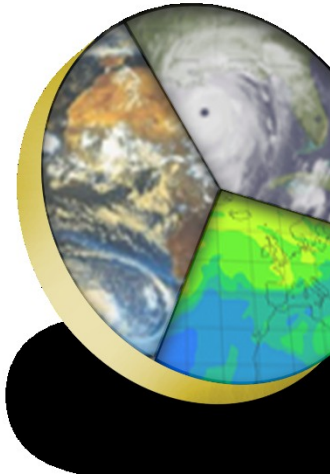
Introduction to HPC

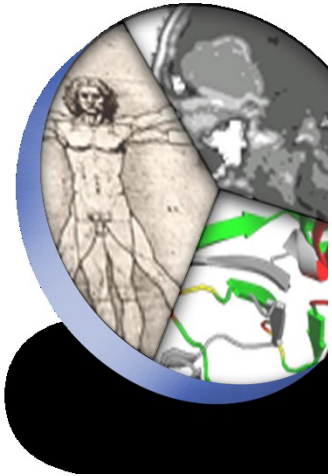
Radosław Januszewski
radekj@man.poznan.pl
EuroHPC, Poznań
15.05.2019

What is HPC?

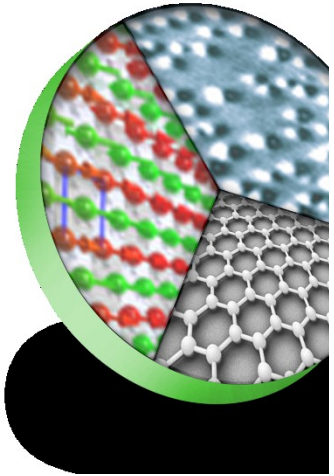
> *High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.*



