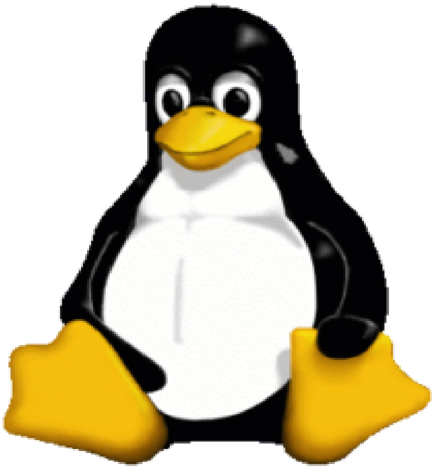
Large system usage HOW TO



George Magklaras PhD
Biotek/NCMM IT
USIT Research Computing Services

Agenda

- Introduction: A Linux server as a collection of memory/disk/CPU
- What is the problem?
- memory and SWAP space
- What are the limits on biotin?
- The 'htop' application
- An algorithm/procedure to help you not overload the system and get your work done
- Examples

Introduction

- Each server has a certain amount of CPU cores
  - `cat /proc/cpuinfo | grep processor | wc -l`
- A certain amount of RAM memory
  - `free`
- And a certain amount of disk space
  - `df -h`

Introduction (2)

```
[georgios@biotin ~]$ cat /proc/cpuinfo | grep processor | wc -l
32
[georgios@biotin ~]$ free
             total       used       free     shared    buffers     cached
Mem:     264514940  202499088   62015852          0    5482912  189410216
-/+ buffers/cache:    7605960  256908980
Swap:     10485752          0   10485752
[georgios@biotin ~]$ df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/internvg-root
                      2.0G  727M  1.2G  38% /
tmpfs                 127G     0  127G   0% /dev/shm
/dev/sda1             504M   99M  380M  21% /boot
/dev/mapper/internvg-opt
                       13G  9.7G  1.8G  85% /opt
/dev/mapper/internvg-tmp
                       63G  180M   60G   1% /tmp
/dev/mapper/internvg-usr
                       39G   23G   14G  63% /usr
/dev/mapper/internvg-var
                      7.9G  1.7G  5.9G  22% /var
/dev/mapper/VGpreben-LVpreben
                      5.4T  3.7T  1.5T  72% /storage/mortharea
/dev/mapper/VGmills-LVmills
                       13T  9.5T  2.5T  80% /storage/millsarea
/dev/mapper/VGEMBGalaxy-LVembgalaxy
                      7.0T  4.7T  2.4T  67% /storage/tools
/dev/mapper/VGscratch-LVhurtado
                      6.0T  4.1T  2.0T  68% /storage/hurtadoarea
/dev/mapper/VGscratch-LVscratch
                      8.6T  1.5T  7.2T  17% /storage/scratch
192.168.8.126:/storage/staerkarea/
                      3.0T  2.2T  884G  72% /storage/staerkarea
frigg.uio.no:/div/frigg/u6
                      500G  151G  350G  31% /div/frigg/u6
frigg.uio.no:/div/frigg/u7
                      500G  173G  327G  35% /div/frigg/u7
```

What is the problem?

- The most important resource in a shared server is memory.
- We cannot overuse memory, because things will start crashing, affecting people's work.
- If you overuse CPU cores or disk space, this can be to a certain extent tolerable. Not with memory.
- Memory exhaustion crashes have been caused twice in the past. We need to learn how to monitor for memory usage and prevent this from occurring.

Memory and SWAP space

- SWAP is hard disk space that is used to replace RAM.
- When the operating system kernel is running out of RAM (or it considers that certain programs are inactive), it transfers them to swap, until memory becomes available again.
- 'swap' kills performance: hard disk is much slower than RAM.
- When a system uses the SWAP space heavily, things are not optimal and it shows that memory is overused.

