COMPUTEROME 2.0

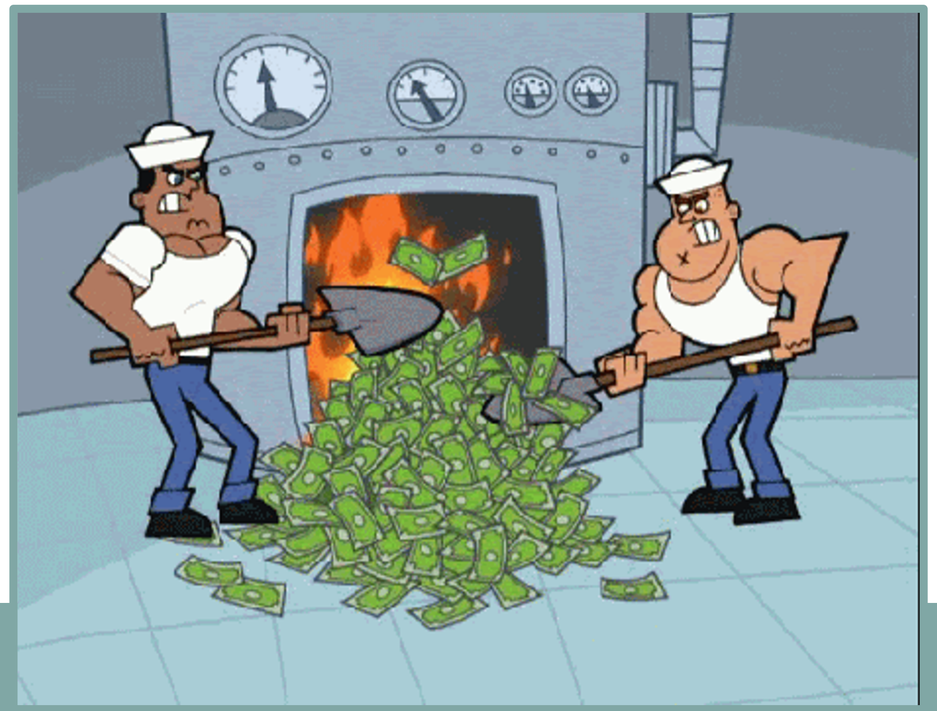
USERS WORKSHOP

CENTER FOR HEALTH DATA SCIENCE (HEADS)
FACULTY OF HEALTH AND MEDICAL SCIENCES,
UNIVERSITY OF COPENHAGEN, APRIL 2021

PART6

F IMIZING THE UTILIZATION OF C2: TIME, MEMORY AND COST

WHY IS IT IMPORTANT TO OPTIMIZE?

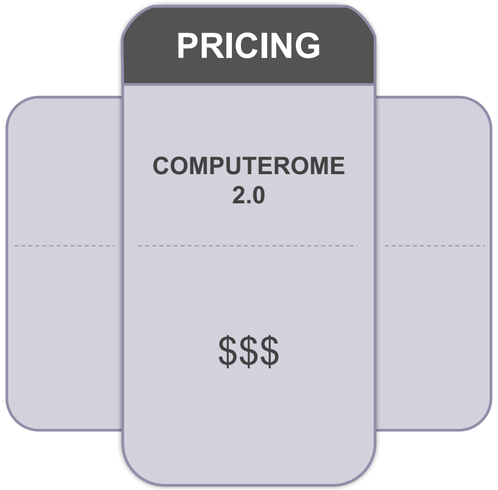


SAVING MONEY BY OPTIMIZATION

C2 pricing model: quantized **per node**

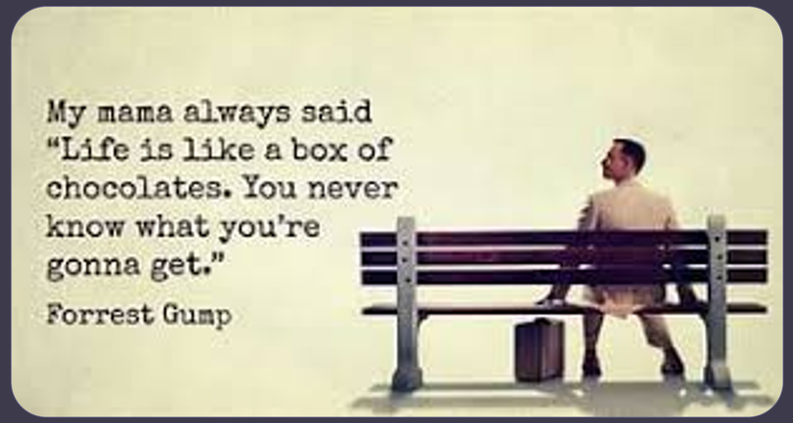
You pay per node, not how many cores or CPUs you used.

