CSCI 5105 Programming Assignment 2

Ramith Jayatilleka

Design document is in `DESIGN.md`. Evaluation is in `EVALUATION.md`. Those documents as well as this readme are available in pdf form under `pdfs/`.
The evaluation includes images, so you will need to view the pdf to see them.

Building and Installing

- `make`
    - Builds the project
- `make clean`
    - Deletes non-source folders
- `make install`
    - Builds and installs to `$HOME/jaya0089-mapreduce`
- `make uninstall`
    - Deletes `$HOME/jaya0089-mapreduce`

Running

Orchestrator

Orchestrator is the test harness I wrote (in Python3). It will read a config, SSH into each host, and launch the storages. After exiting, it cleans up all remote processes and ports. It also copies logs to the local `logs/` and metrics to the local `logs/metrics/`.

Requirements

- Each host has Mapreduce installed at `$HOME/jaya0089-mapreduce`.
    - Running `make install` once on a Cselabs machine solves this.
- Passwordless SSH is setup for each host.

Launching

Run `bin/orchestrator <username> <config_path> <chunk_size> <chunks_per_merge> <redundancy> <fail_probability>`. Ex: `bin/orchestrator jaya0089 test/configs/localhost.txt 1000000 8 1 0`.

<username> should be the user to ssh to the hosts as.
<config_path> is the path to the config. I wrote a few configs to test/configs.
<chunk_size>, <chunks_per_merge>, and <fail_probability> are self-explanatory.
<redundancy> is how many simultaneous tasks to launch for a given sort or merge.

Press Enter to shutdown the servers. It will clean up all remote processes.

Config

```
<Master Host>
<Worker Host:Port>
<Worker Host:Port>
<Worker Host:Port>
...
```

```
csel-x31-01.cselabs.umn.edu
csel-x31-01.cselabs.umn.edu:50001
csel-x31-01.cselabs.umn.edu:50002
csel-x31-02.cselabs.umn.edu:50001
```

The example above will:

- Launch the master at csel-x31-01:50000.
- Launch a worker at csel-x31-01:50001,
- Launch a worker at csel-x31-01:50002.
- Launch a worker at csel-x31-02:50001.

Client

The client needs to run on the same host as the master. Run `bin/client`.
The interface runs commands line by line:

```
mergesort 1000000

> ----- BEGIN mergesort 1000000 -----
> ~~~
> ~~~
> ----- END -----
```

Commands

- `masterInfo`
    - Displays the list of worker servers.
- `mergesort <filename>`
    - This will choose the file at `work/input/<filename>` and run a mergesort on it. Then it outputs:

        - output id (the output will be stored at `work/output/<output_id>`.
        - time elapsed
        - count of tasks run
        - count and percentage of tasks failed
        - convenient diff command to test that the mergesort worked.

Launching Master and Workers Directly

- `bin/master <chunk_size> <chunks_per_merge> <redundancy> [<worker_host:port>]...`
    - Launches a storage at `localhost:50000`, with given parameters and workers.
- `bin/worker <port> <fail_prob>`
    - Launches a storage at `localhost:<port>`, with given fail probability.

Testing

First, launch the servers. Then open the client and run mergesort on any of the files given. The client will run the mergesort, and then generate a convenient `diff` command that compares the given sorted output to the result of the mergesort. When run, the diff should show no output.