**CS4287-5287: Principles of Cloud Computing**

Programming Assignment #5: MapReduce + Docker Swarm

Handed out: Nov 16, 2017; Due: Sunday Dec 10, 2017 in BrightSpace (team-based)

This assignment will enable us to apply our understanding of MapReduce and Docker containers, specifically Docker Swarm, in a single setup. To that end, we will build upon our expertise running a MapReduce program and apply it for the energy data set example (explanation below). You will use the homegrown MapReduce code that has been developed to solve the word count example and modify it to solve the energy example. The provided MapReduce code works seamlessly using both Mininet as well as in the Cloud using Docker Swarm. The README file explains how to run the example. Please see the Lecture 23 slides on Docker Swarm on how to setup the swarm.

The energy application is taken from the ACM DEBS 2014 Grand Challenge on Smart Homes described here: <http://debs.org/?p=75> and snippets of which are reproduced below.

*The ACM DEBS 2014 Grand Challenge is the fourth in a series of challenges which seek to provide a common ground and evaluation criteria for a competition aimed at both research and industrial event-based systems. The goal of the Grand Challenge competition is to implement a solution to a problem provided by the Grand Challenge organizers. The DEBS Grand Challenge series provides problems which are relevant to the industry at large. DEBS Grand Challenge problems allow for evaluation of event based systems using real-life data and queries.*

***Data***

*The Grand Challenge 2014 data set is based on recordings originating from smart plugs, which are deployed in private households. A smart plug plays a role of a proxy between the wall power outlet and the device connected to it. It is equipped with a range of sensors which measure different, power consumption related, values. For the purpose of the DEBS 2014 Grand Challenge a number of smart plugs has been deployed in private households with data being collected roughly every second for each sensor in each smart plug. It has to be noted that the data set is collected in an uncontrolled, real-world environment, which implies the possibility of malformed data as well as missing measurements.*

*For the DEBS 2014 Grand Challenge we assume a hierarchical structure with a house, identified by a unique house id, being the topmost entity. Every house contains one or more households, identified by a unique household id (within a house). Every household contains one or more smart plugs, each identified by a unique plug id (within a household). Every smart plug contains exactly two sensors:  
(1) load sensor measuring current load with Watt as unit (2) work sensor measuring total accumulated work since the start (or reset) of the sensor with kWh as unit.*

*The schema of the base stream is following:*

***id****– a unique identifier of the measurement [32 bit unsigned integer value]****timestamp****– timestamp of measurement (number of seconds since January 1, 1970, 00:00:00 GMT) [32 bit unsigned integer value]****value****– the measurement [32 bit floating point]****property****– type of the measurement: 0 for work or 1 for load [boolean]****plug\_id****– a unique identifier (within a household) of the smart plug [32 bit unsigned integer value]****household\_id****– a unique identifier of a household (within a house) where the plug is located [32 bit unsigned integer value]****house\_id****– a unique identifier of a house where the household with the plug is located [32 bit unsigned integer value]*

**What to accomplish?**

The goal is to compute the average work performed and average load imposed on each unique smart plug, where the uniqueness of a plug is defined by the id of the house, the household and the plug taken together.

**How to proceed?**

* No need to use Vagrant/Ansible and Docker for this assignment. You can manually fire up your Horizon VMs (two of them) and Chameleon VM (say one). Then use your code to run the map and reduce jobs in different virtualized hosts. Use Docker Swarm and let it deploy the map and reduce workers using its built-in load balancing strategies. One such virtualized host can serve as the Master node for the map-reduce framework. Have the final results available in one place. The existing master file already stores results in a file.
* Using the data file that is supplied, and start the map reduce task. Try for different values of M and R, say the following combinations: M=10, R=2; M=50, R=5; M=100, R=10. Time the experiment start to finish.

Please use your previously used floating IP addresses for your Horizon and Chameleon VMs

**Submission:**

Create a zip file with your code and submit as a team. Demonstrate the working of your code to Travis. For that, please set up a time during the week of Dec 11th and show Travis the working of your code.