Remember C2 architecture – CPUs are distributed 40 per node



C2 is like a bar of chocolate

You pay per bar, now how many pieces you have eaten.

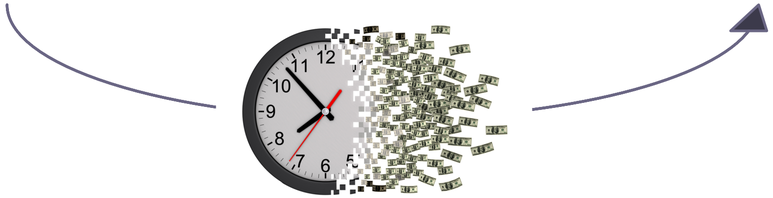


SAVING TIME BY OPTIMIZATION

PARALLELIZATION: more threads used usually means faster execution

not parallelized

parallelized



MEMORY OPTIMIZATION

How much memory does my job need?

Again, remember the C2 architecture

Thin compute node: 696 total thin nodes

40 cores (2CPUs with 20 cores each),
192 GB RAM

Since there are more thin node, the job is launched faster.

Fat compute node: 55 total fat nodes

40 cores (2 CPUs with 20 cores each),
1536 GB RAM (1.5 TB)

Only use the fat nodes if your job needs more than ~ 192 GB RAM.

CORE/CPU USAGE OPTIMIZATION

If your program is not multi-threaded (or cannot use the full 40 threads), run multiple commands in the background, thus utilizing CPU bandwidth.

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=120gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
bwa mem -t 10 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out &
wait
```

CORE/CPU USAGE OPTIMIZATION

If your program is not multi-threaded (or cannot use the full 40 threads), run multiple commands in the background, thus utilizing CPU bandwidth.

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=120gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
bwa mem -t 10 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out &
wait
```

CORE/CPU USAGE OPTIMIZATION

parallel command to run multiple commands together

GNU parallel - developed in Copenhagen by Ole Tange.

Allows replacement of simple loop constructs such as for loops.

Examples:

seq 1 80 | parallel -j 5 bwa mem -t 8 refgenome/canFam31.fasta Canid{}.R1.fastq.gz

OR
have a script with commands - commands.sh?

```
bwa mem -t 8 refgenome/canFam31.fasta Canid1.R1.fastq.gz > Canid1.sam
bwa mem -t 8 refgenome/canFam31.fasta Canid2.R1.fastq.gz > Canid2.sam
...
...
bwa mem -t 8 refgenome/canFam31.fasta Canid80.R1.fastq.gz > Canid80.sam
```

parallel -j 5 < commands.sh

https://www.gnu.org/software/parallel/parallel_tutorial.html

ESTIMATING RESOURCE REQUIREMENTS

How much time to execute a program?

Run a single job first to estimate time and use this as a guide to figure out how much time other similar jobs might require.

Use the `/usr/bin/time` command OR use `qstat`.

```
#!/bin/bash
#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=120gb,walltime=12:00:00

module load bwa/0.7.15

/usr/bin/time -v bwa mem -t 40 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz
> Batch1.out
```

ESTIMATING RESOURCE REQUIREMENTS

```
shygop@g-12-10002 ~
$ /usr/bin/time -v ls > testing
        Command being timed: "ls"
        User time (seconds): 0.00
        System time (seconds): 0.00
        Percent of CPU this job got: 40%
        Elapsed (wall clock) time (h:mm:ss or m:ss): 0:00.00
        Average shared text size (kbytes): 0
        Average unshared data size (kbytes): 0
        Average stack size (kbytes): 0
        Average total size (kbytes): 0
        Maximum resident set size (kbytes): 1016
        Average resident set size (kbytes): 0
        Major (requiring I/O) page faults: 0
        Minor (reclaiming a frame) page faults: 337
        Voluntary context switches: 3
        Involuntary context switches: 1
        Swaps: 0
        File system inputs: 0
        File system outputs: 8
        Socket messages sent: 0
        Socket messages received: 0
        Signals delivered: 0
        Page size (bytes): 4096
        Exit status: 0
shygop@g-12-10002 ~
$
```

IS THIS JOB OPTIMIZED TO LAUNCH ON COMPUTEROME? YES, NO, WHAT IS THE PROBLEM?

