DIT065 - Computational Techniques for Large-scale Data Assignment 1

Huw Fryer
Cem Mert Dalli
Stefano Ribes

March 27, 2022

1 Problem 1

The following is the list of commands executed to extract the desired information from the three available servers, Bayes, Shannon and Markov. The same commands have been executed on all servers. The complete and detailed list of output logs can be viewed in Appendix B.

```
echo "============================================================"
echo "Number of CPU cores"
echo "============================================================"
lscpu | egrep 'CPU\(s\)|Core|Socket|Thread'
echo "============================================================"
echo "CPU Type"
echo "============================================================"
lscpu | egrep "Model name| MHz"
echo "============================================================"
echo "Disk Amount"
echo "============================================================"
df -h --total
echo "============================================================"
echo "Login shell virtual memory usage (code+data+stack) in KB"
echo "============================================================"
ps -o vsz= -p "$$"
```

1.1 Bayes

There are 48 CPUs, which are listed between 0-47. There are 12 cores per socket and 2 threads per socket. Then, we move forward to see type of CPU and clock frequency. At the time that we write this report, CPU MHz was 1000.093, with the CPU max MHz capacity

up to 3700. The model is an *Intel(R) Xeon(R) Gold 6126 CPU @ 2.60GHz*. Out of 100TB disk capacity, 8.4TB is in use and approximately 90T is available. `/data` is the largest file system in Bayes having 73TB, which is followed by `/datainbackup` that has 24T capacity. While 17% of space is used in `/datainbackup` (4.0TB), only 6% of `/data` is used (4.3TB). Finally, in order to measure the virtual memory usage in KB (of combined code, data and stack) of the login shell, we utilized the `ps` command targeting the shell PID (which is stored in the `$$` variable). On the Bayes server, the virtual memory of the login shell takes 7024 KB.

1.2 Shannon and Markov

As confirmed in the logs reported in the Appendix, Shannon and Markov have the same hardware setup. Both have 96 CPUs with the model name *AMD EPYC 7451 24-Core Processor* and 127TB total capacity. Currently, only 7% of the disks are in use and 119TB is available. Regarding the login shell memory size, we followed the same approach as before and obtained that, on both the Shannon and Markov servers, the virtual memory of the login shell takes 6892 KB.

2 Problem 2

2.1 Measurement of Serial Computation

In order to measure the fraction of the serial computation, we manually changed the code to a "fully serial" implementation stripped down of the process management code (assuming the execution time of multi-process handling is negligible compared to the total execution time). Afterward, we were able to measure the for-loop execution time, *i.e.* the parallel portion T_{par}, versus the following reduction part, *i.e.* sequential portion T_{seq}. We can repeat the experiment for different number of steps and average the results. In order to estimate meaningful time fractions, we set fairly high values for the number of steps. One way to have a rough estimation of the number of steps required might simply be to set a threshold difference between the calculated value of pi and its actual value.

2.2 Maximal Theoretical Speedup and Plotting

In order to calculate the maximal theoretical speedup, we followed two approaches: an empirical-based and a theoretical-based.

Empirical Estimation. Given the proportions T_{seq} and T_{par}, which were measured by following the methodology described in the previous subsection, we can calculate the sequential and parallel fractions of the algorithm as follows:

$$f_{seq} = \frac{T_{seq}}{T_{seq} + T_{par}} \ , \ f_{par} = \frac{T_{par}}{T_{seq} + T_{par}}$$

Theoretical Estimation. Alternatively, we can estimate the sequential/parallel fractions by counting the amount of instructions and their amount of cycles required[1], for the two parts. Each loop is made out of two lookups, two multiplications, a comparison, and eventually a final addition, which happens $\frac{\pi}{4}$ times on average. If we consider 7 clock cycles for performing the multiplication and 1 clock cycle for the rest of the instructions, we can estimate T_{par} being:

$$T_{par} = s \cdot (2 \cdot 7 + 3 + \frac{\pi}{4})$$

On the other hand, the sequential part consists instead of a multiplication by 4, and finally a division. Even though the input is floating point, we can assume the multiplication as a shift operation taking 1 cycle. The floating point division can instead take a considerable amount of cycles and so we assume it being 12 cycles. This brings us to the following estimation of T_{seq}:

$$T_{seq} = 1 + 12$$

Amdhal's Law. Finally, given the execution time $T(p)$ utilizing p parallel workers, the speedup $S(p)$ can be estimated via the Amdhal's law as follows:

$$S(p) = \frac{T(1)}{T(p)} = \frac{T_{seq} + T_{par}}{T_{seq} + \frac{T_{par}}{p}} = \frac{1}{f_{seq} + \frac{1 - f_{seq}}{p}}$$

