker

Using the Go programming language and Work Queue, implement each of the following versions of a brute-force password cracker:

1. Serial
2. Parallel (multi-core)
3. Distributed (multi-node)

Once implemented, perform an experiment comparing all three versions.

Computing for the Sciences and Mathematics

Perform large scale data analysis and simulation

Using IPython Notebook and Work Queue, perform distributed simulations and analysis of various datasets:

1. Aggregating and processing web data
2. Analyzing and filtering images
3. Monte Carlo simulations

Perform visualization of the data using the notebook interface.

Research

Work Queue enables undergraduates to quickly dive into research projects involving distributed computing [4]. It is particularly well-suited for data-intensive applications.

DSABR

Distributed system for automated Blender rendering

Using Python and Work Queue, implement a system for animating Blender files using multiple distribute clusters.

DP3

Distributed photo processing pipeline [2]

Support field scientists by allowing them to upload image data to a system that will automatically process and archive their photos.

Scalable Image Transcoding

Test the scalability of an image processing workflow [3]

Use Work Queue to implement an image processing workflow that converts different sets of images into different formats and measure the application of the system.

Software

- CCTools: http://www.nd.edu/~ccl/
- Work Queue MapReduce: http://bitbucket.org/pbui/work-queue-mapreduce

Acknowledgements

We thank the following groups for supporting our work:

- Cooperative Computing Lab, University of Notre Dame
- Office of Research and Sponsored Programs, University of Wisconsin - Eau Claire
- Center for High Throughput Computing, University of Wisconsin - Madison

References