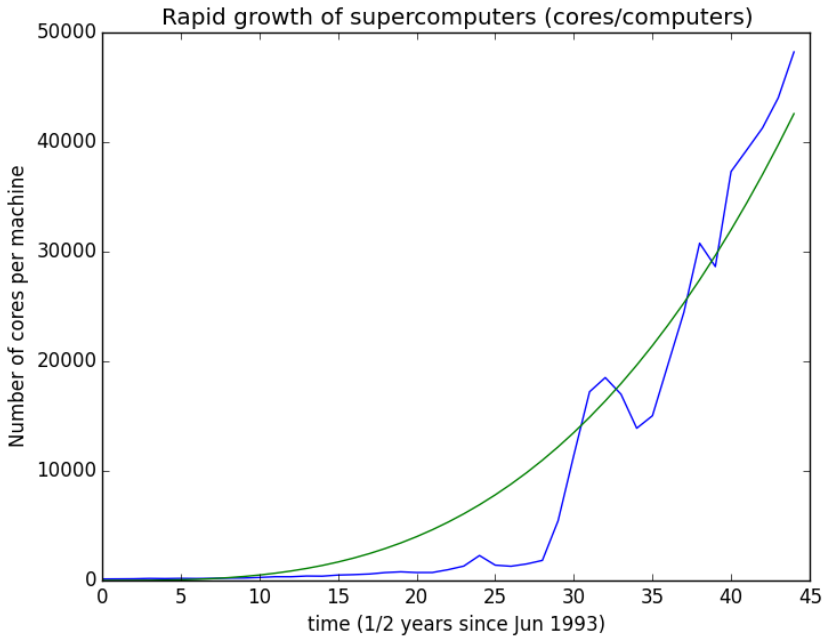
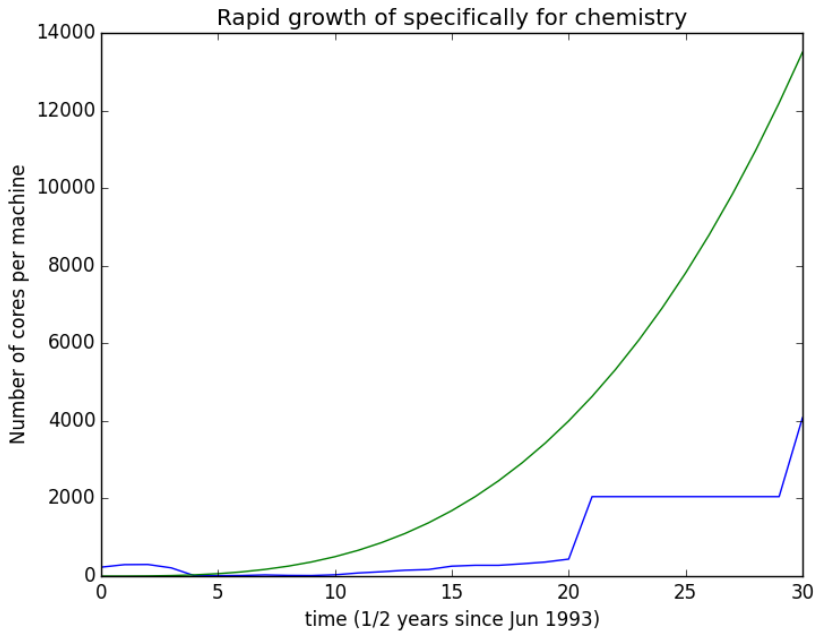
Jeremy First
12 October 2015

Assignment 0

Exercise 0.0

Using the top500.org website, I compared the number of cores per machine in unspecified supercomputers vs. chemistry-specific supercomputers. These plots are shown below. A simple fit of y = 1/2 x^3 in green is provided to guide your eye.





Of course, it is well-known already that computers are getting fatter (more nodes per computer). However, chemistry specific super clusters are not growing nearly as fast generalized super clusters. The following code was used to compute the graphs:

```
 1 #!/usr/bin/env python
 2
 3 import numpy as np
 4 import matplotlib.pyplot as plt
 5
 6
 7 for file in 'compute', 'chem_compute' :
 8
 9     data = np.genfromtxt(file+'.csv',delimiter=',')
10
11     x = range(0,len(data))
12
13     numcores = data[:,4] / data[:,0]
14     datachem =
np.genfromtxt('chem_compute.csv',delimiter=',')
15
16     plt.plot(x,numcores,label='Number of cores')
17     plt.plot(x, 0.50*np.power(x,3),label='Simple fit y = 1/2
x^3')
18     plt.xlabel('time (1/2 years since Jun 1993)')
19     plt.ylabel('Number of cores per machine')
20     if file == 'compute' :
21         plt.title('Rapid growth of supercomputers
(cores/computers)')
22     elif file == 'chem_compute' :
```

```
 23          plt.title('Rapid growth of specifically for
chemistry')
```

Exercise 0.1

Other than the obvious reason that there are more processors/socket and more nodes at Stampede, Stampede is faster than Lonestar even with a slower clock speed because the Xeon E5 processors are much more efficient than the older Westmere processors. This means that they can vectorize and parallelize operations better to be more computationally efficient. Additionally, the Westmere processor has 12M of cache on the chip, while the Xeon E5 has 20M. More space in cache means less time waiting on data to arrive at the processor, leading to faster computation. The clock-speed of a processor is not representative of its "calculation speed" since modern microarchitectures are doing more work with less cycles.

