Homework 13

Sang Doan

12/7/2020

Problem 1

Run on EC2: `create_folders()` → `create_results_bucket()` → `save_csv_files()` → `map_matrices()`.

```
system('aws s3 cp s3://hw13-prob1-results/2019-02.csv processed_data/.')

mat201902 <- fread(
    'processed_data/2019-02.csv',
    header = TRUE
  )

mat201902 %>% select(1:5) %>% head
```

```
##       1 2  3   4 5
## 1: 466 0  0   0 0
## 2:   0 0  0   0 0
## 3:   0 0 22   0 0
## 4:  11 0  0 553 0
## 5:   0 0  0   0 2
## 6:   0 0  0   0 0
```

Problem 2

Group of two with most connectivity in February 2019 is 236 and 237 with 155796 trips in total.

```
find_most_connected('02', '2019', 2)
```

```
## [1]    233    234 155796
```

Group of three with most connectivity in February 2019 is 236, 237, and 141 with 232099 trips in total.

```
# Run on EC2
find_most_connected('02', '2019', 3)
```

Problem 3

Run on EC2: `find_three_most_connected_all_months()`.

Code

prob1.R

```r
save_csv_files <- function() {
  # Downloads csv files from S3 bucket nyc-tlc in 2019 and 2020 to local folder.

  # Copy yellow files with names having patterns "2019" or "2020" to folder raw_data.
  system('aws s3 cp "s3://nyc-tlc/trip data/" raw_data/. --recursive --exclude "*" --include "yellow*20
}

# Create folders.
create_folders <- function()
  system('mkdir raw_data processed_data dynamic')

# Create S3 bucket.
create_results_bucket <- function()
  system('aws s3 create-bucket -bucket hw13-prob1-results')

processed_files <- function() {
  system('aws s3 ls s3://hw13-prob1-results > dynamic/processed_files.txt')
  processed <- readLines('dynamic/processed_files.txt')

  return(processed)
}

map_matrices <- function() {
  # Converts all csv files in folder raw_data into matrices and saves them to an S3 bucket.

  # Convert each csv file into a matrix
  csvfiles <- list.files(
    'raw_data',
    full.names = TRUE
  )

  processed <- processed_files()

  for(csvfile in csvfiles) {
    month <- str_sub(csvfile, 26, 32) # Example: 2019-01

    # Detect if the file has already been processed.
    # If yes, skip the loop.
    if(str_detect(processed, month))
      next

    dat <- fread(csvfile)[
        , .(PULocationID, DOLocationID) # Keep what matters
      ]

    # Build matrix
    out_matrix <- dat %>%
      graph.data.frame %>%
      get.adjacency %>%
      as.matrix
    # Reorder matrix
    out_matrix <- out_matrix[, order( as.numeric( colnames(out_matrix) ) )]
    out_matrix <- out_matrix[ order( as.numeric( rownames(out_matrix) ) ), ]

    # Write matrix
    out_path <- str_glue('processed_data/', month, '.csv')

    fwrite(
      out_matrix,
      out_path
    )

    # Push to S3
    str_glue(
      'aws s3 cp ',
      out_path,
      ' s3://hw13-prob1-results'
    ) %>%
      system
  }

}
```

prob2.R

```r
find_most_connected <- function(month, year, num_nodes) {

  # Retrieve data from S3
  str_glue(
    'aws s3 cp s3://hw13-prob1-results/',
    year, '-', month, '.csv ', 'dynamic/matx.csv'
  ) %>%
    system

  # Read in data and convert into an igraph object
  dat <- fread('dynamic/matx.csv', header = TRUE) %>%
    as.matrix %>%
    graph_from_adjacency_matrix

  # Create a matrix of possible combinations.
  # Excluding sets with repeated numbers (e.g., 1, 1, 2).
  # Order does not matter (e.g., 1, 2, 3 = 2, 3, 1).
  combs <- gtools::combinations(gorder(dat), num_nodes)

  # Build a vector of numbers of connections in each combination
  connectivity <- mclapply(
      1:nrow(combs), # Number of combinations
      function(x) {
        # Create subgraph from big graph
        subgr <- subgraph(dat, combs[x,])
          # combs[x,] = combination being considered = vertices of new graph
        # Return number of edges of subgraph (ie number of connections)
        return(gsize(subgr))
      },
      mc.cores = 8
    ) %>%
    unlist

  most_connected <- combs[which.max(connectivity),]
  out <- append(most_connected, max(connectivity))

  return(out)
}
```

prob3.R

```
find_three_most_connected_all_months <- function() {

  months <- processed_files %>%
    str_sub(32, 38) # Vector example: "2019-01" "2019-02" "2019-03"

  # Set a plan to instruct the next future function to run on 6 cores
  future::plan(multicore, workers = 6)
  out <- furrr::future_map_dfr(
      months,
      function(month) {
        mon <- str_sub(month, 6, 7)
        yr <- str_sub(month, 1, 4)

        most_connected <- find_most_connected(mon, yr, 3)

        out <- data.frame(
            time = month,
            r1 = most_connected[1],
            r2 = most_connected[2],
            r3 = most_connected[3],
            ntrips = most_connected[4],
            stringsAsFactors = FALSE
          )

        return(out)
      }
    )

  out_path <- 'processed_data/three_most_connected.csv'
  fwrite(out, out_path)

  str_glue('aws s3 cp processed_data/three_most_connected.csv s3://hw13-prob1-results/.') %>%
    system

}
```

config.R

```r
library(dplyr)
library(stringr)
library(data.table)
library(furrr)
library(igraph)
library(gtools)
library(parallel)

source('prob1.R')
source('prob2.R')
source('prob3.R')
```