STA242 - HW04 - Working with Large files

Hugh Crockford

March 3, 2013

1 Using Shell tools

2 Plotting

Plotting the resulting grid was achieved using scipy's toimage function. Each car cell had to be described in RGB format, which was mapped using integer describing color. One bug in my code I was unable to identify resulted in vertical bars of red cars located along left of plot. This seemed to only occur when generating large maps (larger than 100 squared), and I assume had something to do with my random selection code.

3 Moving cars.

Moving cars was achieved by selecting all color based on time, then incrementing their x or y positions respectively. Any cars exceeding the dimensions of map in either dimension was placed back at beginning.

I didn't include dimensions with map object as in H02, but perhaps this could be achieved using a similar list function, and remove the need to include map dimensions in call to move function. The move function worked well for smaller maps where I could visually check the results, however because I couldn't get it to function within a loop I was unable to assess it on larger maps.

4 Checking Jams.

Using Python I struggled with the referencing of car locations that would allow me to check jams. As discussed above I tried numerous methods but continued to come up against immutable tuples, a concept I was not familiar with before this project. Using string functions to paste positions and then check for duplicates in old vs new positions (using the any function) was also a method I tried to generate a list of blocked cars, but I ran into problems when mapping positions of blocked cars back to their old positions and preventing these cars from being moved. Another issue that confused me in my initial attempts was the concept of modifying lists/arrays in place versus merely changing a view.

I've since found a few books on basic python with which I spent most of my time this week learning the basics of the language, and also pandas, scikit-learn, and numpy tutorials and books online which will assist my application of machine learning tasks in this language. I hope to complete my project for this course using image recognition tools (possibly neural nets??) that are present in these packages, assuming I can overcome the steep learning curve.

5 Git

I've been using git since the start of the year to manage projects for classes and research projects I'm involved in. Using ssh keypairs and running git from the shell allows easy version control between multiple computers and was a great tool when collaborating on writing grants/papers etc, especially when using raw text such as latex documents. My github repo can be found at: https://github.com/hughec1329

6 CODE

6.1 R

```
# STA242 H04 - cli for large data
# 20130227 - HCrockford


##################################
## With shell tools
##################################

 ports = c("LAX","OAK","SMF","SFO")
 coll = paste(ports,"|",sep="", collapse = "") # lthrow
    together for grep.
 pflat = substr(coll,1,nchar(coll)-1)
 fill = "~/data/airports/open/1987.csv"
com = sprintf("cat %s | awk -F ',' '{print $17}'| grep
    -E '%s' | sort | uniq -c",fill,pflat)
 ret = system(com,intern = TRUE)
 foo = sapply(ret,function(i) unlist(strsplit(i," ")))
 planes = structure(as.integer(lapply(foo,function(i) i[
    length(i)-1])) , names = lapply(foo,function(i) i[
    length(i)]))


### mean and SD of arrival delay times.

 ports = c('"LAX"','"OAK"','"SMF"','"SFO"')         # damn
    quotes.

 fill = "~/data/airports/open/1987.csv"
com = sprintf("cat %s | awk -F ',' '{if ($17 == %s) s+=
    $15} END {print s}'",fill,ports)
com = sprintf("cat %s | awk -F ',' '{if ($17 == "LAX")
    s+=$15} END {print s}'",fill,ports)        # damn
    quotes!
```

```
ret = system(com,intern = TRUE)
foo = sapply(ret ,function(i) unlist(strsplit(i," ")))
planes = structure(as.numeric(lapply(nom,function(i) i[
   length(i)-1])) , names = lapply(nom,function(i) i[
   length(i)]))


com = paste("./plan.sh", "mean", fill)
ret = system(com,intern = TRUE)
foo = sapply(ret ,function(i) unlist(strsplit(i," ")))
means = structure(as.numeric(foo[2,]),names = foo[1,])
means


#### SD

com = paste("./plan.sh", "sd", fill)
ret = system(com,intern = TRUE)
foo = sapply(ret ,function(i) unlist(strsplit(i," ")))
sds = structure(as.numeric(foo[2,]),names = foo[1,])
sds


###############################
### within R - Cant do? without messy slow loops?
###############################

fill = "~/data/airports/open/1987.csv"
con = file(fill)
nlin = as.integer(strsplit(system(paste("wc -l",fill),
   intern=TRUE)," ")[[1]][1])
block = 1000
arrdelay = numeric(1000000)
nplanes =0
passes = round(nlin/block)
# for(i in 1:passes){            # very slow with only
   1000 planes!!
      plan = readLines(con,block)
```

```
          lapply(strsplit(plan,","), function(i) {
                  if(i[17] == "LAX"){
                          nplanes = nplanes+1
                          arrdelay[nplanes] = i[15]
                  }
                  men = sum(arrdelay)/nplanes
                  sdd = sum((arrdelay - men)^2)
                  }
          )
                  # return(sdd)
# }

################################
### with DB's
################################

#### SQLITE

library(RSQLite)
con = dbConnect(SQLite(),dbname="~/data/airports/delays
   .db")
ports = c("'LAX'","'OAK'","'SMF'","'SFO'")

## COUNTS
com = sprintf("SELECT COUNT(arrd) FROM delays WHERE
   origin = %s",ports)
lapply(com,function(i) fetch(dbSendQuery(con,i)))

## AVG
com = sprintf("SELECT AVG(arrd) FROM delays WHERE
   origin = %s",ports)
ret = lapply(com,function(i) fetch(dbSendQuery(con,i)))
avg = structure(as.numeric(unlist(ret)),names = ports)
   #

## SD
com = sprintf("SELECT SUM((arrd-%s)*(arrd-%s))/COUNT(
   arrd) FROM delays WHERE origin = %s",avg,avg,ports)
```

```
ret = lapply(com,function(i) fetch(dbSendQuery(con,i)))
sd = structure(sqrt(as.numeric(unlist(ret))),names =
   ports) #

## check
# cat 1987.csv | awk -F "," "{if ($17 == 'LAX') print
   $15}" > lax ..From shell
lax = read.table("lax")
sd(lax,na.rm=TRUE)       # 33 - agrees w shell but not
   sql
# made test table to check calculation
# N-1!!!
```