Exercise 0.2

There are 6400 nodes at Stampede, each with 2 Xeon E5-2680 processors (at 2.7 GHz) and 1 Xeon Phi MIC Coprocessor (at 1.1 GHz). Each Xeon E5 has 8 cores that run 8 flops/cycle, and the Xeon Phi has 61 cores that run 16 flops/cycle.

8 Flops/cp * 2.7 Gcp/s * 2 sockets/node * 8 processors/socket * 6400 nodes = 2.212 PFlops/s from the Xeon E5s and
8 Flops/cp * 1.1 Gcp/s * 1 socket/node * 61 processors/socket 6400 node = 6.87104 PFlops/s from the Xeon Phi Coprossesors.

That leads to a total of 9.083 PFlops/s on Stampede.

Exercise 0.3

The command "genent passwd | cut –f 7 –d: | sort | uniq" is a workflow of several command put together.

First, the command "genent passwd" gets the entries of passwd and prints them to standard output. The file passwd contains an entry for every TACC user containing the user name, user number, preferred name, path to home directory, preferred default shell, and some other information. All of this is printed to standard output.

The "|" pipe takes this standard output and redirects it as standard input to the next command "cut," which cuts each line. The option "-f 7" selects only the seventh field to print, and the option "-d:" sets the delimiter of these fields to a colon. Therefore, only the preferred default shells of every entry of passwd are passed to standard output.

Next, this output is piped to the standard input of "sort," which sorts the inputs by line. By default, "sort" sorts the data alphanumerically and prints the data to standard output. This standard output is now a sorted list of all of the default shells of the TACC users.

Finally, this output is piped to "uniq," which takes the standard input, removes any duplicate lines, and prints to standard output. This is now a list of unique shells used amongst all the TACC users. With no remaining pipes or commands, the standard output is printed to the screen.

Exercise 0.4

7590 users use the bash shell on TACC. This number was found by using the command:

```
login3.stampede(25)$ getent passwd | grep /bin[_/]bash | wc -l
16426
```

86 users use the tcsh shell on TACC. This numer was generated using the command:

```
login3.stampede(26)$ getent passwd | grep /bin[_/]tcsh | wc -l
202
```

The next most popular shell on TACC is the csh, which has 38 users. This was generated via the following commands (note that /bin/csh and /bin_csh are assumed to be the same).

```
login1.stampede(32)$ shells=$(getent passwd | cut -f 7 -d: |
sort | uniq) ; for shell in $shells ; do printf "$shell:  " ;
getent passwd | grep $shell | wc -l ; done
/bin_bash:  8825
/bin/bash:  7601
/bin_csh:  13
/bin/csh:  25
/bin/sync:  1
/bin_tcsh:  116
/bin/tcsh:  86
/bin_zsh:  9
/bin/zsh:  18
/sbin/halt:  1
/sbin/nologin:  29
/sbin/shutdown:  1
```

Exercise 0.5

There are two TACC users with usernames that are 9 characters. They are haldaemon and nfsnobody. This was generated using the following command:

```
login4.stampede(16)$ getent passwd | cut -f 1 -d: | awk '{ print
length " " $0 }' | sort | tail
8 zongsian
8 zsimpson
8 zstratto
8 zterrano
8 zubatyuk
8 zxiaohan
8 zyl65537
8 zzhang52
```

```
9 haldaemon
9 nfsnobody
```

There are 7707 users with exactly 8 characters in their username. This was generated by the following command:

```
login2.stampede(15)$ getent passwd | cut -f 1 -d: | awk '{ print
length }' | grep 8 | wc -l
7707
```

Exercise 0.6

Own question.

