1. Pre Steps :

* Install 16 nodes swift cluster
* Create 40gb volume and attached to head node
* Download and Extract 10 gb input file on new mounted volume on headnode
* Setup swift on head node

1. Compile swift programs using : swift mywordcount.swift
2. Check output: After successfully completion of program, output folder is generated. In that you will find a “all\_mergefile.out” output

Source code :

1. Mywordcount.swift

// Cloud-Computing Programming Assignment 2

// Swift WORDCOUNT PROGRAM

// Author : Ronakkumar Makadiya

// Created on: 10/20/2014

// creating variable type file

type file;

// word count function

app (file out, file log) word\_logic (file input1, file wcscript)

{

// calling wc.sh file for wordcount

bash @wcscript @input1 stdout=@out stderr=@log;

}

app (file out, file log) merge\_allfileoutput (file ronak[], file m\_s)

{

// calling join.sh file

bash @m\_s filenames(ronak) stdout=@out stderr=@log;

}

// creating 2 file type variable

file wcscript <"wc.sh">;

file script\_file<"join.sh">;

// define file type mappers array. So read all txt file as inputs

file textfiles[] <filesys\_mapper; location="inputs", suffix="txt">;

file wc\_inter[];

// read 1 by 1 text input file

foreach textfile,i in textfiles

{

file wc\_out <single\_file\_mapper; file=strcat("output/sim\_",i,".out")>;

file wc\_log <single\_file\_mapper; file=strcat("output/sim\_",i,".log")>;

// calling word count

(wc\_out, wc\_log) = word\_logic (textfiles[i], wcscript);

wc\_inter[i] = wc\_out; //store output in wc\_inter array

}

// Join all the outputs

file out\_var<"output/all\_mergefile.out">;

file log\_var<"output/all\_mergefile.log">;

// call mearge\_allfileoutput function

(out\_var, log\_var) = merge\_allfileoutput(wc\_inter, script\_file);

1. Join.sh

echo "FILES TO MERGE : $\*" 1>&2

cat $\* | sort -k2 | awk -F\ '{a[$2]+=$1} END {for (i in a) print i" "a[i]}' | sort

1. Wc.sh

if [ -z $1 ]

#check input file arguments

then

echo "input file as first argument"

fi

# word count

cat $1 | sed -e 's/\([[:punct:]]\)//g' | sed 's/\.//g;s/\(.\*\)/\L\1/;s/\ /\n/g' | sort | uniq -c | tr -d $'\r'

1. Swift.confs

sites: cloud-static

site.local {

filesystem {

type: "local"

URL: "localhost"

}

execution {

type: "local"

URL: "localhost"

}

workDirectory: /tmp/${env.USER}/swiftwork

maxParallelTasks: 32

initialParallelTasks: 31

app.ALL {executable: "\*"}

}

site.cloud-static {

execution {

type:"coaster-persistent"

URL: "http://127.0.0.1:50010"

jobManager: "local:local"

