MUSC
COLLEGE of GRADUATE STUDIES

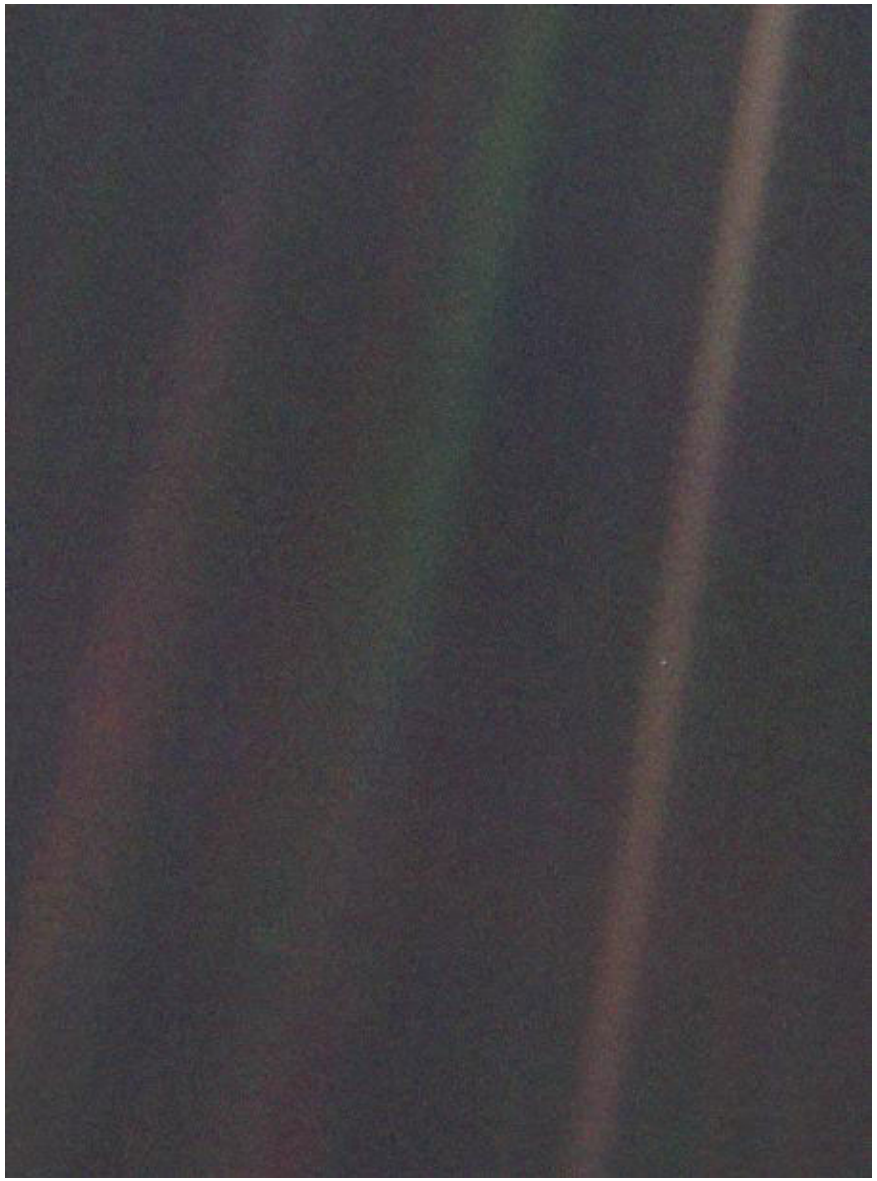
Spring 2013 BMTRY 789-02

Parallel Processing in R

Adrian Michael Nida

DPHS

April 10, 2013



MUSC
COLLEGE of GRADUATE STUDIES

Outline of Talk

- Introduction
- Cluster
- Parallel Processing

"The time has come," the Walrus said, "To talk of many things:..."

