UiO : USIT

HPC - Best Practices


Best Practice Guide
Modern Processors

Ole Widar Saastad, University of Oslo, Norway
Kristina Kapanova, NCSA, Bulgaria
Stoyan Markov, NCSA, Bulgaria
Cristian Morales, BSC, Spain
Anastasiia Shamakina, HLRS, Germany
Nick Johnson, EPCC, United Kingdom
Ezhilmathi Krishnasamy, University of Luxembourg, Luxembourg
Sebastien Varrette, University of Luxembourg, Luxembourg
Hayk Shoukourian (Editor), LRZ, Germany

Updated 5-5-2021

PRACE

Ole W. Saastad, ITF/FI

Best Practices HPC

First order approximation : «I’ve tested it and it works»

Higher orders : How to improve, efficiency, performance, experience, resource usage, good behavior, sharing experience, help onboarding[1]

1:

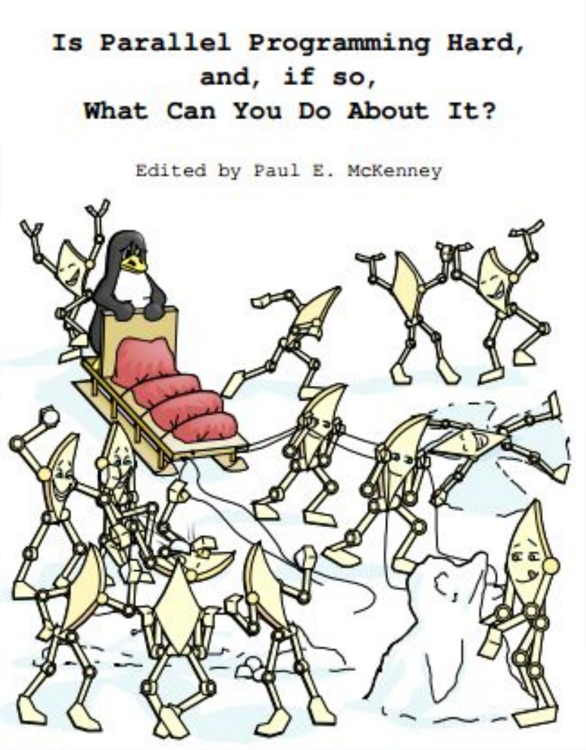
Know your workload

Understanding what you run on the system is the key factor.

Remember you are not alone

HPC systems are shared by users who are all eager to get their work done!

Don't waste resources



What information do I need ?

How do I get this data ?

Gathering data about my work

Applications

- Single core / multicore ?
- Shared memory / Distributed memory ?

Gathering data about my work

The most important :

- Memory footprint
- Cores/processors - Scaling
- Input & output
- Run time

Applications

Single node application - single or multicore - Shared memory

Keywords : OpenMP, multicore, shared memory, threaded, pthread, POSIX thread, NCPU

ldd ./pi.x | grep -i omp
ldd ./pi.x | grep -i thread
strings ./pi.x | grep OMP
strings ./pi.x | grep thread

Applications

Multiple node application - single or multicore - Distributed memory

Keywords : MPI, distributed memory

```
ldd pi.x| grep mpi
strings  pi.x| grep mpi
```

Running applications

Memory, cores and run time

- Manual
- Other users
- Guess
- Trial and error

SLURM test jobs, exploring parameter space

Running applications

Interactively ?