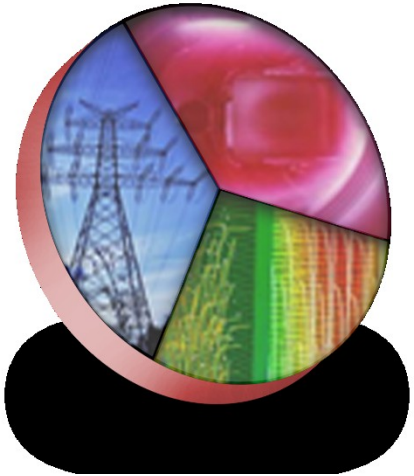
Environment
Weather, climate prediction



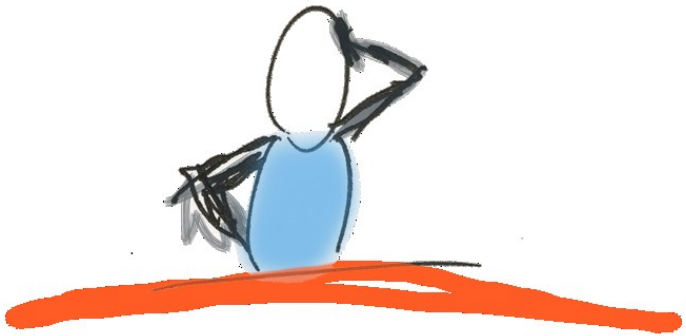
Aging population
medicine
biology



Material science
Nanotechnology
Quantum chemistry

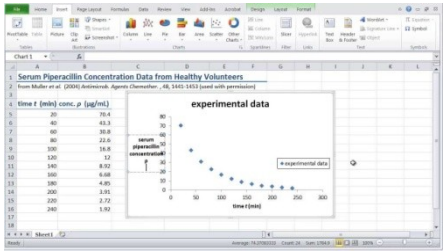


Energy
fusion
Green energy
HEP

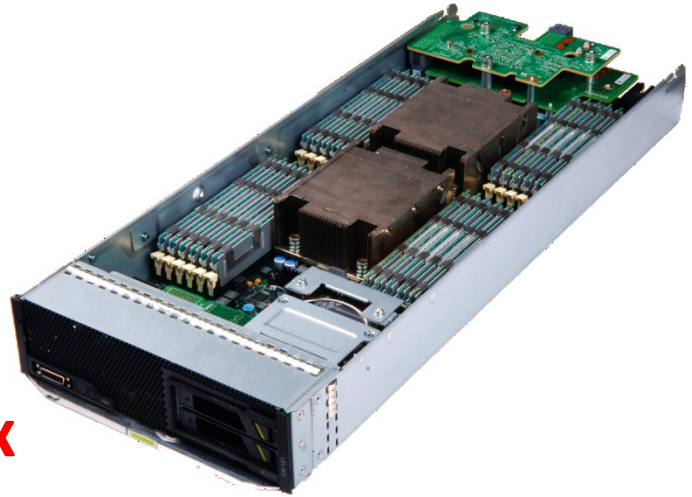



6:18
Saturday, June 18
Lunch with Eric
Café
11:00 AM-12:00 PM

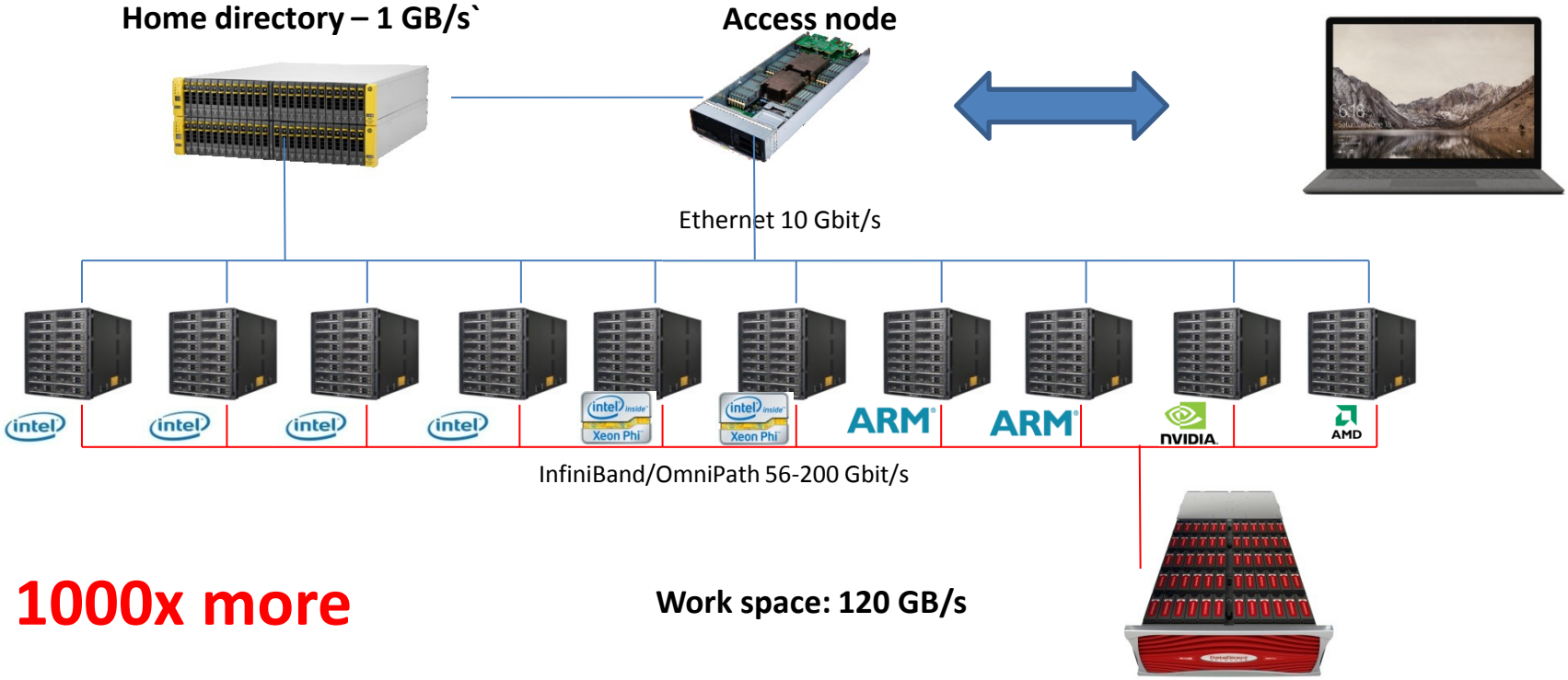
Local computing: limits



10x – 100x

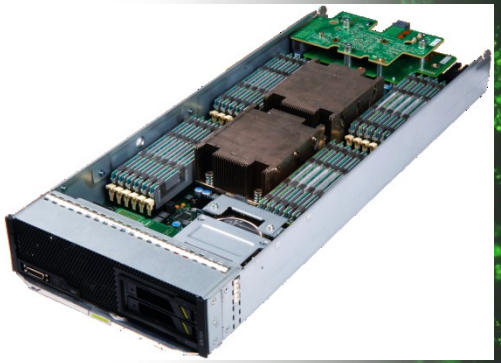


Supercomputer / cluster



Important terms:

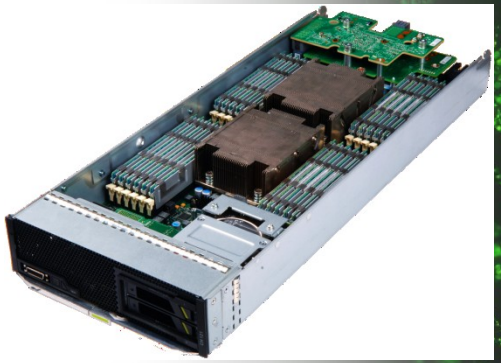
login/access node:



Server (node = server) that is accessible from outside world. Used for file manipulation and editing, submitting jobs etc.
Access basically via ***ssh terminal*** ... with some exceptions

```
radekj@eagle: ~
Plik  Edycja  Widok  Wyszukiwanie  Terminal  Pomoc
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           5487039 standard CommandL    plgpw CF   19:16:14      1 e0860
           5487162    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5487163    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5487198    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5485936    bigmem HD0_J0_1   komasa PD       0:00      1 (Resources)
           5485938    bigmem HD0_J0_1   komasa PD       0:00      1 (Priority)
           5485939    bigmem HD0_J0_1   komasa PD       0:00      1 (Priority)
           5487159    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5487160    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5251533 standard orca2.sh    scanya PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:e1077)
           5489095 standard jenkins_     bolen  R       1:33      2 e[0013,0038]
           5489080 standard jenkins_     bolen  R      36:58      2 e[0013,0038]
           5488662 standard s0_J9_11    komasa  R   16:23:40      1 e0065
           5488663 standard      bash  andreas  R   16:31:46      1 e0054
           5488661 standard s0_J9_11    komasa  R   16:32:46      1 e1079
           5488659 standard s0_J9_11    komasa  R   16:32:49      1 e0681
           5488660 standard s0_J9_11    komasa  R   16:32:49      1 e0683
           5488658 standard s0_J9_11    komasa  R   16:32:52      1 e0680
           5488657 standard s0_J9_11    komasa  R   16:32:55      1 e0679
           5488656 standard s0_J9_11    komasa  R   16:32:56      1 e0687
           5488655 standard s0_J9_11    komasa  R   16:32:58      1 e0686
           5488654 standard s0_J9_11    komasa  R   16:33:01      1 e0691
```

Compute node:



Assigned for computations by queue system
Physicall server
No public addressess
Usually can access internet
Available only for the lifetime of the application (with some exceptions :))

Home directory/filesystem:

Shared filesystem visible on the login node and compute nodes. Used to store input and result files, NOT supposed to be used for computations (limited performance)
Usually backup is performed on this filesystem data.

