

# Hogwash: Simple parallel computing for everyone

David McClosky

`dmcc@cs.brown.edu`

# Overview

- What can Hogwash do?
- Parallel frameworks at Brown
- Setting up Hogwash
- Two interfaces
- Advanced Topics

# What can Hogwash do?

- Facilitate running **jobs** on our parallel computing systems
- A job is a program run non-concurrently, typically some slice of a larger task
- Jobs tend to be CPU, I/O, and/or RAM intensive
- Some examples:
  - Parsing 24 million sentences (50,387 jobs, each job parsed 250 sentences)
  - Simulations with different random seeds
  - Pretty much any operation that can be done in parallel over a set of files or dataset

# Design goals

- Abstract details of running jobs in parallel
- Help organize the output of said jobs (act as research journal/database)
- Completely scriptable in Python
- Programming-free interface for common cases

# Relation to MapReduce

- Google's MapReduce and Hadoop calculate:

$$result = reduce(f_{reduce}, map(f_{map}, args))$$

- Hogwash doesn't have the `reduce` component
  - but it doesn't seem to be necessary for most tasks
- Dependencies between jobs are possible in Hogwash, though `map` and `reduce` jobs will not run concurrently.

# Parallel computing at Brown

- There are a ton of parallel frameworks at Brown (CCV, CCMB, ...)
- Hogwash provides a unified interface to three of them.
  - Quahog
    - developed in-house, uses “idle” desktop machines in the department, **deprecated**
  - Sun Grid Engine
    - 64 multiprocessor machines, several for high memory and benchmarking
  - Local operation
    - Can be used for testing/debugging but also normal use

# A bit of bit-ness business

- Most desktop machines in 32-bit mode
- Machines on the Grid are almost entirely 64-bit (four 32-bit machines)
- Compile platform dependent code on the Grid
- Can be a headache. Maybe tstaff will fix?

# Compiling 64-bit code on the grid

1. `ssh to sge`
2. **Source** `/com/sge/default/common/settings.sh` or `/com/sge/default/common/settings.csh` depending on your shell.
3. Run `qlogin -l arch=lx24-amd64 -now n` to get a login shell on a 64-bit machine.
4. Compile your code.

# Setting up Hogwash

Just need one small environment change to `.environment`:

```
pathappend PATH /pro/dpg/dmcc/bin/
```

To use the Python interface add:

```
pathappend PYTHONPATH /pro/dpg/dmcc/python/
```

```
pathappend PYTHONPATH /pro/dpg/contrib/python/
```

Darcs source code repository:

```
/pro/dpg/dmcc/darcs/Hogwash/
```

# Two interfaces

- Command-line interface (most tasks)
- Python interface

Demo time!

# Advanced Hogwash

- Dependencies and pipelines
- Actions (different functions per job)
- Barrier pattern

# More information

## Website

`http://cs.brown.edu/~dmcc/hogwash/`

## Materials in this talk

`/pro/dpg/dmcc/hogwash-demo/`