Very short jobs (less than 5 min wall time) can be tested on front end

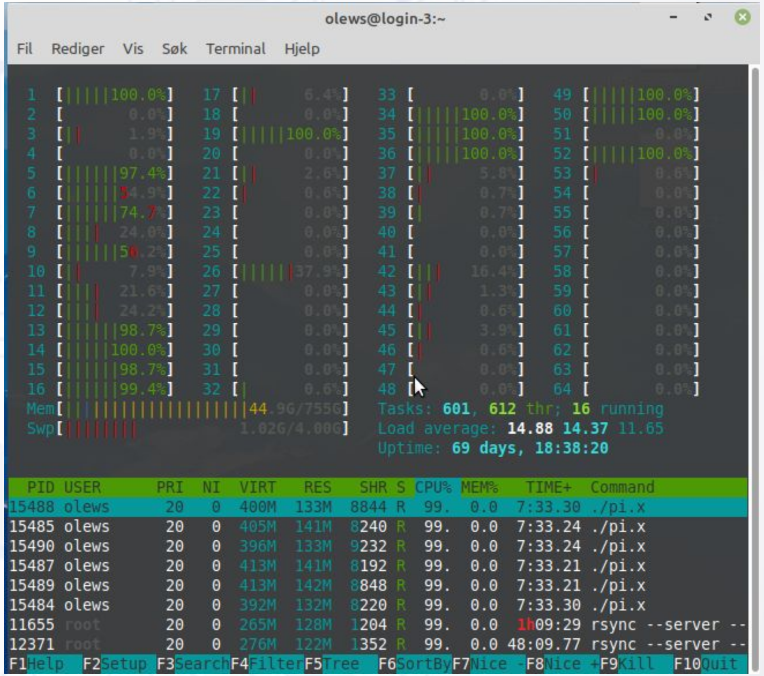
Reserve a node using SLURM, run interactively and explore.

Running applications

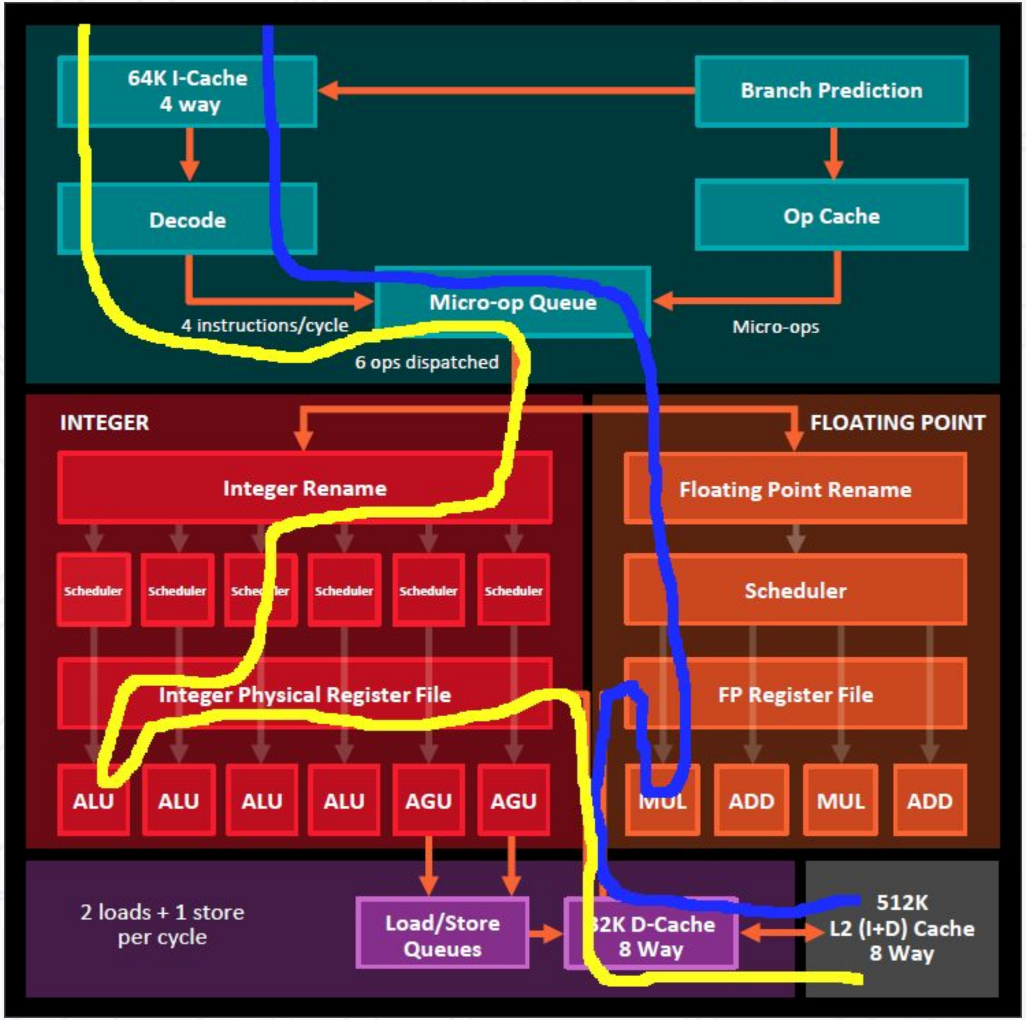
Interactive tools providing memory footprint and cores