Scratch space:

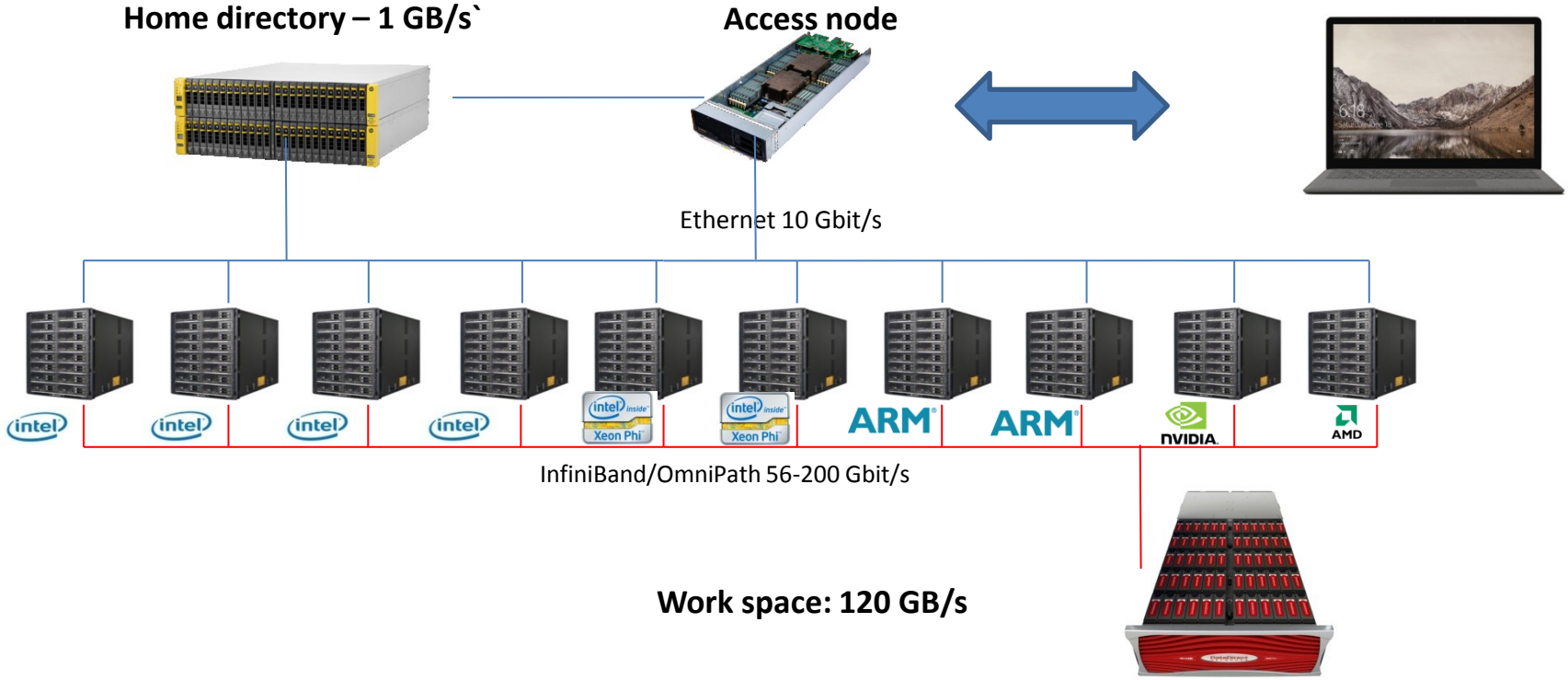
A shared filesystem accessible on login node and compute nodes, much faster than home directory. Main purpose: temporary storage used during computation. No backups for this space

On PSNC cluster it is /tmp/lustre_shared directory

HPC network:

A specialized, low latency (10x lower than „normal”), high troughput network, in most
Cases Infiniband or OmniPath
Important to use it instead of „legacy” ethernet (not allways present)

Supercomputer / cluster



Preliminary requirements:

Rudamentary knowledge about Linux needed

Applications are usually installed by admins

Available support from admins, sometimes also programming/porting teams

Examples
https://wiki.man.poznan.pl/kdm

PCSS Poznańskie Centrum Superkomputerowo-Sieciowe

WIKI KDM

OPROGRAMOWANIE

A
- Abaqus
- Abinit
- Acml
- Amber
- Ansys

B
- Bowtie
- Bcftools
- Bedops
- Bedtools
- Blasr
- Blast
- Boost
- Breakdancer
- Bwa

C
- Codeanalyst
- Cudatoolkit
- Canu
- Cgal
- Cnvnator
- Control-freec
- Cp2k
- Cufflinks