– Lewis Carroll *Through the Looking-Glass and What Alice Found There*

MUSC
COLLEGE of GRADUATE STUDIES

Introduction

- UNIX != Windows
- History
- Executable Syntax
- Common Commands
- Editing Files
- Secure Shell (ssh)
- Source Control (optional)

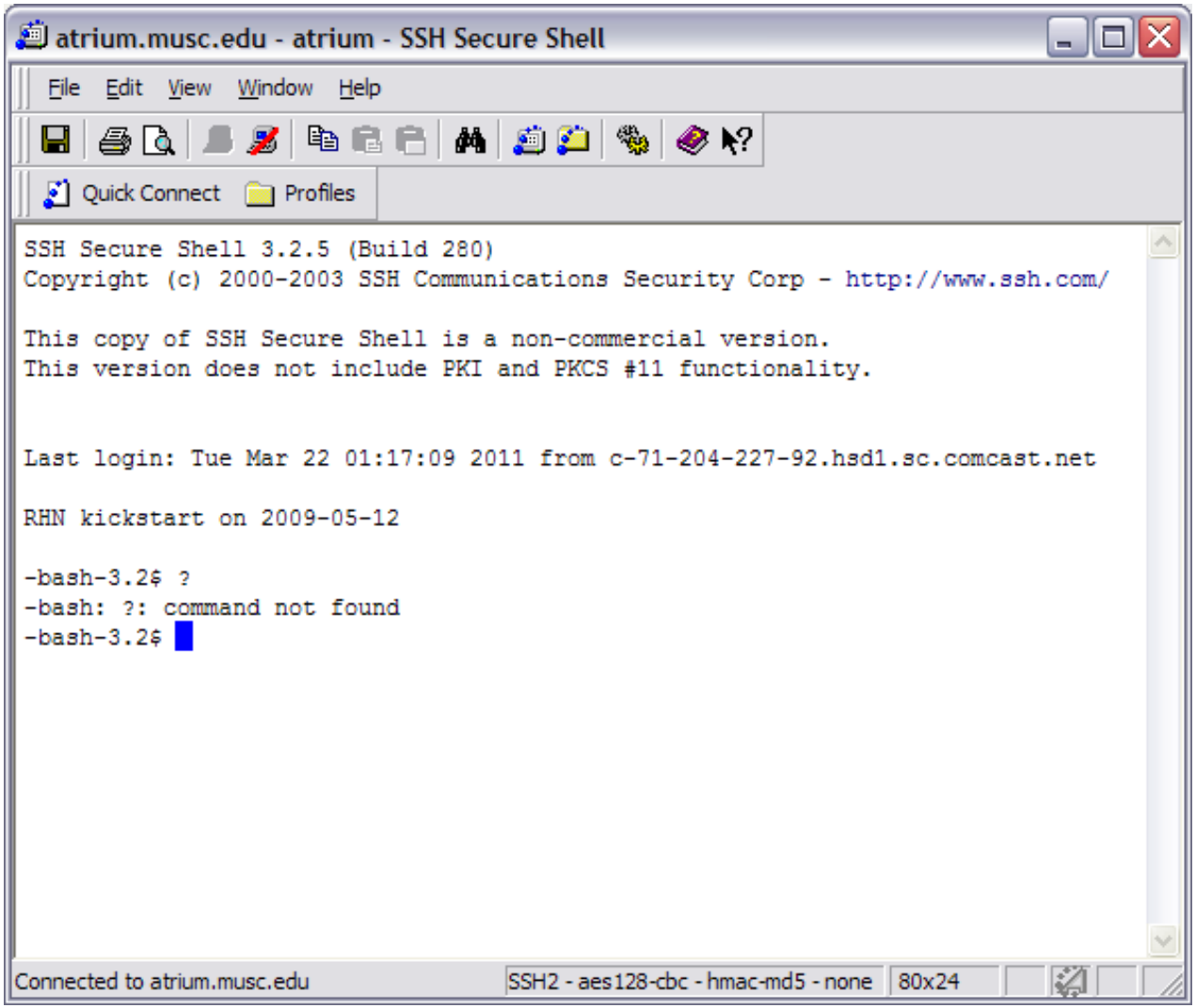
"Sure, Unix is a user-friendly operating system. It's just picky with whom it chooses to be friends."