Evaluation and Plotting. Figures 1 and 2 show the measured speedups against the aforementioned theoretically and empirically estimated speedups. The results were collected by running the command: `python problem-2-2.py -r 10 -s 10000000`, meaning that we averaged the execution time of 10 runs and that we used 10 million steps to calculate pi. We can see that the two models described above perfectly match and that the theoretical speedups are matching the number of utilized parallel workers. This is mainly because of the parallel part dominating the algorithm execution. In fact, in our empirical evaluation we measured that T_{par} accounts for 99.994% of the total execution time.

Regarding the measured speedups instead, we can notice that already from $p = 4$, the gap between theoretical and measured speedups become more and more accentuated. We speculate that the main reason for the difference might lie in the overhead of spawning child processes, which might be significant compared to the actual computation time (which depends on the amount of steps to execute).

2.3 Random Seeding

The application used a hard-coded seed with a value 1 in the main process, but threads in the subprocesses were not set. A seed argument was added to replace the explicit seed with the seed provided. In order to ensure that the seeds in the subprocesses were repeatable and in order, a seed for each worker was created in the `compute_pi` function. The `sample_pi`

[1]Ignoring the parallelism coming from pipelined processors and the Python interpreter overhead.

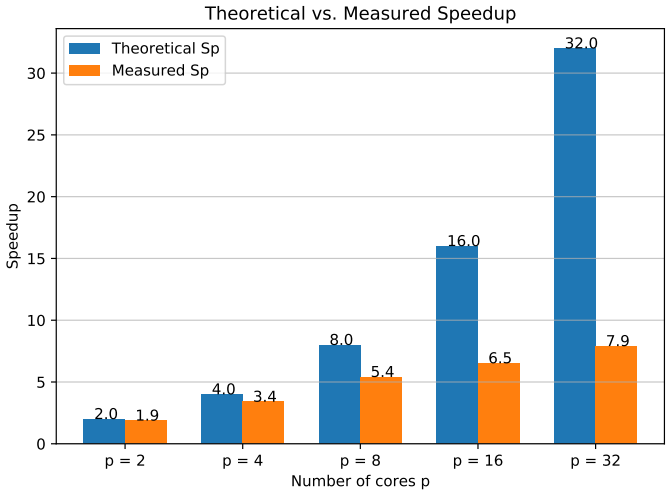

Figure 1: Measured speedups versus theoretical speedup model.

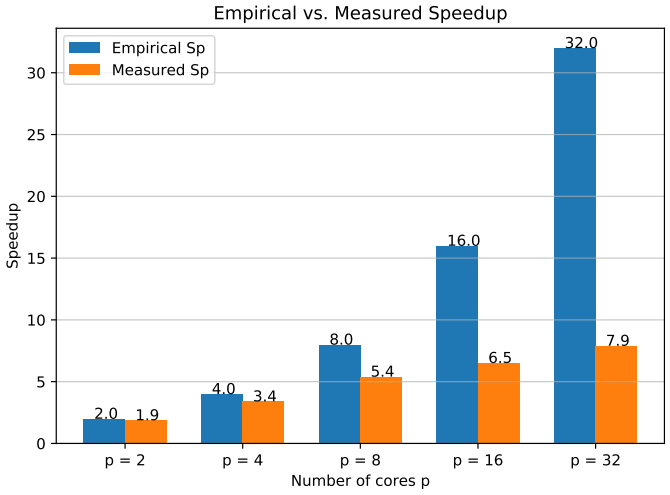

Figure 2: Measured speedups versus empirical speedup model.

function was modified to accept a tuple including the steps and the seed, but also retained backwards compatibility with an integer argument. Since the python `random.seed()` function requires an integer rather than a float (or alternatively, no parameters), it was necessary to create a random integer for which a range must be defined. We empirically found that 2^{16} was sufficiently large to ensure each subprocess had a different seed.

Appendices

A Running Scripts on Shannon and Markov

In order to run commands on Shannon and Markov, we included the following header information in a script file to be called via Slurm (`sbatch script_name.sh`):

```
#!/bin/bash
#SBATCH --partition=cpu-markov
#SBATCH --cpus-per-task=1
#SBATCH --output=/bayes_datainbackup/home/2022/ribes/assignmen1/markov.out
#SBATCH --error=/bayes_datainbackup/home/2022/ribes/assignment1/markov.error
#SBATCH --chdir=/bayes_datainbackup/home/2022/ribes/assignment1/ # Working directory
#SBATCH --export=ALL,TEMP=/scratch,TMP=/scratch,TMPDIR=/scratch
```

The line `#SBATCH --partition=cpu-markov` refers in particular to a script to be run on Markov. For running on Shannon, it has been modified to `#SBATCH --partition=cpu-shannon`.