6.2 shell

```
#!/bin/bash
case "$1" in
        count)
                lax=`cat $2 | awk -F ',' '{if ($17 == "
                   LAX" ) s+=$15} END {print s}'`
                oak=`cat $2 | awk -F ',' '{if ($17 == "
                   OAK" ) s+=$15} END {print s}'`
                smf=`cat $2 | awk -F ',' '{if ($17 == "
                   SMF" ) s+=$15} END {print s}'`
                sfo=`cat $2 | awk -F ',' '{if ($17 == "
                   SFO" ) s+=$15} END {print s}'`
                echo "lax" $lax
                echo "oak" $oak
                echo "smf" $smf
                echo "sfo" $sfo
                ;;
        mean)
                lax=`cat $2 | awk -F ',' '{if ($17 == "
                   LAX" ) {sum+=$15;nrec+=1}} END {
                   print sum/nrec}'`
                oak=`cat $2 | awk -F ',' '{if ($17 == "
                   OAK" ) {sum+=$15;nrec+=1}} END {
                   print sum/nrec}'`
```

```
            smf=`cat $2 | awk -F ',' '{if ($17 == "
               SMF" ) {sum+=$15;nrec+=1}} END {
               print sum/nrec}'`
            sfo=`cat $2 | awk -F ',' '{if ($17 == "
               SFO" ) {sum+=$15;nrec+=1}} END {
               print sum/nrec}'`
            echo "lax" $lax
            echo "oak" $oak
            echo "smf" $smf
            echo "sfo" $sfo
            ;;
      sd)
            lax=`cat $2 | awk -F ',' '{if ($17 == "
               LAX" ) {sum+=$15;nrec+=1;array[nrec
               ]=$15}} END {for(x=1;x<=nrec;x++)
               sumsq+=(array[x]-(sum/nrec))^2 ;
               print sqrt(sumsq/nrec)}'`
            oak=`cat $2 | awk -F ',' '{if ($17 == "
               OAK" ) {sum+=$15;nrec+=1;array[nrec
               ]=$15}} END {for(x=1;x<=nrec;x++)
               sumsq+=(array[x]-(sum/nrec))^2 ;
               print sqrt(sumsq/nrec)}'`
            smf=`cat $2 | awk -F ',' '{if ($17 == "
               SMF" ) {sum+=$15;nrec+=1;array[nrec
               ]=$15}} END {for(x=1;x<=nrec;x++)
               sumsq+=(array[x]-(sum/nrec))^2 ;
               print sqrt(sumsq/nrec)}'`
            sfo=`cat $2 | awk -F ',' '{if ($17 == "
               SFO" ) {sum+=$15;nrec+=1;array[nrec
               ]=$15}} END {for(x=1;x<=nrec;x++)
               sumsq+=(array[x]-(sum/nrec))^2 ;
               print sqrt(sumsq/nrec)}'`
            echo "lax" $lax
            echo "oak" $oak
            echo "smf" $smf
            echo "sfo" $sfo
            ;;
esac
```