– Ken Thompson

MUSC
COLLEGE of GRADUATE STUDIES

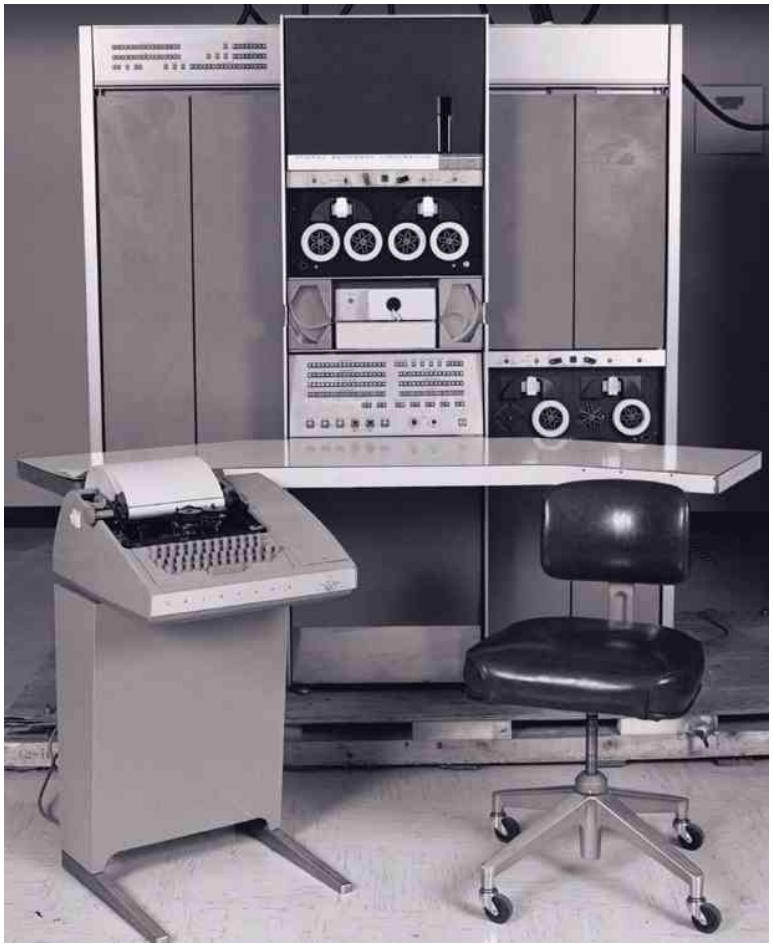
UNIX != Windows

UNIX != Windows (cont.)