How many TACC users share your same group number? This can be generated by the following command:

```
login2.stampede(41)$ id=$(getent passwd | grep jfirst | awk -F
':' '{print $4}') ; getent passwd | grep $id | wc -l
75
```

Exercise 0.7

1. The number of words was counted that begin with each letter of the alphabet and the results were printed using the following command.

```
login4.stampede(12)$ for letter in {a..z} ; do printf $letter: ;
cat dict.txt | grep ^$letter | wc -l ; done
a:25192
b:19786
c:33264
d:19800
e:14292
f:13955
g:11544
h:15489
i:13719
j:2782
k:3857
l:10904
m:20156
n:14200
o:13411
p:36429
q:2980
r:18783
s:45065
t:21825
u:23021
v:5642
```

```
w:10050
x:453
y:1320
z:1270
```

2. 85316 words end with the letter ‘s’. This number was generated with the following command:

```
login4.stampede(16)$ cat dict.txt | grep s$ | wc -l
85316
```

Of these words, none of them are possessive. Command ;

```
login4.stampede(55)$ cat dict.txt | grep \'s$ | wc -l
0
```

3. A one-liner that matches foo but not foot, fool or food :
I tried to do this in one call to grep and came up with:

```
login4.stampede(32)$ cat dict.txt | grep -E foo[^dtl]
```

This however leaves out the word ‘foo’. So instead, here is the solution using two calls to grep:

```
login4.stampede(133)$ cat dict.txt | grep foo | grep -v foo[tdl]
Amersfoort
archbuffoon
buffoon
buffooneries
buffoonery
buffoonesque
buffoonish
buffoonishness
buffoonism
buffoons
Chefoo
Dryfoos
foo
foobar
foofaraw
foofaraws
foo-foo
fooner
fooster
foosterer
fooyoung
fooyung
foozle
foozled
foozler
foozlers
foozles
foozling
```

```
Furfooz
Furfooz-grenelle
mafoo
samfoo
tuffoon
```

4. Own Question.

Generate a one liner to find out how many words in the dictionary have two, three, four, five, six, or seven vowels in a row. What is word with the most number of vowels in a row? Don't match uppercase vowels, because 'AAAAAA' is boring.

```
login4.stampede(1)$ for num in {2..7} ; do printf "$num: " ; cat
dict.txt | grep -E [aeouiAEOUI]{$num} | wc -l ; done
2: 179408
3: 9924
4: 302
5: 13
6: 2
7: 0
login4.stampede(2)$ cat dict.txt | grep -E [aoeui]{6}
euouae
```

Exercise 0.8

The three file systems on Stampede are found in $SCRATCH, $WORK, and $HOME directories. There is alos /tmp and $ARCHIVE

Home : Default directory, backed up, quota limit of 5GB and 150K files. Intended to store source code and executables.
Work : Not backed up, quota limit of 1 TB, 3M files. Intended to have data stored for post-processing, etc.
Scratch: No backed up, no quoata, can be purged without warning if files have not been accessed in 10 days. Jobs should be run on this file system and then transferred to Work or Home.
/tmp : /tmp files are disks directly on the node (the previous file systems are not) . This makes I/O to /tmp extremely fast. The data however is wiped when you log out of the node. There is a quota of 80Gb.
Ranch : The archival storage Range is a tape storage facility to store long term data that you want to fully protect, but don't need to access. You cannot run anything on Ranch. There is no quota limit, and the data is backed up.

Exercise 0.9

1. Stampede's login nodes are running a Linux kernel, version 2.6.32-431.17.1.e16.x86_64. This information was generated with the following command:

```
login4.stampede(157)$ uname -a
Linux login4.stampede.tacc.utexas.edu 2.6.32-431.17.1.el6.x86_64
#1 SMP Wed May 7 23:32:49 UTC 2014 x86_64 x86_64 x86_64
```

```
GNU/Linux
```

2. The kernel was downloaded using the following command:

```
wget
"https://cdn.kernel.org/pub/linux/kernel/v2.6/longterm/v2.6.32/l
inux-2.6.32.68.tar.xz"
```

3. The kernel is 419Mb. Command:

```
login4.stampede(218)$ du -hs
419M    .
```

There are 30494 files in the kernel. Command:

```
login4.stampede(242)$ find . -type f | wc -l
30494
```

Of these, 13154 are .c files. Command:

```
login4.stampede(245)$ find . -type f -name *.c | wc -l
13154
```

4. There are 1878 subdirectories within the kernel source directory. Three commands were used to find this, ls, tree and find. Commands:

```
login4.stampede(377)$ tree -d --noreport | wc -l
1878
login4.stampede(378)$ find . -type d | wc -l
1878
login4.stampede(379)$ ls -R | grep : | wc -l
1878
```

5. The largest file (by size) is 875265 bytes. This file is 13947 lines in length. Commands:

```
login4.stampede(318)$ find . -type f -printf "%s %h/%f \n" |
sort -n | tail -n1
875265 ./fs/nls/nls_cp949.c
login4.stampede(319)$ cat fs/nls/nls_cp949.c | wc -l
13947
```

6. There 10946464 lines of code in the .c and .h files of this kernel. Command:

```
login4.stampede(343)$ find . -type f -name *.[ch] | xargs cat |
wc -l
10946464
```

7. How many of these lines are the start of a for-loop?

```
login4.stampede(14)$ find . -type f -name *.[ch] | xargs cat |
grep -E "for ?\(" | wc -l
45799
```