D
- DL_POLY Classic
- Darshan
- Discovar
- Dot

G
- Gamess
- Gaussian
- Gromacs
- Genome-strip
- Gmp

H
- Hmmer

I
- Ingap
- Interproscan
- Ior
- Irods

L
- Lammps
- Lordec
- Lsc

M
- Mapdamage
- Matlab
- Mumax
- Meme

N
- Namd
- Netcdf
- Nwchem

O
- Orca

P
- Plink
- Picard-tools
- Pilon
- Pindel
- Plink
- Preseq
- Proovread
- Psi

Q
- Quantum Espresso

R
- Rna-Seqc
- Rsem
- R
- Rdxplorer

S
- Siesta
- Samtools
- Segemehl
- Seqtk

T
- Tabix
- Tbb
- Tophat
- Trinityrnaseq

V
- Vcftools
- Velvet
- Vowpal-wabbit

W
- Wgs
- Wxparaver

EAGLE CLUSTER

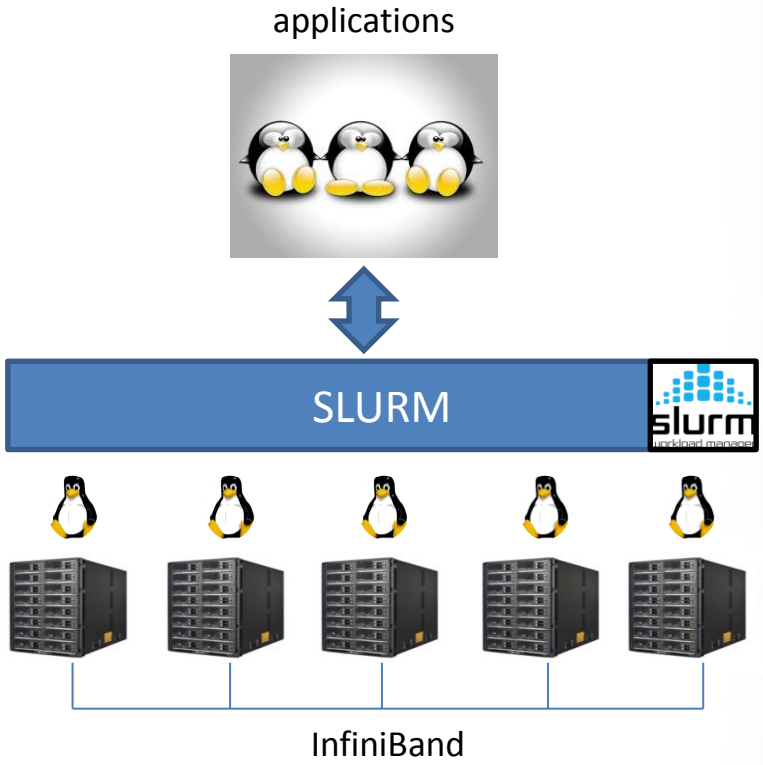
1088 servers, 33k cores,

- Intel Xeon: 2x14 cores @2,6 GHz
- **64-256** GB RAM/server
- IB FDR **56Gbit**
- No local HDD

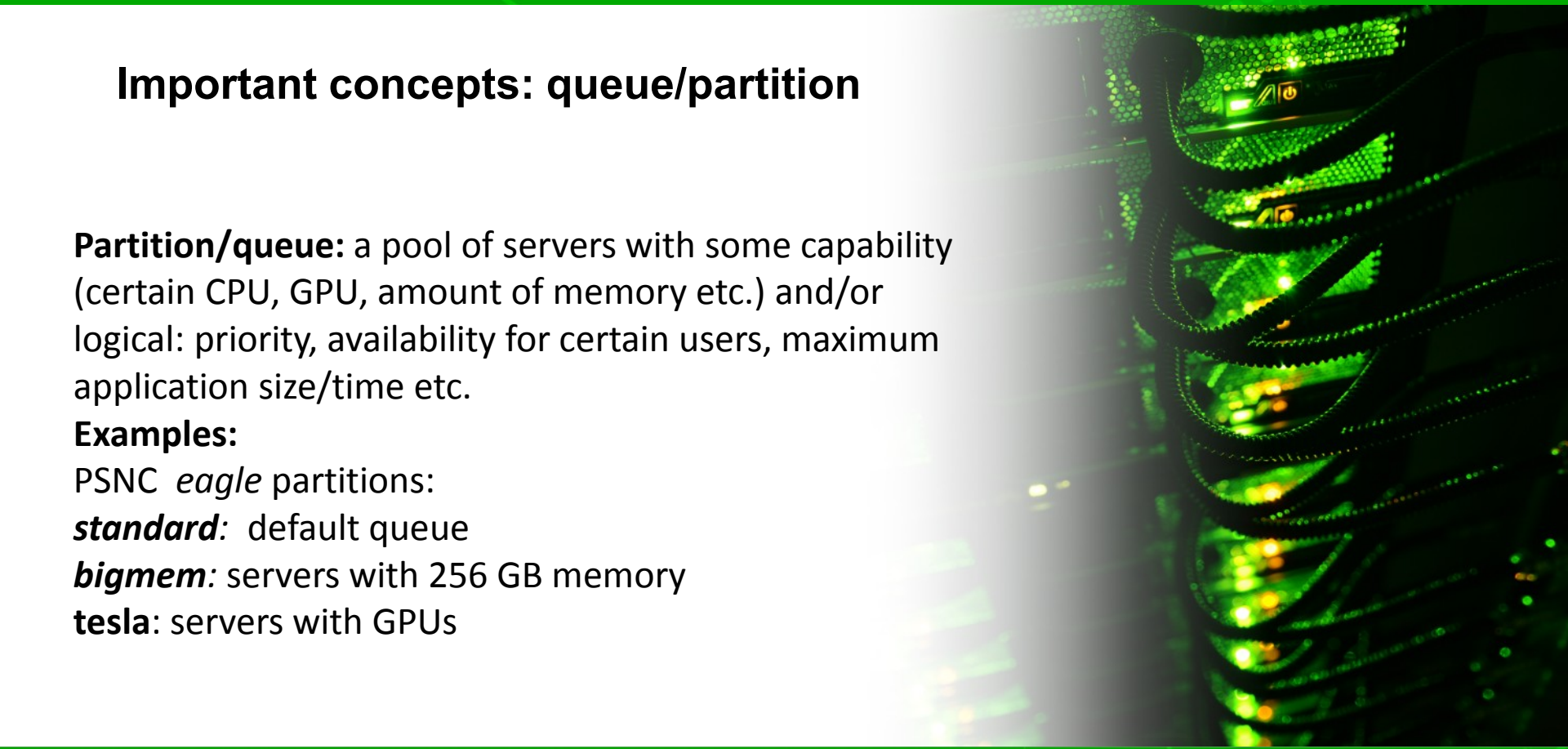
Storage

- Home: 1GB/s, 10 PB
- Scratch –120 GB/s, 5PB

How do I know which servers are mine to use?



Important concepts: queue/partition



Partition/queue: a pool of servers with some capability (certain CPU, GPU, amount of memory etc.) and/or logical: priority, availability for certain users, maximum application size/time etc.