MUSC
COLLEGE of GRADUATE STUDIES

A History of UNIX



The history

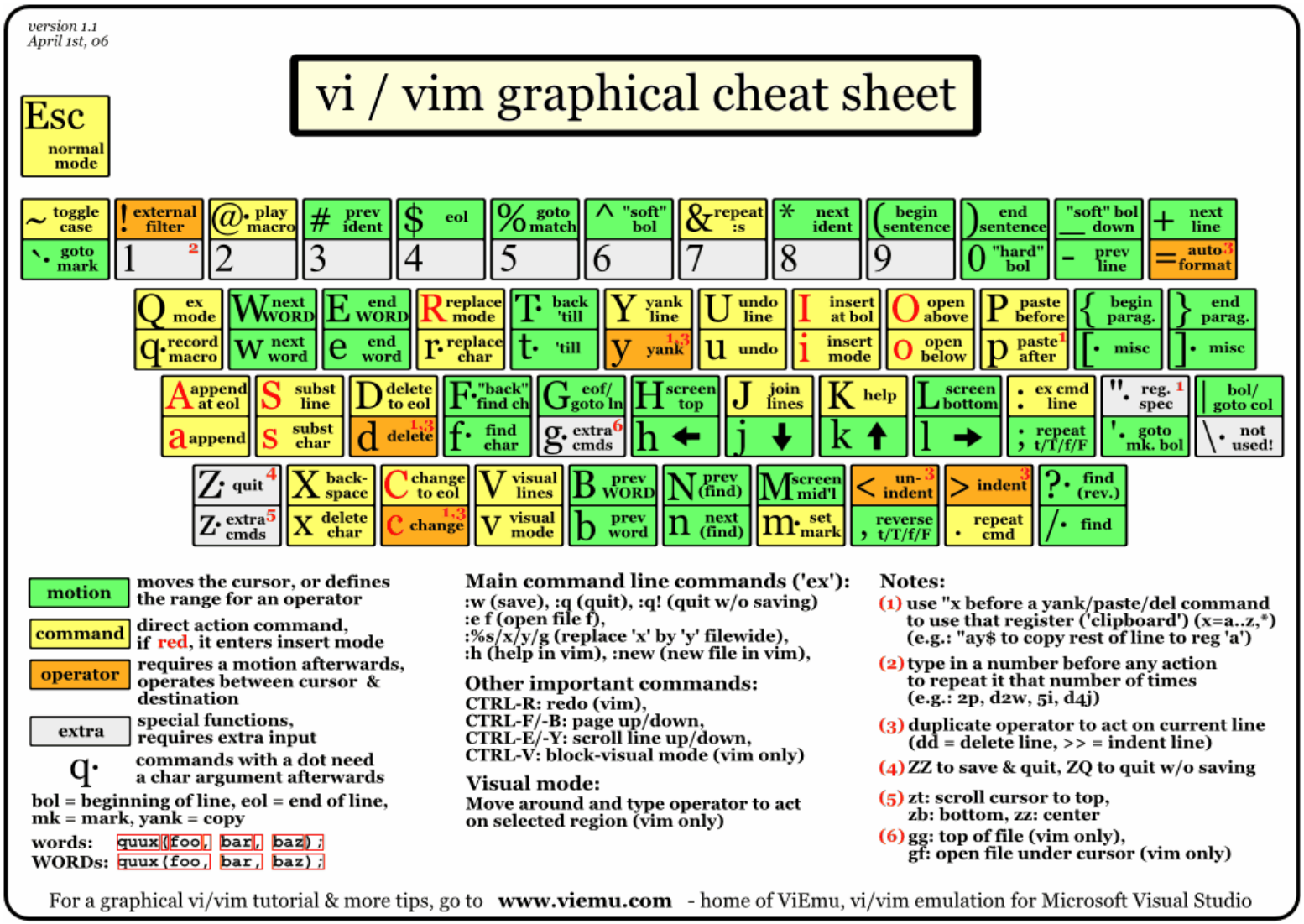
MUSC
COLLEGE of GRADUATE STUDIES

Executable Syntax

‘/path/to/program [options] [files]‘
where:

- program is the name of the program you wish to rum
  - /path/to is used to specify where on the filesystem program is located (Hint: If this location is in your $PATH, you won't need to type it) (Another Hint: The current directory '.' is NOT in your path, so to execute things there you must type './program')
- options are "switches" passed into the program to alter its code flow.
  - They can start with '-', '- -', or nothing at all.
- files are the files your program requires to run. This can be none at all.

man [program]	Displays help for a command (try 'man man', 'man hier')
cd [directory]	Change to directory
mkdir [newdir]	Make a directory named newdir in the current directory
ls [-lha] [directory]	List contents of directory
cp [-ra] SOURCE DEST	copy SOURCE to DEST
mv SOURCE DEST	copy and then delete SOURCE to DEST
rm [-rf] file(s)	REMOVE file(s)
chmod [-R] ugo file	Change mode (permissions) of a file (x=1, w=2, r=4)
chown [-R] owner:group file	Change Owner (and group)
find [directory] -option PATTERN	Search for files matching option's PATTERN
head | tail [-n lines] [file]	print first | last lines of file
grep [-inrv] PATTERN file(s)	Search for pattern in file(s)
sed [-i] 's/FIND/REPLACE/[g]' [file]	find & replace in 'stream'
awk 'FS=":"print $1, $6' [file]	print 1st & 6th fields of file
exit	terminate CLI session
$\sim$ | > >> 2& > 1	Home, piping, and STD[IO|ERR] redirection



Taken from: VIemu

MUSC
COLLEGE of GRADUATE STUDIES

Secure Shell (ssh)

- To connect to another computer, you will need to use this program from the OpenSSL group.

ssh [-1246AaCfgkMNnqsTtVvXxY] [-b bind_address] [-c cipher_spec] [-D [bind_address:]port] [-e escape_char] [-F configfile] [-i identity_file] [-L [bind_address:]port:host:hostport] [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port] [-R [bind_address:]port:host:hostport] [-S ctl_path] [-w tunnel:tunnel] [user@]hostname [command]