options {

maxJobs: 10

tasksPerNode: 4

}

}

initialParallelTasks: 20

maxParallelTasks: 20

filesystem.type: local

workDirectory: /tmp/swiftwork

staging: local

app.ALL {executable: "\*"}

}

lazyErrors: false

executionRetries: 0

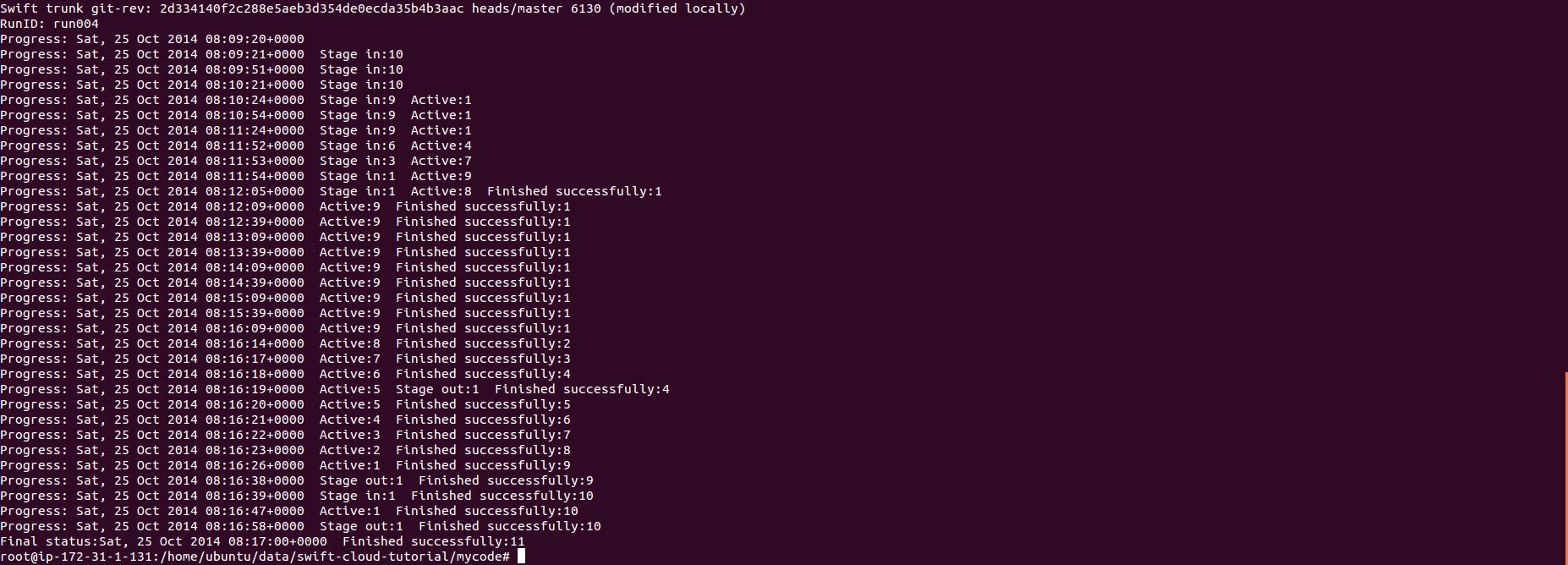