Examples:

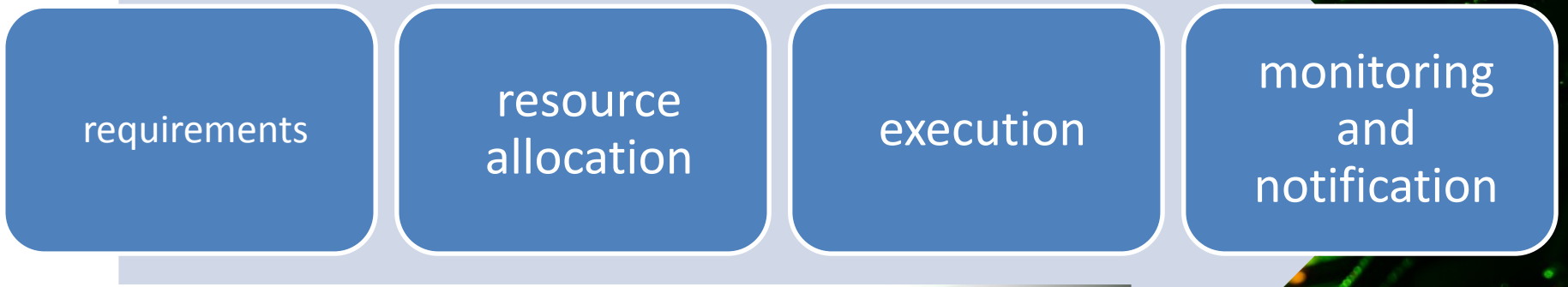
PSNC *eagle* partitions:

standard: default queue

bigmem: servers with 256 GB memory

tesla: servers with GPUs

Queue system:
main resource manager of the supercomputer



Queue system in practice:

Step 1: create file with application description

```
#!/bin/bash
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=28
#SBATCH --mem=128gb
#SBATCH --time=01:00:00
#SBATCH –p standard

# loadin preinstalled application modules
module load my_app

cp data.txt  /tmp/lustre/my_work_directory/

# start applications
mpirun  my_app.exe data.txt > result.txt

cp /tmp/lustre/my_work_directorey/result.txt  /home/my_home_dir
```

Server count

Cores per server

Max memory

Estimated runtime

Partition we want to use

Queue system in practice:
Step 2: submit job

On the login node:

sbatch my_app.slurm

Monitoring – what is going on with my application?

radekj@eagle: ~

Plik Edycja Widok Wyszukiwanie Terminal Pomoc

```
radekj@eagle:~$ squeue -u komasa
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           5487162    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5487163    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5487198    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5485936    bigmem HD0_J0_1   komasa PD       0:00      1 (Resources)
           5485938    bigmem HD0_J0_1   komasa PD       0:00      1 (Priority)
           5485939    bigmem HD0_J0_1   komasa PD       0:00      1 (Priority)
           5487159    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5487160    bigmem s2_J2_12   komasa PD       0:00      1 (Priority)
           5488662  standard s0_J9_11   komasa  R   16:32:26      1 e0065
           5488661  standard s0_J9_11   komasa  R   16:41:32      1 e1079
           5488659  standard s0_J9_11   komasa  R   16:41:35      1 e0681
           5488660  standard s0_J9_11   komasa  R   16:41:35      1 e0683
           5488658  standard s0_J9_11   komasa  R   16:41:38      1 e0680
           5488657  standard s0_J9_11   komasa  R   16:41:41      1 e0679
           5488656  standard s0_J9_11   komasa  R   16:41:42      1 e0687
           5488655  standard s0_J9_11   komasa  R   16:41:44      1 e0686
           5488654  standard s0_J9_11   komasa  R   16:41:47      1 e0691
           5488653  standard s0_J9_11   komasa  R   16:41:50      1 e0072
           5487154  standard h2_J2_Dg   komasa  R   18:21:10      1 e0305
```

Notifications

```
#!/bin/bash
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=28
#SBATCH --mem=128gb
#SBATCH --time=01:00:00
#SBATCH –p standard

#SBATCH –mail-type=<type>
```

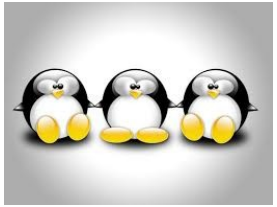
Email notification

<type> it may be: BEGIN, END, FAIL, ALL …

Other options: skype, facebook etc.

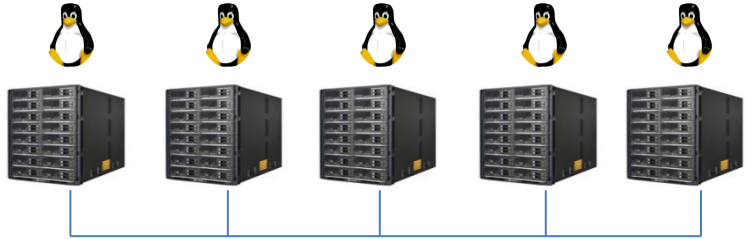
Cloud / HPC environment

apps



Queue system

Cloud engine



InfiniBand

10 GEth

HPC vs Cloud:

HPC is more resource aware (eg. physicall topology aware)

HPC is performance oriented

In HPC there is no **overbooking** or **sharing** – everything is **dedicated**

HPC = known hardware and stable software – important for numerical stability and debugging

HPC is not really a real-time environment… (usually)

(usually) **no persistant** services

Steep learning curve...

Software layer:

SaaS or IaaS depending on how you look at it

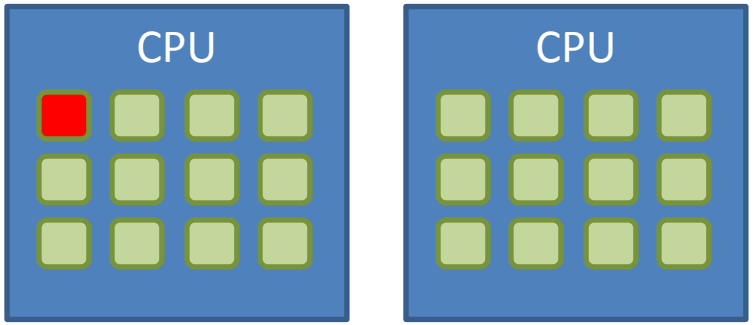
Linux only (with some exceptions)

basically no Virtual Machines, existence of containers (Singularity)

Schedulers/resource managers/queue systems: (open source) **Slurm**, Torque, **PBS**, (proprietary): load leveler and some more