JOB 1

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=10,mem=120gb,walltime=12:00:00

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
bwa mem -t 10 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out &
wait
```

JOB 1

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=10,mem=120gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
bwa mem -t 10 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out &
wait
```

JOB 2

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=30,mem=120gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
wait
```

JOB 3

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=200gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
bwa mem -t 10 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out &
wait
```

JOB 4

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=48,mem=160gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 12 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 12 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 12 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
bwa mem -t 12 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out &
wait
```

JOB 5

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=120gb,walltime=12:00:00

bwa mem -t 40 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out
```

JOB 6

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=120gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out
bwa mem -t 10 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out
wait
```

JOB 7

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=150gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
bwa mem -t 10 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out &
wait
bwa mem -t 10 canFam31.fasta Batch5.R1.fastq.gz Batch5.R2.fastq.gz > Batch5.out &
bwa mem -t 10 canFam31.fasta Batch6.R1.fastq.gz Batch6.R2.fastq.gz > Batch6.out &
bwa mem -t 10 canFam31.fasta Batch7.R1.fastq.gz Batch7.R2.fastq.gz > Batch7.out &
bwa mem -t 10 canFam31.fasta Batch8.R1.fastq.gz Batch8.R2.fastq.gz > Batch8.out &
wait
```

JOB 8

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=500gb,walltime=60:00:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 14 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 16 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
wait
```

JOB 9

```
#!/bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=120gb,walltime=12:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 10 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 10 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
bwa mem -t 5 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out &
bwa mem -t 5 canFam31.fasta Batch5.R1.fastq.gz Batch5.R2.fastq.gz > Batch5.out &

wait
```

JOB 10

```
#!/bin/bash

#PBS -N test
#PBS -e test.err
#PBS -o test.log
#PBS -l nodes=1:ppn=40,mem=500gb,walltime=60:00:00:00

module load bwa/0.7.15

bwa mem -t 10 canFam31.fasta Batch1.R1.fastq.gz Batch1.R2.fastq.gz > Batch1.out &
bwa mem -t 14 canFam31.fasta Batch2.R1.fastq.gz Batch2.R2.fastq.gz > Batch2.out &
bwa mem -t 16 canFam31.fasta Batch3.R1.fastq.gz Batch3.R2.fastq.gz > Batch3.out &
wait
```

JOB 11!

```
#! /bin/bash

#PBS -W group_list=ku_fa -A ku_fa
#PBS -N canid1Map
#PBS -e canid1.err
#PBS -o canid1.log
#PBS -l nodes=1:ppn=40,mem=50gb,walltime=1:00:00
#PBS -d /home/projects/C2_test

### Load modules
module load bwa/0.7.15
module load samtools/1.9
module load htslib/1.9

### Run your jobs
bwa mem -t 40 refgenome/canFam31.fasta Canid1.R1.fastq.gz > Canid1.sam
samtools view -b Canid1.sam > Canid1.bam
samtools index Canid1.bam
```

Solutions

Job 2: Only using 30 cores - `#PBS -l nodes=1:ppn=30,mem=120gb,walltime=12:00:00`

Job 3: Do you need 200 Gb? - `#PBS -l nodes=1:ppn=40,mem=200gb,walltime=12:00:00`

Job 4: 48 cores? - `#PBS -l nodes=1:ppn=48,mem=160gb,walltime=12:00:00`

Job 5: Missing module load - bwa command not found.

Job 6: Missing “&” at the end of the commands -

```
bwa mem -t 10 canFam31.fasta Batch4.R1.fastq.gz Batch4.R2.fastq.gz > Batch4.out
wait
```

Job 7: All good!

Job 8: 500 Gb and 60 days? - do you need these resources

```
#PBS -l nodes=1:ppn=40,mem=500gb,walltime=60:00:00:00
```

Job 9: All good!

Job 10: Missing account information! Also check resource requirements.

Job 11: You tell me!


That's all Folks!