- There are Windows alternatives
  - PuTTY
  - SSH Secure Shell (^{TM})

MUSC
COLLEGE of GRADUATE STUDIES

Source Control

- When working between many computers, you will eventually have to organize your documents so changes get passed correctly.
- Source Control allows one to "check [in|out]" versions of documents in ways that allow a revisionist history.
- Subversion is the SCM used by the department formally known as DBBE:
  - svn co https://projects.dbbe.musc.edu/nida/School/
  - svn status
  - svn up
  - Make Changes
  - svn diff
  - svn add [file]
  - svn ci -m 'Message'
- http://tortoisesvn.tigris.org/ is a well received Windows client.
- If you want an account, SPEAK UP

MUSC
COLLEGE of GRADUATE STUDIES

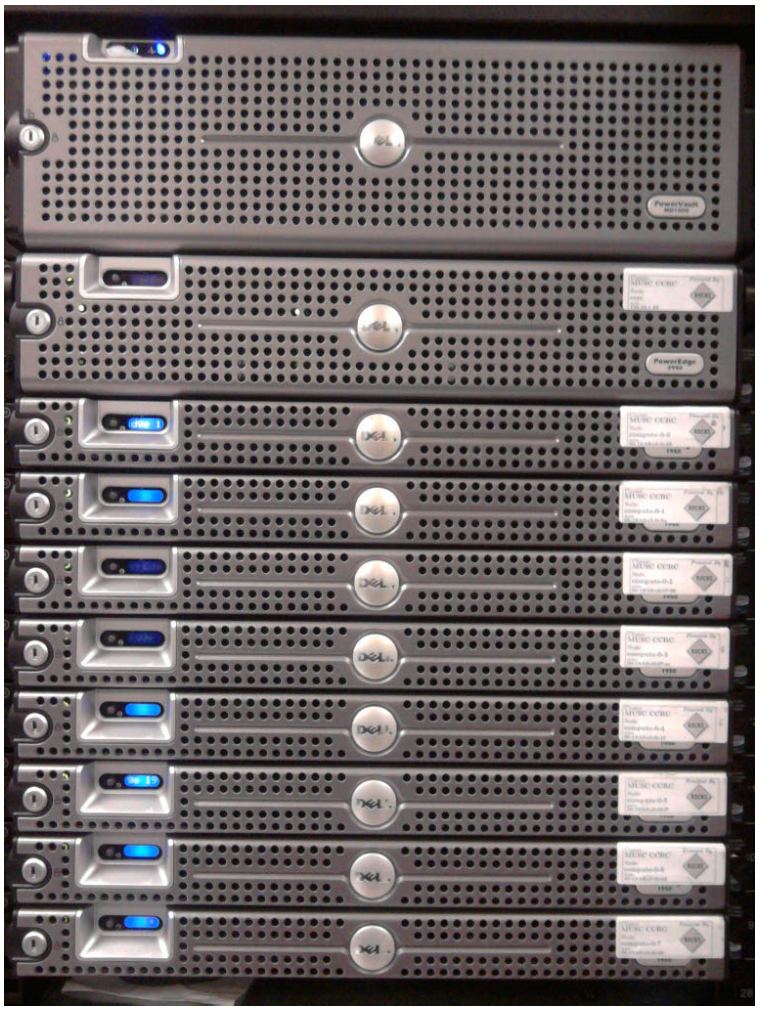
Cluster

- Hardware capabilities
- User Accounts
- Environment

"Imagine a Beowulf cluster of these!"