API: no standard APIs, some schedulers have APIs (e.g. pyslurm). AFAIK no REST

How to make use of supercomputer: single thrads

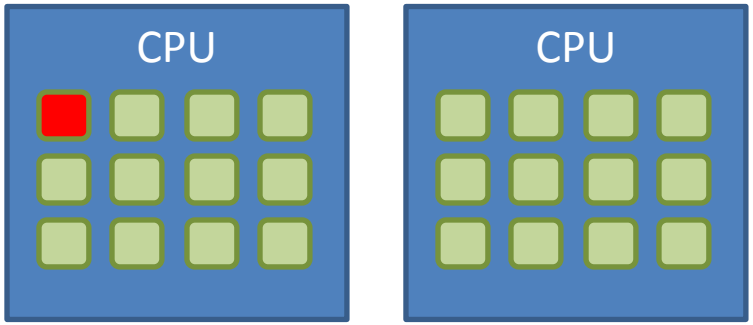


- Easy to use
- One need to distribute problem
- … and manage gathering results
- Perfect parallelism ☺
- Array jobs are handy

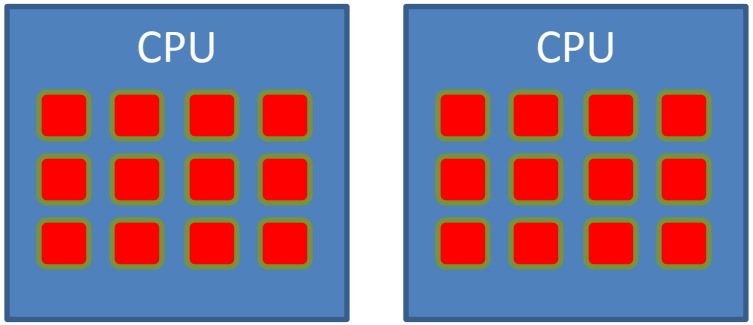
sbatch my_app.exe dane.txt
Hundreds or thousands jobs

Multithreading

Starting point: month-long simulation



./my_app.exe dane.txt



./export omp_num_threads=28
./my_app.exe dane.txt
28x faster ☺

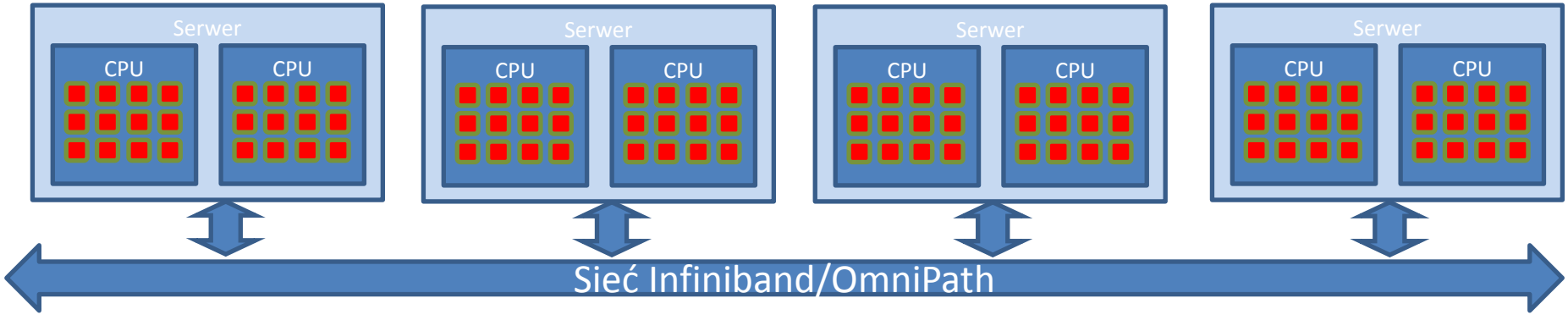
Multithreading programming

- Relatively simple to programm and use
- A lot of tools and libraries supporting this model
- (most probably) supported in all modern programming languages
- Parallelism limit: one server ☹

- Most applications installed by admins can exploit this method

Real challenge: massively parallell applications

Usually (but not exclusively) done by the means of using **MPI**



*28*4*

mpirun -np 112 my_app.exe dane.txt

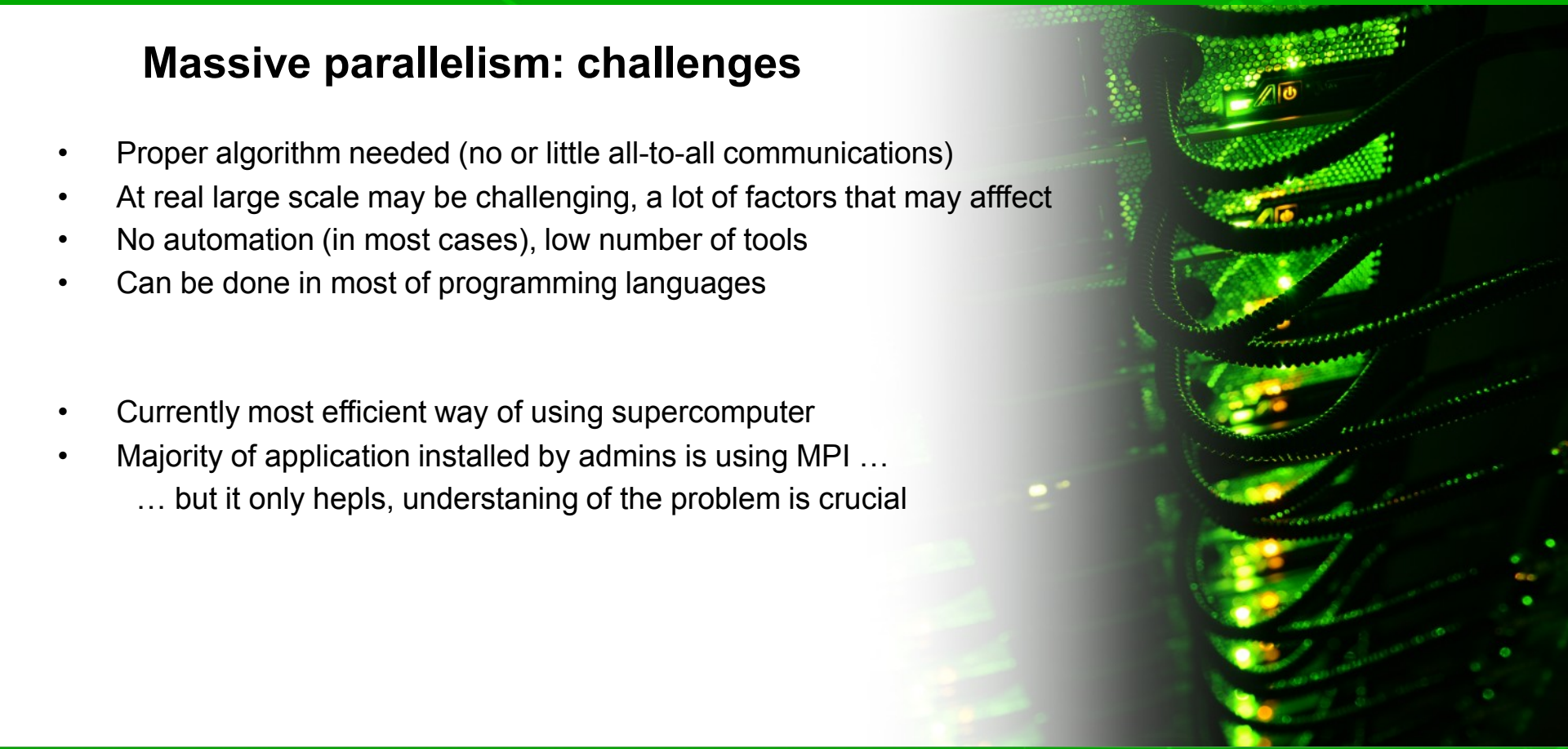
Ending of simulation: 6,5 hours vs 1 month,