Example of heavy SWAP usage

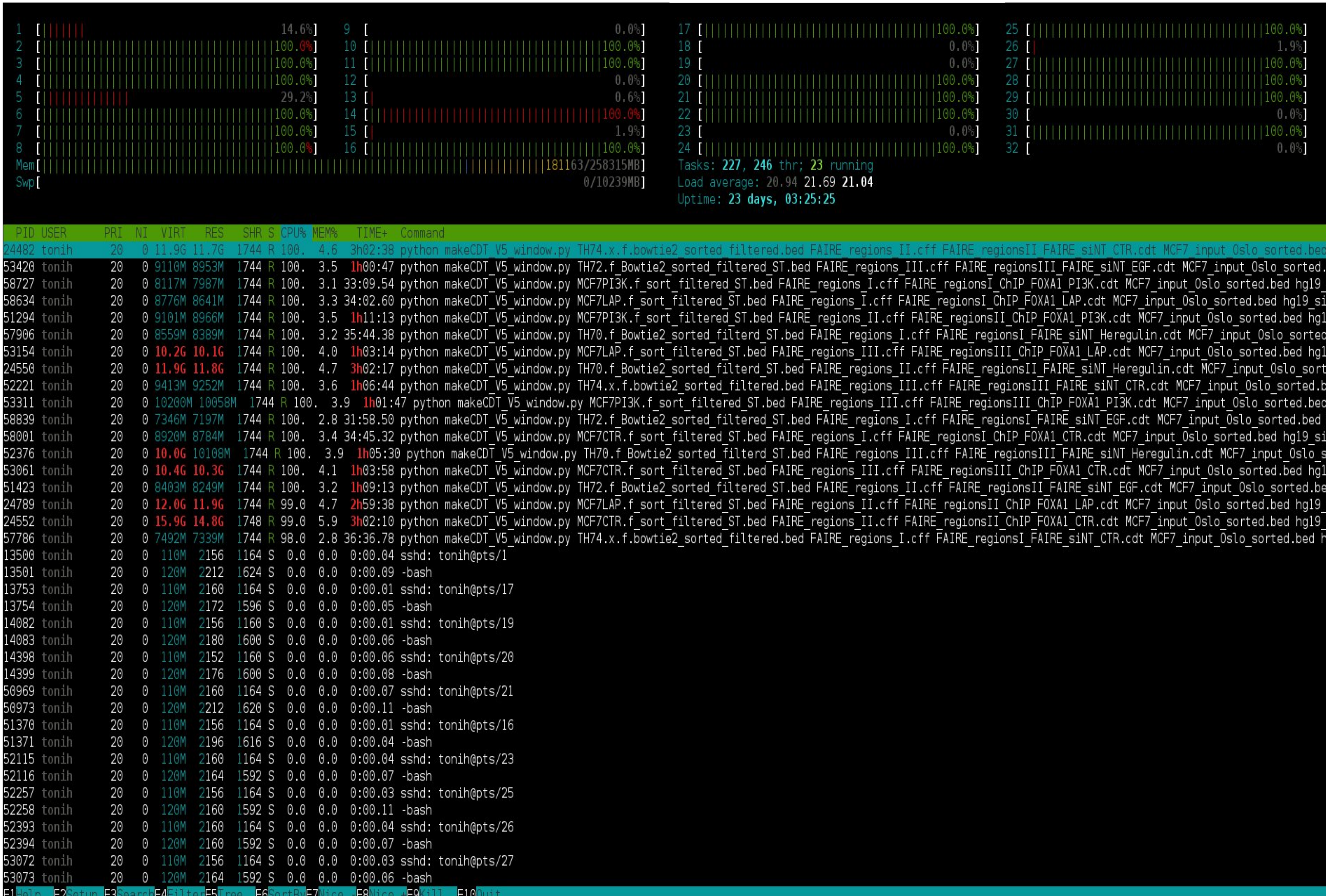
```
top - 13:41:43 up 46 days,  1:32,  3 users,  load average: 1.27, 1.16, 1.17
Tasks: 1772 total,   2 running, 1759 sleeping,  11 stopped,   0 zombie
Cpu(s): 10.4%us,  0.4%sy,  0.0%ni, 89.1%id,  0.1%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  529247244k total, 497745684k used, 31501560k free,    12780k buffers
Swap: 10485752k total,  5242608k used,  5243144k free, 187682484k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
50058 [redacted] 20   0  293g 288g 1724 R 96.2 57.2   3799:54 bogart
49642 root      20   0 20464 2496  884 R 10.5  0.0   0:00.10 top
    1 root      20   0 25684  792  532 S  0.0  0.0   0:53.80 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.60 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:39.75 migration/0
    4 root      20   0     0    0    0 S  0.0  0.0   1374:58 ksoftirqd/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:06.82 watchdog/0
    7 root      RT   0     0    0    0 S  0.0  0.0   0:08.32 migration/1
    8 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/1
    9 root      20   0     0    0    0 S  0.0  0.0 286:54.16 ksoftirqd/1
   10 root      RT   0     0    0    0 S  0.0  0.0   0:03.70 watchdog/1
   11 root      RT   0     0    0    0 S  0.0  0.0   0:04.69 migration/2
   12 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/2
   13 root      20   0     0    0    0 S  0.0  0.0 287:27.51 ksoftirqd/2
```