– Anonymous (Coward) Slashdot Troll

Hardware capabilities



The Cluster's Homepage

MUSC
COLLEGE of GRADUATE STUDIES

User Accounts

- Accounts (should) have been created for all of you
- Synched with University's Lightweight Directory Access Protocol (i.e., same NetID/Password combo you already know)
- Very few have the keys to the kingdom (i.e., sudo access)

MUSC
COLLEGE of GRADUATE STUDIES

Environment

/export (this is on the head node. This is mounted as /share/ from all nodes)

- apps
  - ...
  - R
    - R-2.1.0
    - R-2.10.1
    - R-2.12.2
    - R-2.13.0
    - R-2.8.1
  - resources
  - ...
- bio
  - hmmer
  - ncbi

MUSC
COLLEGE of GRADUATE STUDIES

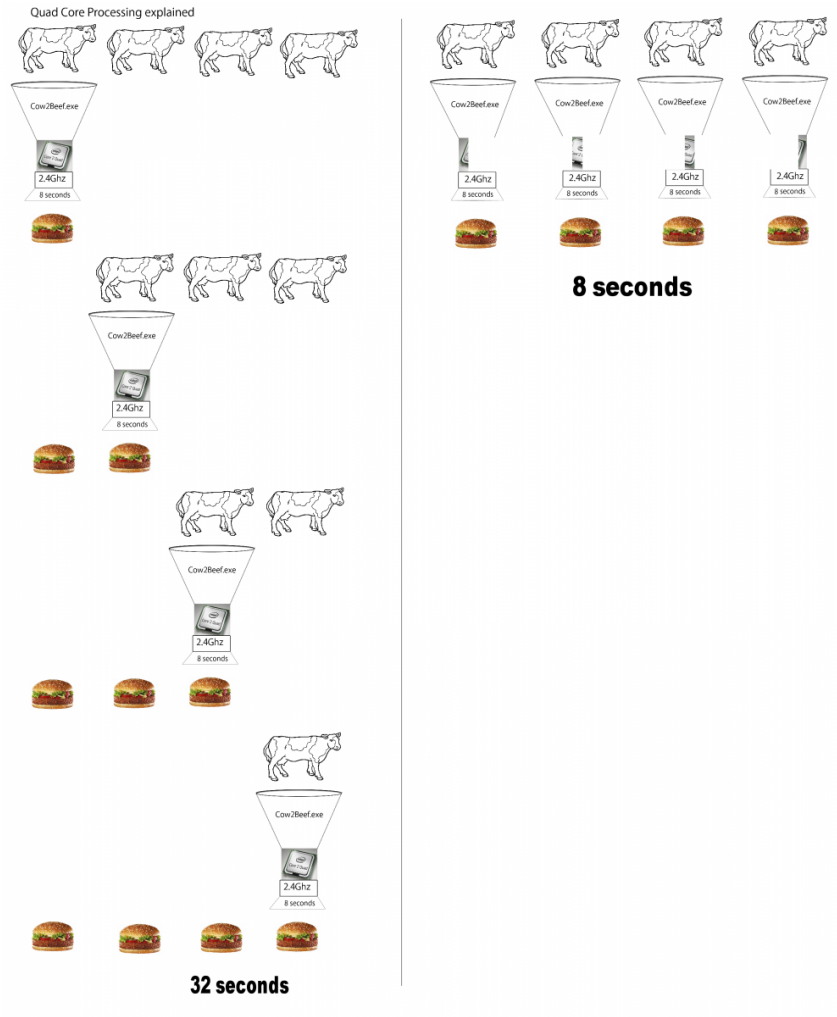
Parallel Processing

- Advantages
- Problems
- The two types

"There are 3 rules to follow when parallelizing large codes. Unfortunately, no one knows what these rules are."

– W. Somerset Maugham, Gary Montry

Advantages



Author Unknown

MUSC
COLLEGE of GRADUATE STUDIES

Problems

- Hard to implement
  - Critical Regions
  - Race Conditions
- Knowing what you can parallelize.

MUSC
COLLEGE of GRADUATE STUDIES

Two Types

- Batch Programming
- Truly Parallel

TIMTOWTDI

MUSC
COLLEGE of GRADUATE STUDIES

Two Types

- Batch Programming
- Truly Parallel

TIMTOWTDIBSCINABTE

Batch Programming

```
R CMD BATCH [options] ["--args arg1 ..."] my_script.R [outfile]
```

where my_script.R is in the form:

```
args <- commandArgs(TRUE) #Specifies only trailing args
print(args) #Print args character vector
...
q(status=0) #Any other number signifies error
```

Bash Scripting Commands

Command	Description
qsub [script.sh]	Submit batch jobs
qsub -I	Submit an interactive job
qstat -u [userid]	Check status of all of userid jobs
qhold [jobID]	Put a job on hold (before it starts)
qrls [jobID]	Release a job from hold status
qdel [jobID]	Delete a job, running or not