B Output Logs from Problem 1

B.1 Bayes

```
================================================================================
Number of CPU cores
================================================================================
CPU(s):                          48
On-line CPU(s) list:             0-47
Thread(s) per core:              2
Core(s) per socket:              12
Socket(s):                       2
NUMA node0 CPU(s):               0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,
                                 34,36,38,40,42,44,46
NUMA node1 CPU(s):               1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,
                                 35,37,39,41,43,45,47
================================================================================
CPU Type
================================================================================
Model name:                      Intel(R) Xeon(R) Gold 6126 CPU @ 2.60GHz
CPU MHz:                         1470.528
CPU max MHz:                     3700.0000
CPU min MHz:                     1000.0000
================================================================================
Disk Amount
```

```
================================================================================
Filesystem                                Size  Used Avail Use% Mounted on
udev                                      378G     0  378G   0% /dev
tmpfs                                      76G  3.2M   76G   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv         109G   88G   17G  85% /
tmpfs                                     378G   88M  378G   1% /dev/shm
tmpfs                                     5.0M     0  5.0M   0% /run/lock
tmpfs                                     378G     0  378G   0% /sys/fs/cgroup
/dev/sda2                                 976M  287M  623M  32% /boot
/dev/loop2                                 56M   56M     0 100% /snap/core18/2284
/dev/mapper/vg_data-lv_datainbackup        24T  4.0T   21T  17% /datainbackup
/dev/mapper/vg_data-lv_data                73T  4.3T   68T   6% /data
tmpfs                                      76G   20K   76G   1% /run/user/128
tmpfs                                      76G     0   76G   0% /run/user/8791
tmpfs                                      76G     0   76G   0% /run/user/30122
/dev/loop8                                111M  111M     0 100% /snap/core/12725
/dev/loop1                                 62M   62M     0 100% /snap/core20/1361
tmpfs                                      76G  4.0K   76G   1% /run/user/231489
/dev/loop3                                 44M   44M     0 100% /snap/certbot/1842
/dev/loop0                                 62M   62M     0 100% /snap/core20/1376
/dev/loop4                                111M  111M     0 100% /snap/core/12821
/dev/loop6                                 44M   44M     0 100% /snap/certbot/1888
/dev/loop7                                 56M   56M     0 100% /snap/core18/2344
tmpfs                                      76G  4.0K   76G   1% /run/user/21565
tmpfs                                      76G  4.0K   76G   1% /run/user/24531
tmpfs                                      76G  4.0K   76G   1% /run/user/13738
tmpfs                                      76G  4.0K   76G   1% /run/user/19920
tmpfs                                      76G  4.0K   76G   1% /run/user/10627
tmpfs                                      76G  4.0K   76G   1% /run/user/2499
tmpfs                                      76G  4.0K   76G   1% /run/user/15506
tmpfs                                      76G  4.0K   76G   1% /run/user/5224
tmpfs                                      76G  4.0K   76G   1% /run/user/1627
total                                      99T  8.4T   91T   9% -
================================================================================
Login shell virtual memory usage (code+data+stack) in KB
================================================================================
  7024
```