Theretically given enough resources one can do it in 1s ☺ but…

Massive parallelism: challenges

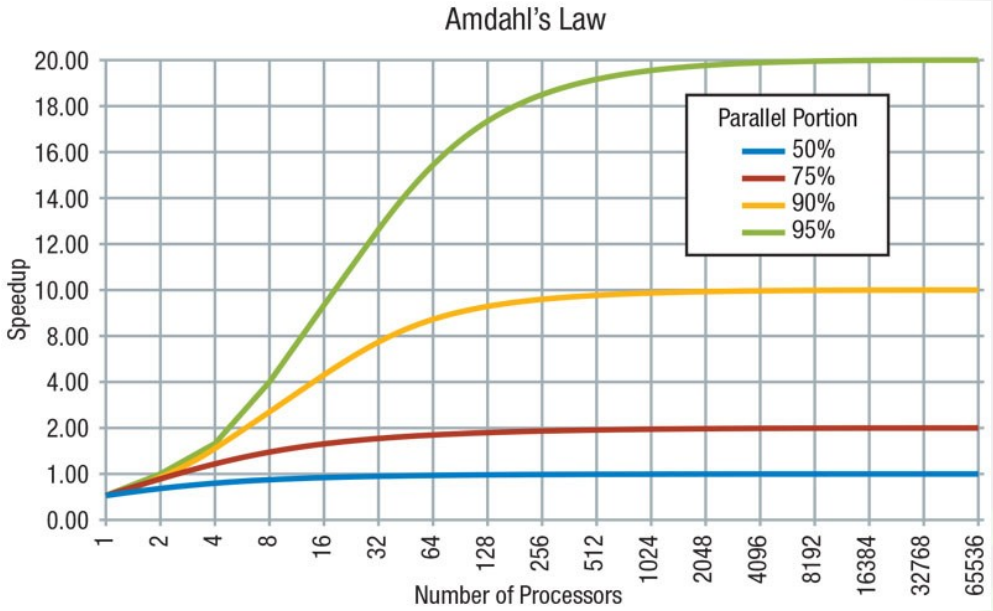
- Proper algorithm needed (no or little all-to-all communications)
- At real large scale may be challenging, a lot of factors that may afffect
- No automation (in most cases), low number of tools
- Can be done in most of programming languages

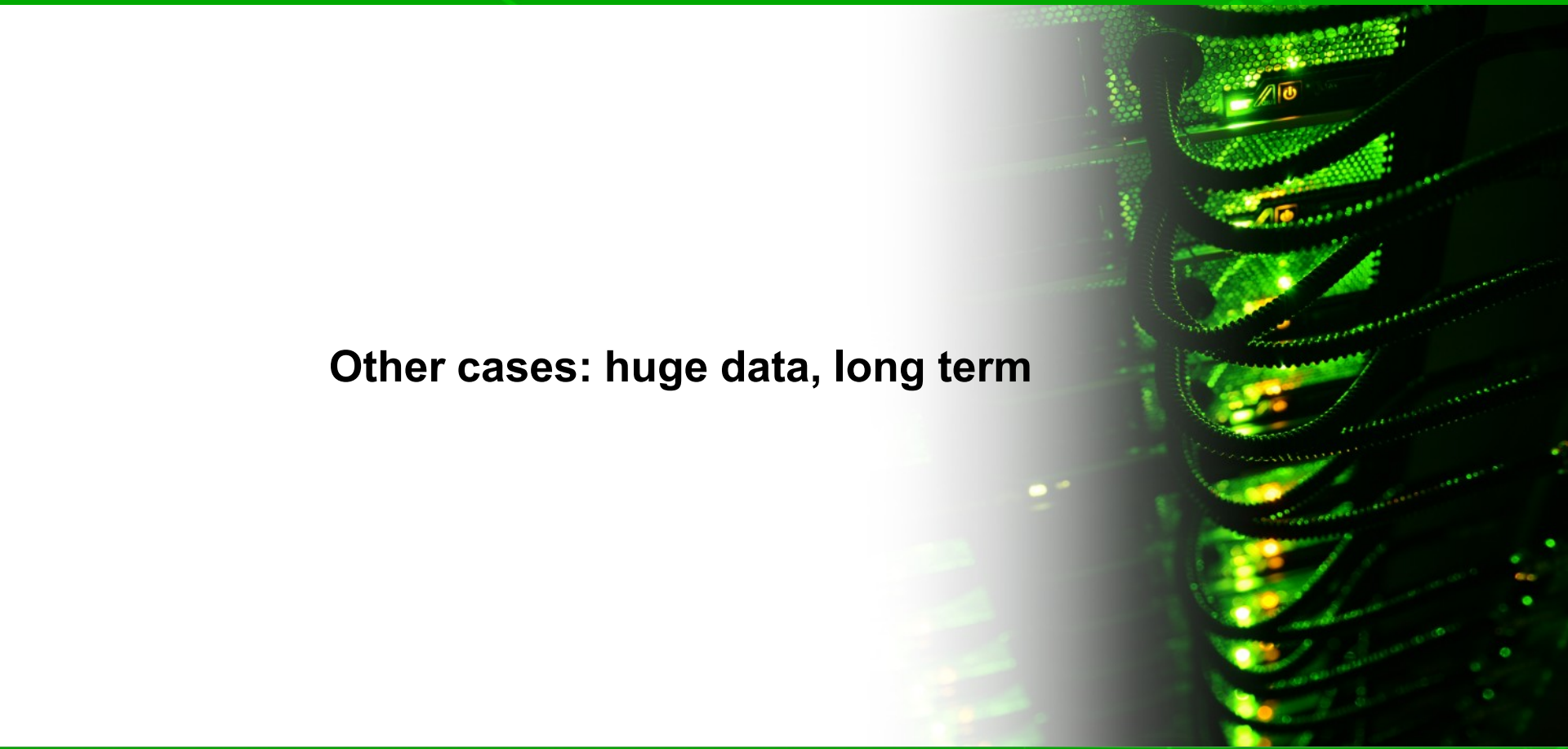
- Currently most efficient way of using supercomputer
- Majority of application installed by admins is using MPI …
  … but it only hepls, understaning of the problem is crucial