keepSiteDir: true

providerStagingPinSwiftFiles: false

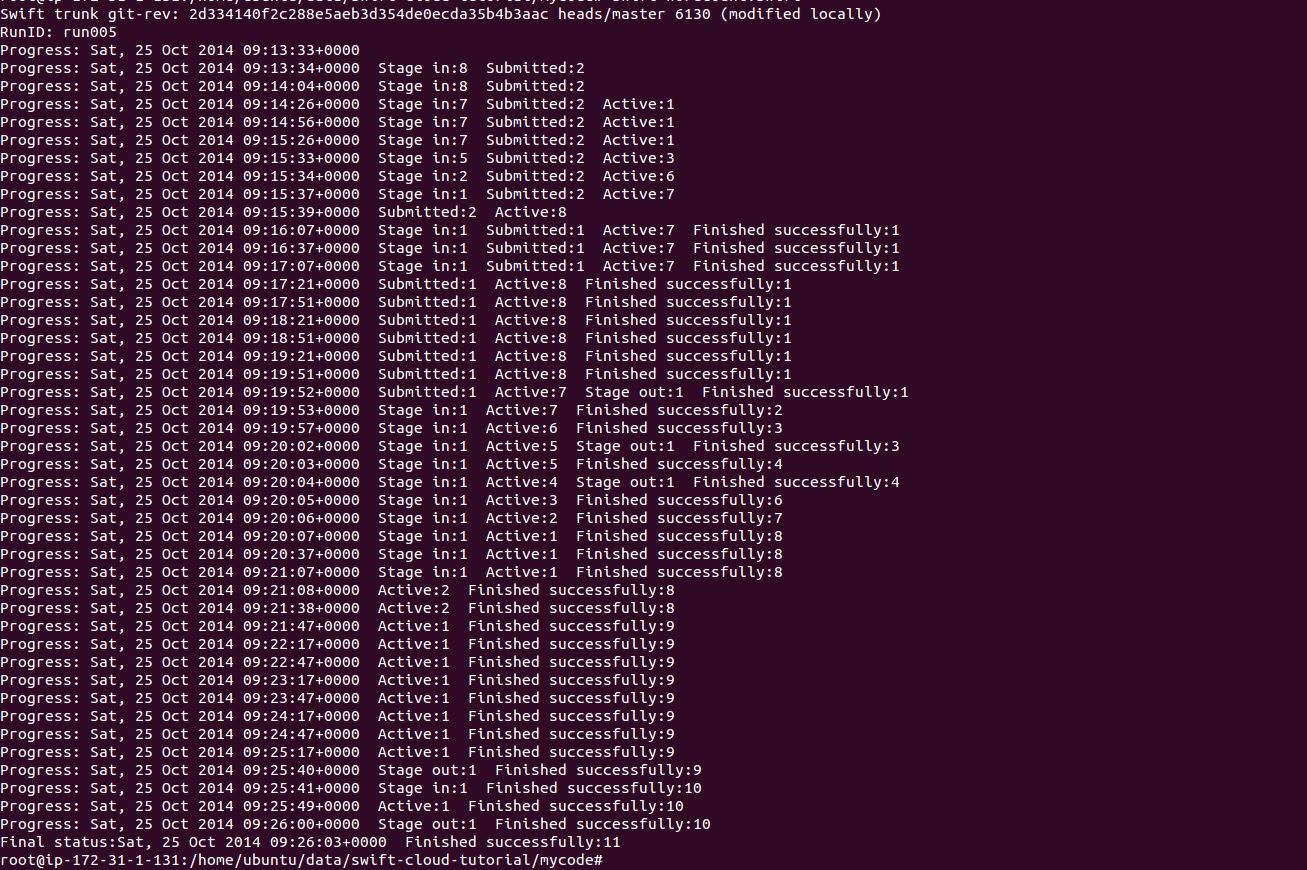
alwaysTransferWrapperLog: true

OUTPUT :

16 Nodes :

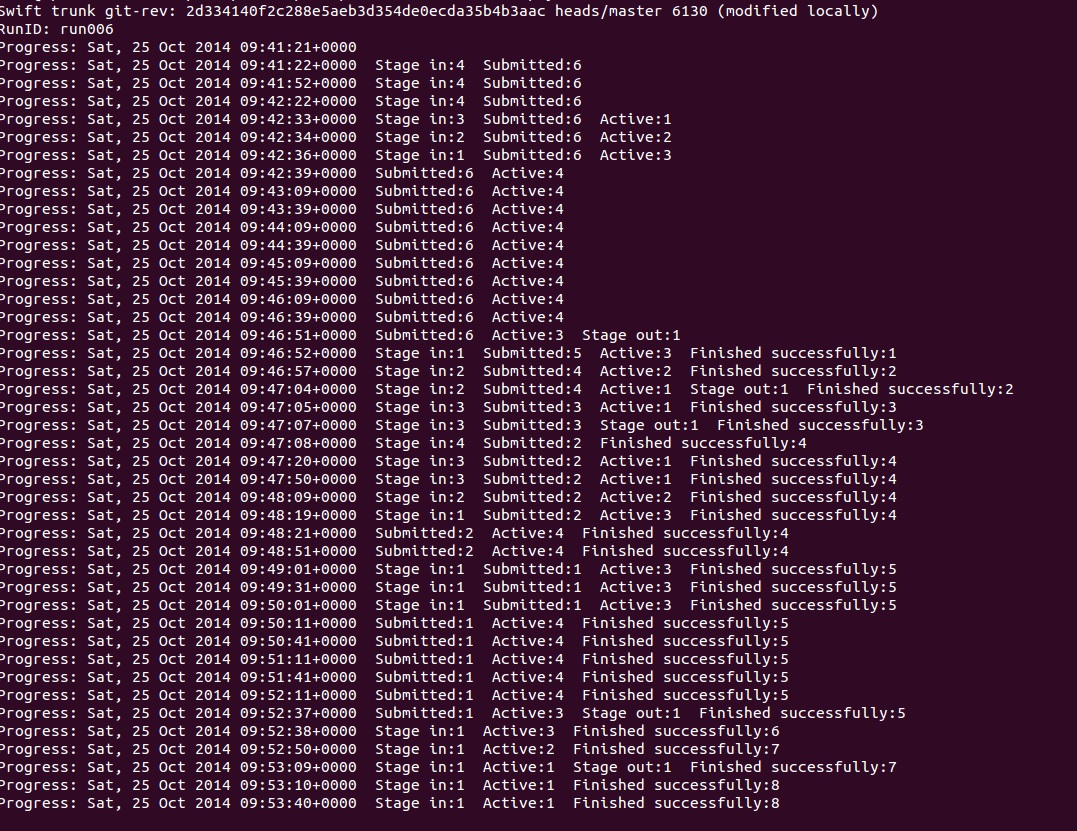


8 Nodes:



4 Nodes :

Part 1:



Part 2 :