- top
- htop

Review the man pages



A core running two instruction streams (one integer + one floating point), two threads.



Running applications

SLURM test jobs, exploring parameter space

```
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=1
#SBATCH --time=00:10:00
#SBATCH --mem-per-cpu=2000M
```

This is a 5 dimensional space, only educated guesses are possible.

Running applications

SLURM test jobs, valuable feedback in logs:

```
Task and CPU usage stats:
      JobID JobName   AllocCPUS    NTasks    MinCPU MinCPUTask AveCPUElapsed Exit Code
------------ ---------- ---------- -------- ---------- ---------- ---------- ---------- --------
5312354           pi          8                                          00:01:02     0:0
5312354.bat+      batch       4      1         00:03:53       0    00:03:53      00:01:02     0:0
5312354.ext+      extern      8      2    00:00:00       0    00:00:00       00:01:02     0:0

Memory usage stats:
      JobID MaxRSS MaxRSSTask AveRSS MaxPages    MaxPagesTask    AvePages
------------ ---------- ---------- ---------- -------- -------------- ----------
5312354
5312354.bat+   1113772K        0   1113772K     0            0            0
5312354.ext+      0            0            0     0            0            0
```

Input and output - storage

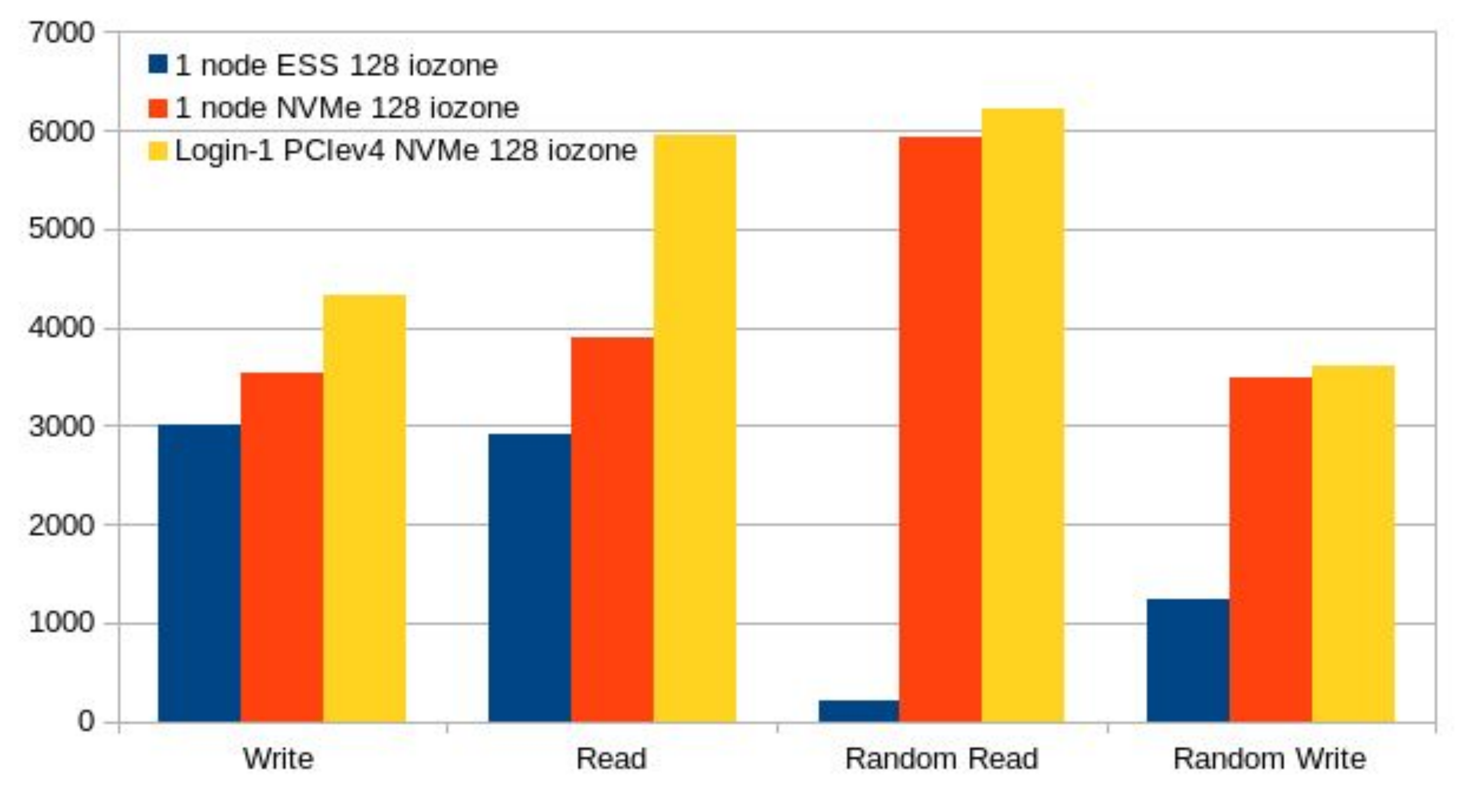
SLURM provide stats for IO

dd if=/dev/zero of=$SCRATCH/dd.tmp bs=1024 count=10000000

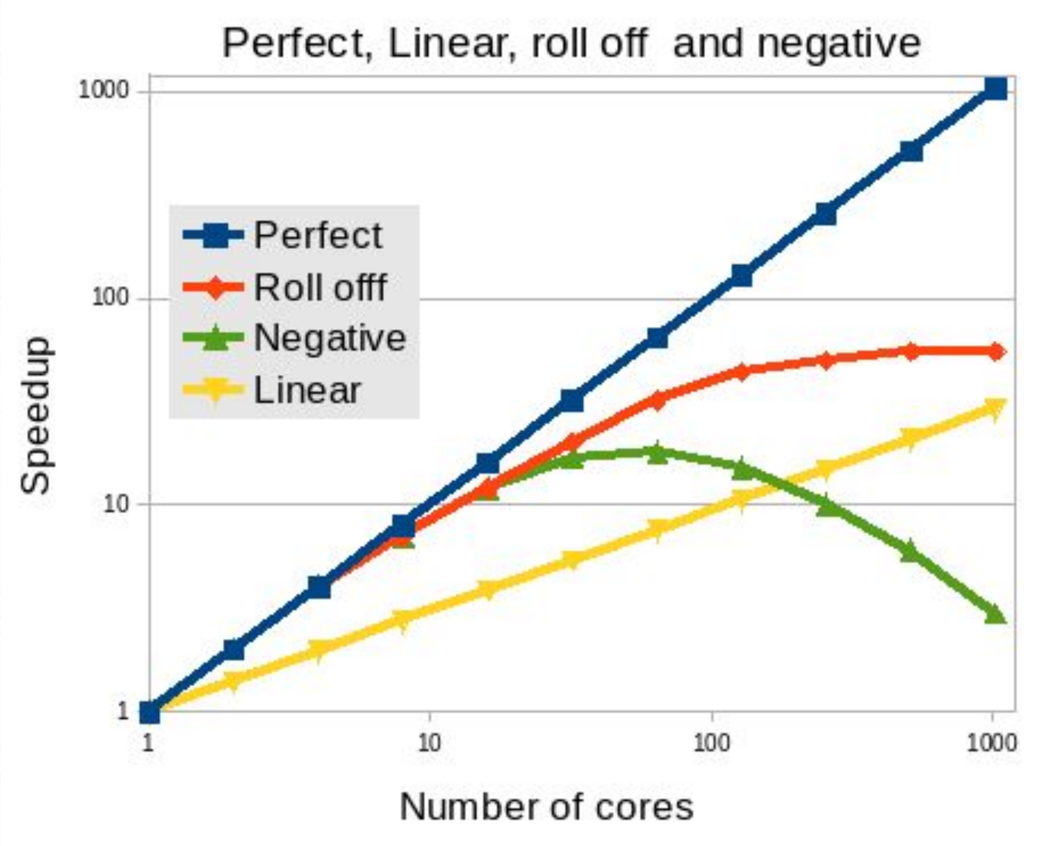
Disk usage stats:

JobID	MaxDiskRead	MaxDiskReadTask	AveDiskRead	MaxDiskWrite	MaxDiskWriteTask	AveDiskWrite
5312653						
5312653.bat+	9765.78M	0	9765.78M	9765.63M	0	9765.63M
5312653.ext+	0.00M	0	0.00M	0	0	0

Input and output - storage



Scaling



Perfect, Linear, roll off and negative
Perfect
Roll offf
Negative
Linear
Speedup
1000
100
10
1
1
10
100
1000
Number of cores

Amdahl's law

$$Speedup = \frac{1}{(1-p) + p/N}$$

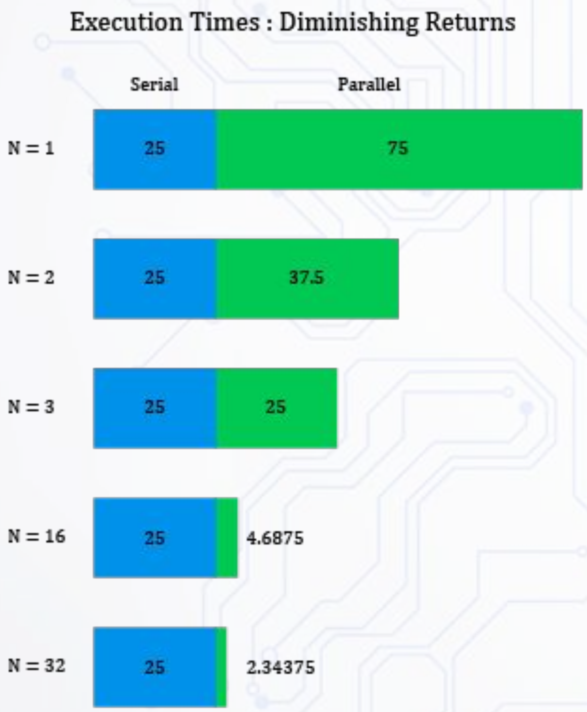
Suppose a car is traveling between two cities 60 miles apart, and has already spent one hour traveling half the distance at 30 mph. No matter how fast you drive the last half, it is impossible to achieve 90 mph average before reaching the second city. Since it has already taken you 1 hour and you only have a distance of 60 miles total; going infinitely fast you would only achieve 60 mph.

Gustavssons' law

$$\begin{aligned} S &= s + p \times N \\ &= s + (1 - s) \times N \\ &= N + (1 - N) \times s \end{aligned}$$

Suppose a car has already been traveling for some time at less than 90mph. Given enough time and distance to travel, the car's average speed can always eventually reach 90mph, no matter how long or how slowly it has already traveled. For example, if the car spent one hour at 30 mph, it could achieve this by driving at 120 mph for two additional hours, or at 150 mph for an hour, and so on.

Amdahl’s law & Gustavssons’ law



/projects/ec34/inf9380/Exercises-23.Mar/