Manual

To run the code of the programming assignment #3, one will need to run only one executable per node. Two helping scripts have been written to simplify deployment of the code on EC2 instances. In this document, we will start by explaining how to deploy the assignment on EC2 and then we will describe how to run the main program on every node.

The source code can be found in the `dfs` folder and the code related to EC2 is stored in the `ec2` folder.

1. EC2 helpers.

In addition to the main program, two scripts have been developed to help deploying the assignment on EC2 instances. They both assume that instances running on EC2 are nodes used for the assignment. To run these scripts, the user will need to install the `boto` Python package.

- The `deploy_config.py` script takes in parameters the type of DFS the user wants to deploy (distributed or centralized), an AWS credentials file, and the EC2 SSH key. It lists the running nodes on EC2, automatically creates a configuration file and copy this configuration files to all running nodes on EC2. By running locally this command, we configure all the nodes for the programming assignment.

  ```
  python deploy_config.py centralized credentials.csv ssh_key.pem
  python deploy_config.py distributed credentials.csv ssh_key.pem
  ```

  If the chosen system type is centralized, then a random node on EC2 is taken as the central indexing server and the other ones are nodes running standard clients. In the case of a distributed system, all the nodes run the same program which include the distributed indexing server, a file server, and an user interface.
- The `connect.py` script takes the generated configuration file, a node id, and the EC2 SSH key as arguments. It connects the user in SSH to the corresponded EC2 instance. This simplifies the connection to the nodes by letting the user use a node identifier (i.e. an integer) to connect to it instead of having to get the IP address every time.

2. Main executable.

We have simplified the user's actions to run the program on an EC2 node. We created a script `run.py` that parses the configuration stored on the node -- `deploy_config.py` must have been called on a local machine before for this to function -- and runs the corresponding executable (central indexing server, node in centralized system, or node in distributed system) with the correct parameters.

Once the program is running, the user can interact with it and give it command through a command line interface. The following actions are possible:

- `exit` terminates the connections, and exit the program.
- `lookup <filename>` requests the indexing server (IS) for the lists of other peers that have the file and download the file from the available peers.
- `search <filename>` requests the IS for the lists of other peers having that file. It is different from `lookup` because it does not download the file.
- `register <filepath>` registers a file to the IS.
- `register <regex> true` registers all the files matching the regular expression to the IS.
- `list` lists all the files indexed by the IS.

- `help` displays the help screen.
- `benchmark 1 <lookup|search|register>` runs 10000 `lookup`, `search`, or `register` sequentially. It computes the total time of the 10000 requests in seconds.
- `benchmark 2 <1K|10K|100K|1M|10M|100M|1G>` retrieves all the files of a given size that are stored in the other nodes. It computes the total time of these operations in seconds.