MUSC
COLLEGE of GRADUATE STUDIES

Batch Script

Very simple example:

```
#!/bin/sh
#$ -N NameOfYourJob
#$ -M EmailAlias@musc.edu
#$ -m beas
#$ -S /bin/bash
#$ -V
#$ -cwd

cd /path/to/where/my_script/is

R CMD BATCH [options] ["--args arg1 ..."] my_script.R [outfile]
```

MUSC
COLLEGE of GRADUATE STUDIES

An Intro to Homework

- On the class website, you will find five files.
  - Assignment (the PDF of this portion of the talk)
  - Genome input file – 50000 'Chromosome' file with 3000 'nucleotides' / 'Chromosome' (144MB)
  - mineAminos.R (the single threaded version – shown on next slide)
  - mineAminos.batch.R (the batch script version of the above file)
  - create.batchfile.R (a program that will create the batch files you will need to process through the Sun Grid Engine)

MUSC
COLLEGE of GRADUATE STUDIES

mineAminos.R (single-threaded)

```
ChromosomeLength = 3000
genome <- scan("genome.txt", what=character(ChromosomeLength))
total <- length(genome)
AminoAcids <- list()
for (i in 1:total) {
        chromosome <- genome[i]
        for(j in seq(1, ChromosomeLength, 3)) {
                amino <- substr(chromosome, j, j+2)
                if (!is.null(AminoAcids[[amino]])) {
                        numAminos <- AminoAcids[[amino]]
                        AminoAcids[[amino]] <- (1 + as.integer(numAminos))
                } else {
                        AminoAcids[[amino]] <- 1
                }
        }
}
Names <- sort(names(AminoAcids))
for (i in 1:length(Names)) {
        cat(Names[i], paste(AminoAcids[[Names[i]]], "\n", sep=''), sep="\t")
}
print(proc.time()[3])
```

MUSC
COLLEGE of GRADUATE STUDIES

Output

```
> source("mineAminos.R")
Read 50000 items
aaa     780293
aac     781510
aag     781449
aat     779933
aca     779984
...
ttc     781373
ttg     780609
ttt     782149
 elapsed
2017.413
```

MUSC
COLLEGE of GRADUATE STUDIES

mineAminos.batch.R

```
ChromosomeLength = 3000
genome <- scan("genome.txt", what=character(ChromosomeLength))
total <- length(genome)
AminoAcids <- list()


Args <- commandArgs(TRUE)
Beginning <- as.integer(Args[1])
Ending <- as.integer(Args[2])
for (i in Beginning:Ending) {


        chromosome <- genome[i]
        for(j in seq(1, ChromosomeLength, 3)) {
                amino <- substr(chromosome, j, j+2)
                if (!is.null(AminoAcids[[amino]])) {
                        numAminos <- AminoAcids[[amino]]
                        AminoAcids[[amino]] <- (1 + as.integer(numAminos))
                } else {
                        AminoAcids[[amino]] <- 1
                }
        }
}
Names <- sort(names(AminoAcids))
for (i in 1:length(Names)) {
        cat(Names[i], paste(AminoAcids[[Names[i]]], "\n", sep=''), sep="\t")
}
print(proc.time()[3])
```

MUSC
COLLEGE of GRADUATE STUDIES

create.batchfile.R

Feel free to review this file. It is not coded efficiently, but it gets the job done (or does it?). This is an example of how you should run it:

```
R CMD BATCH --vanilla --slave '--args $NumSlaves $Name $EmailAlias' create.batchfile.R
```

You will have to run it with at least three different NumSlaves so you can compare the times to the single threaded version. You will also have to sum the outputs from each run to compare them to the single-threaded version.

Let's try it ...