The limits on the biotin server

- Biotin has 256 Gigs of RAM memory.
- We can allocate 150 Gig to run for life science computing.
- The other 106 Gigs we need to run other critical services (VM cluster, file cache, etc)
- This means that amongst ALL of you, you should not exceed 150 Gigs of RAM.
- Cores and dedicated disk space we do not care. Memory we do care about. Limits have been enforced to ensure that failsafes function.

The 'htop' application



The 'htop' application (2)

- In the previous slide, if we add all the numbers under the 'RES' column for user 'tonih', we get approx 150 Gigs of RAM in total.
- And that was for a single user, which would be OK if nobody else was running on the system.
- Difference: `htop` versus `htop -u [username]`
- You have to account for others that might be executing programs at the same time as you.

An algorithm/procedure to regulate memory consumption

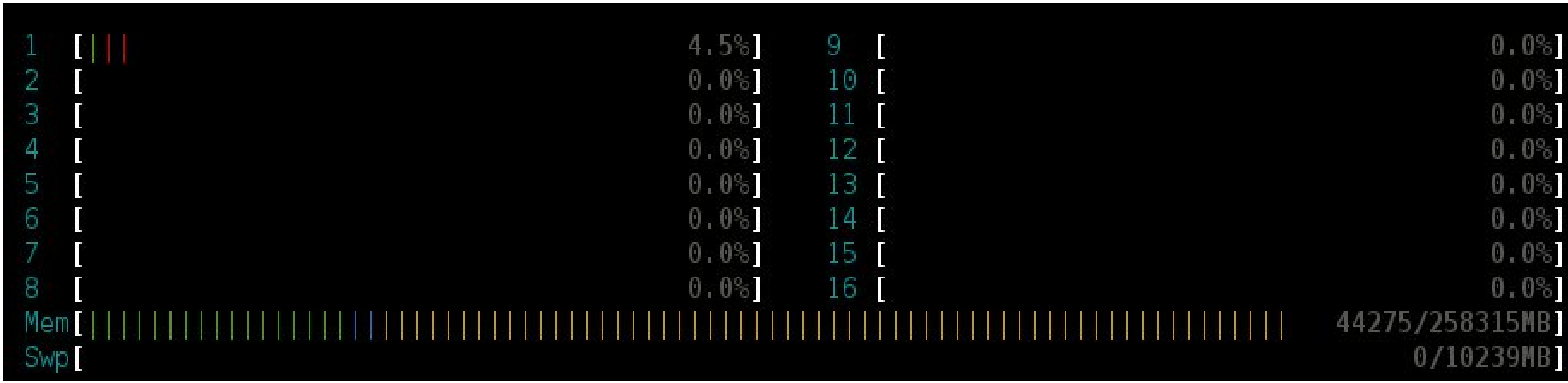
- **<u>Before</u>** you execute your heavy programs on biotin:
  - 1)Run `htop` and locate the `Mem` horizontal line which measures actual RAM consumption.
  - 2)If the amount of RAM **>= 150 Gigs (153600 MB)**, **please wait** until it comes <u>well under</u> 100 Gigs.
  - 3)If the amount of RAM < **150 Gigs(153600 MB)**, you can consume

    `150 Gigs – [current_consumption_by others]`

    for all your processes to share.