B.2 Markov

```
================================================================================
Number of CPU cores
================================================================================
```

```
CPU(s):                          96
On-line CPU(s) list:             0-95
Thread(s) per core:              2
Core(s) per socket:              24
Socket(s):                       2
Model name:                      AMD EPYC 7451 24-Core Processor
NUMA node0 CPU(s):               0,8,16,24,32,40,48,56,64,72,80,88
NUMA node1 CPU(s):               2,10,18,26,34,42,50,58,66,74,82,90
NUMA node2 CPU(s):               4,12,20,28,36,44,52,60,68,76,84,92
NUMA node3 CPU(s):               6,14,22,30,38,46,54,62,70,78,86,94
NUMA node4 CPU(s):               1,9,17,25,33,41,49,57,65,73,81,89
NUMA node5 CPU(s):               3,11,19,27,35,43,51,59,67,75,83,91
NUMA node6 CPU(s):               5,13,21,29,37,45,53,61,69,77,85,93
NUMA node7 CPU(s):               7,15,23,31,39,47,55,63,71,79,87,95
================================================================================
CPU Type
================================================================================
Model name:                      AMD EPYC 7451 24-Core Processor
CPU MHz:                         2884.119
================================================================================
Disk Amount
================================================================================
Filesystem                        Size  Used Avail Use% Mounted on
udev                              252G     0  252G   0% /dev
tmpfs                              51G  2.6M   51G   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv 438G   55G  361G  14% /
tmpfs                             252G  1.1M  252G   1% /dev/shm
tmpfs                             5.0M     0  5.0M   0% /run/lock
tmpfs                             252G     0  252G   0% /sys/fs/cgroup
/dev/sda2                         976M  457M  453M  51% /boot
/dev/mapper/vg_data-lv_data        30T   30G   29T   1% /data
/dev/sda1                         511M  5.3M  506M   2% /boot/efi
129.16.29.97:/datainbackup         24T  4.0T   21T  17% /bayes_datainbackup
129.16.29.97:/data                 73T  4.3T   68T   6% /bayes_data
/dev/loop4                         56M   56M     0 100% /snap/core18/2284
/dev/loop9                        111M  111M     0 100% /snap/core/12725
/dev/loop2                         62M   62M     0 100% /snap/core20/1361
/dev/loop8                         68M   68M     0 100% /snap/lxd/22526
/dev/loop3                         62M   62M     0 100% /snap/core20/1376
/dev/loop0                        111M  111M     0 100% /snap/core/12821
/dev/loop5                         56M   56M     0 100% /snap/core18/2344
/dev/loop1                         68M   68M     0 100% /snap/lxd/22753
```

```
tmpfs                                51G     0   51G   0% /run/user/65534
total                               127T  8.4T  119T   7% -
================================================================================
Login shell virtual memory usage (code+data+stack) in KB
================================================================================
  6892
```

B.3 Shannon

```
================================================================================
Number of CPU cores
================================================================================
CPU(s):                          96
On-line CPU(s) list:             0-95
Thread(s) per core:              2
Core(s) per socket:              24
Socket(s):                       2
Model name:                      AMD EPYC 7451 24-Core Processor
NUMA node0 CPU(s):               0,8,16,24,32,40,48,56,64,72,80,88
NUMA node1 CPU(s):               2,10,18,26,34,42,50,58,66,74,82,90
NUMA node2 CPU(s):               4,12,20,28,36,44,52,60,68,76,84,92
NUMA node3 CPU(s):               6,14,22,30,38,46,54,62,70,78,86,94
NUMA node4 CPU(s):               1,9,17,25,33,41,49,57,65,73,81,89
NUMA node5 CPU(s):               3,11,19,27,35,43,51,59,67,75,83,91
NUMA node6 CPU(s):               5,13,21,29,37,45,53,61,69,77,85,93
NUMA node7 CPU(s):               7,15,23,31,39,47,55,63,71,79,87,95
================================================================================
CPU Type
================================================================================
Model name:                      AMD EPYC 7451 24-Core Processor
CPU MHz:                         2831.555
================================================================================
Disk Amount
================================================================================
Filesystem                        Size  Used Avail Use% Mounted on
udev                              252G     0  252G   0% /dev
tmpfs                              51G  2.7M   51G   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv  438G   81G  334G  20% /
tmpfs                             252G  764K  252G   1% /dev/shm
tmpfs                             5.0M     0  5.0M   0% /run/lock
tmpfs                             252G     0  252G   0% /sys/fs/cgroup
/dev/sda2                         976M  457M  453M  51% /boot
/dev/sda1                         511M  5.3M  506M   2% /boot/efi
```

```
/dev/mapper/vg_data-lv_data          29T   30G   29T   1% /data
129.16.29.97:/datainbackup           24T  4.0T   21T  17% /bayes_datainbackup
129.16.29.97:/data                   73T  4.3T   68T   6% /bayes_data
/dev/loop7                           56M   56M     0 100% /snap/core18/2284
/dev/loop9                          111M  111M     0 100% /snap/core/12725
/dev/loop5                           62M   62M     0 100% /snap/core20/1361
/dev/loop8                           68M   68M     0 100% /snap/lxd/22526
/dev/loop3                           62M   62M     0 100% /snap/core20/1376
/dev/loop0                          111M  111M     0 100% /snap/core/12821
/dev/loop4                           56M   56M     0 100% /snap/core18/2344
/dev/loop1                           68M   68M     0 100% /snap/lxd/22753
tmpfs                                51G     0   51G   0% /run/user/65534
total                               127T  8.4T  119T   7% -
================================================================================
Login shell virtual memory usage (code+data+stack) in KB
================================================================================
  6892
```