MUSC
COLLEGE of GRADUATE STUDIES

Extra Credit!

library("Rmpi")

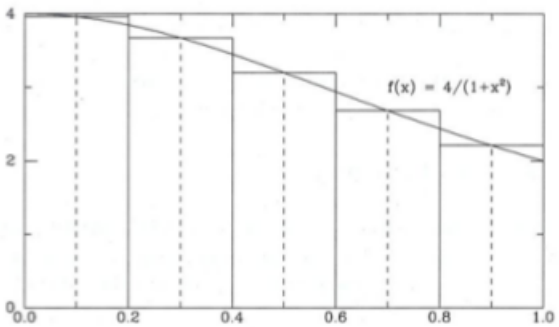
```
# Load the R MPI package if it is not already loaded.
if (!is.loaded("mpi_initialize")) {
    library("Rmpi")
    }

# Spawn as many slaves as possible
mpi.spawn.Rslaves()

# In case R exits unexpectedly, have it automatically clean up
# resources taken up by Rmpi (slaves, memory, etc...)
.Last <- function(){
    if (is.loaded("mpi_initialize")){
        if (mpi.comm.size(1) > 0){
            print("Please use mpi.close.Rslaves() to close slaves.")
            mpi.close.Rslaves()
        }
        print("Please use mpi.quit() to quit R")
        .Call("mpi_finalize")
    }
}

# Tell all slaves to return a message identifying themselves
Result <- mpi.remote.exec(paste(mpi.get.processor.name(),"is",mpi.comm.rank(),"of",mpi.comm.size()))
print(Result)

# Tell all slaves to close down, and exit the program
mpi.close.Rslaves()
mpi.quit(save="no")
```

CLEMSON
COMPUTING AND INFORMATION TECHNOLOGY

Example: Parallelizing Code

Calculating pi by numerical integration:

$$\int_0^1 \frac{1}{1+x^2}\,dx = \arctan(1) - \arctan(0) = \arctan(1) = \frac{\pi}{4}$$



$$\pi \approx \left(\sum_{n=1}^{\#\,intervals} \frac{4}{(1+x^2)} \right) \Big/ \#\,intervals$$

where $x = \dfrac{interval - 0.5}{\#\,intervals}$

Start with the serial version:

`pi.R`

Galen Collier (galen@clemson.edu)

```
intervals <- as.integer(readline("Please enter the number of intervals: "))

computeInterval <- function(intervals) {
        ysum <- 0.0;
        for (i in 1:intervals) {
            xi <- (1.0/intervals)*(i+0.5)
            ysum <- ysum + 4.0/(1.0+xi*xi)
        }
        myarea <- ysum*(1.0/intervals)
        return(myarea)
}

Result <- computeInterval(intervals)
print(paste("Area is", Result))
```

```
if (!is.loaded("mpi_initialize")) {     #Added
    library("Rmpi")                     #Added
}                                       #Added

mpi.spawn.Rslaves()                     #Added
intervals <- as.integer(readline("Please enter the number of intervals: "))

computeInterval <- function(intervals) {
        rank <- mpi.comm.rank()         #Added
        size <- mpi.comm.size()         #Added
        size <- size - 1                #Added WHY IS THIS NEEDED?
        ysum <- 0.0;
        for (i in seq(rank, intervals, by=size)) {
            xi <- (1.0/intervals)*(i+0.5)
            ysum <- ysum + 4.0/(1.0+xi*xi)
        }
        myarea <- ysum*(1.0/intervals)
        return(myarea)
}

mpi.bcast.Robj2slave(intervals)         #Added
mpi.bcast.Robj2slave(computeInterval)   #Added
#Changed
Result <- mpi.remote.exec(computeInterval(intervals))

area <- apply(Result, 1, sum)           #Added
print(paste("Area is", area))           #Changed (slightly)

mpi.close.Rslaves()                     #Added
mpi.quit(save="no")                     #Added
```

MUSC
COLLEGE of GRADUATE STUDIES

Homework (cont.)

- Now that you've seen examples, you have to apply what you have learned to the homework. You are asked to take the single threaded version of mineAminos and convert it to an Rmpi version.
- Hints:
  - Run a different 'Chromosome' on a different slave. (Compare 'i' to 'rank')
  - The results returned by mpi.remote.exec will be a 'list-of-lists' use as.matrix(as.numeric(Results[i])) to convert to matrix columns
  - Get started early!
- GOOD LUCK!

MUSC
COLLEGE of GRADUATE STUDIES

Final Thoughts

We're just getting started!

Hadoop!

Do you have a question(s)?