- How to gauge your current memory consumption:
  - Run '`htop -u [your_userid]`' and ADD UP all the figures under the `RES` column. This is how much you are currently using on the system.

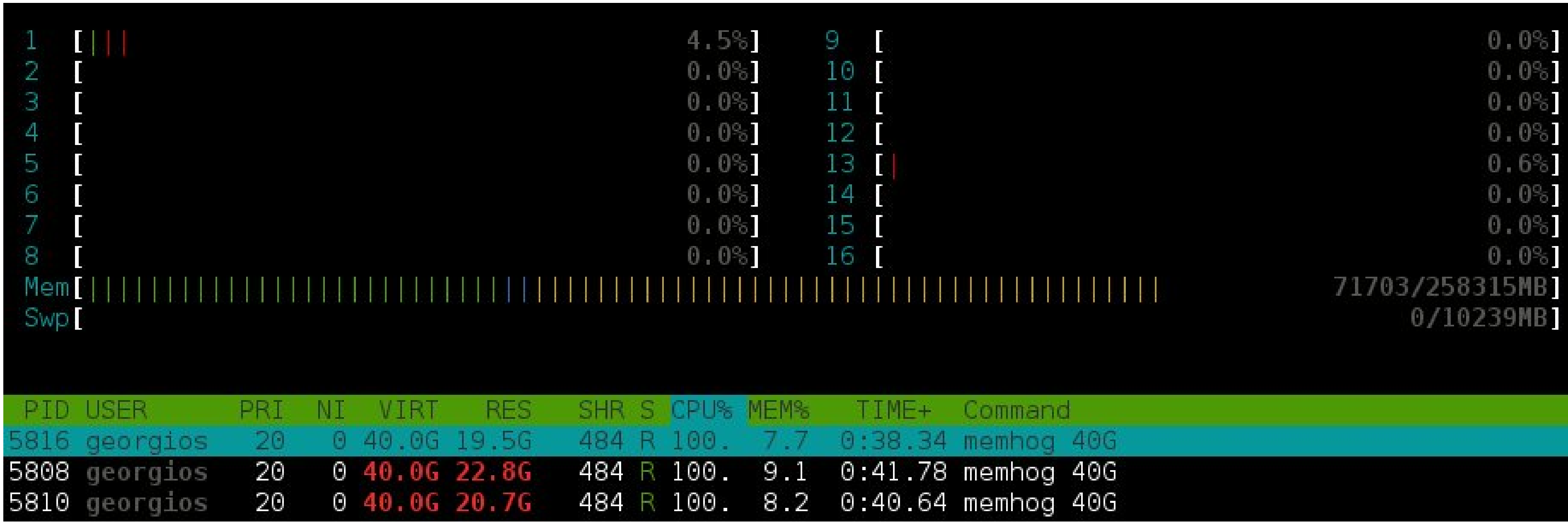
Example 1:



The current memory consumption on 'biotin' is approximately 44 Gigs (44275). How many 'novoalign' processes can I start at the same time, without overloading the system?

<u>Solution</u>: If I know that a single novoalign process can take up to 12 Gigs of RAM (depended on the parameters and my data set), I have 150 – 44 = 106 Gigabytes. Which means I can start a max of 8 of them (106/12 -> rounded to the lower integer).

Example 2:



The current memory consumption on 'biotin' is 70 Gigs (71703 MB). User georgios seems to be running these large processes that keep growing in size. How many Python make_CDT_window_V5.py scripts can I safely start without overloading the system.

<u>Solution</u>: If I know that a single script of that type can take up to 15 Gigs of RAM (depended on the parameters and my data set), I should have 150 – 70 = 80 Gigabytes. However, because I see user georgios acquiring RAM rapidly, I look at the VIRT column and I consider this as the maximum amount of RAM-> 120 Gigs (3 x 40 Gigs). Therefore, I have 150-120=30 Gigs, thus only 2 of these Python scripts could be started safely, given the current circumstances.

Questions?