Speed-up challenges:

Speedup challenges vol.2





Other cases: huge data, long term

http://www.astron.nl/dailyimage/pictures/20160518/First_byte_Poznan.png

TAPE LIBRARIES

Old technology… for IT ancient

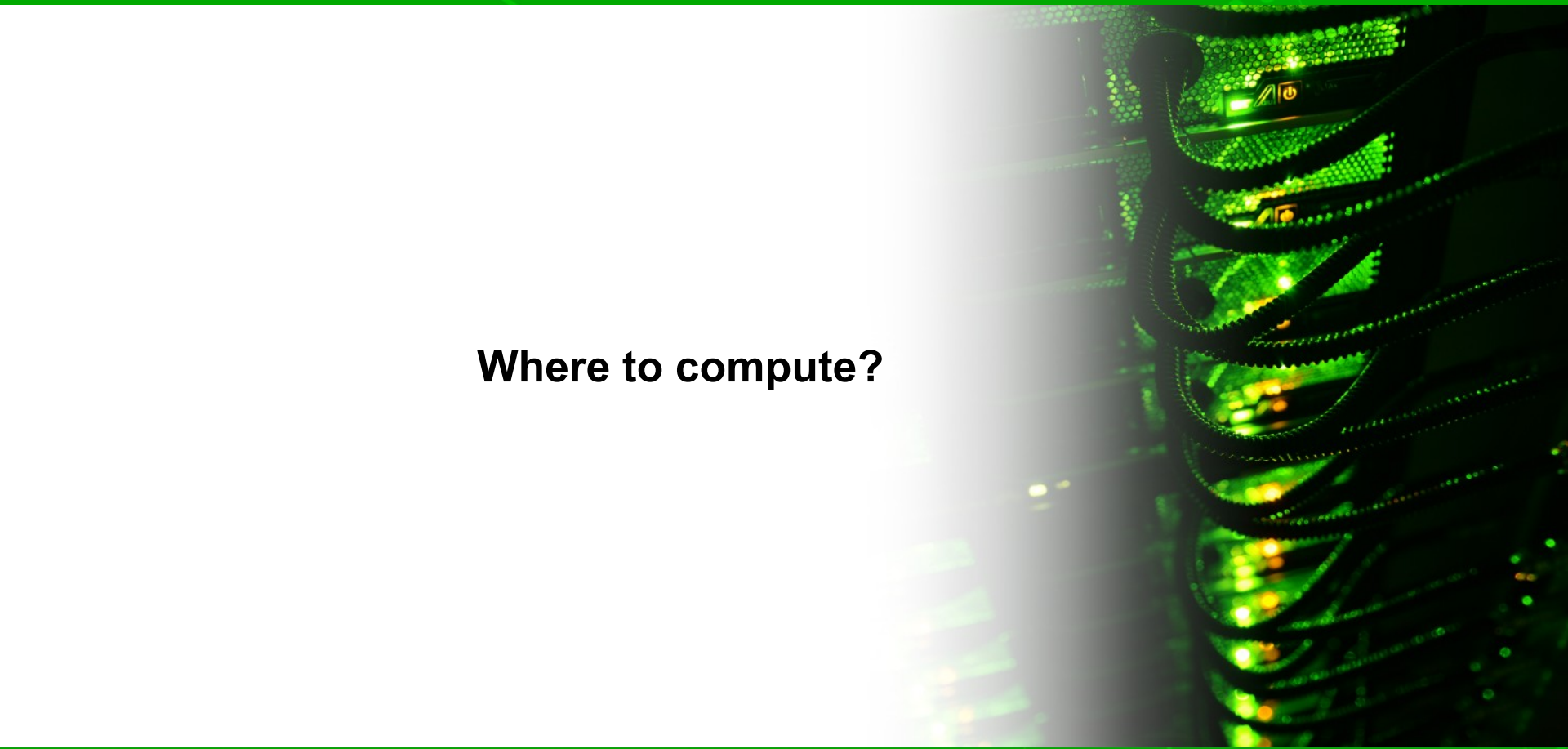
~~Fast~~ LARGE

Services: archives, backup

Configuration:

- 3.5 PB LTO5
- 25 PB IBM TS1150 (Jaguar)
- software: IBM TSM, GPFS, HSM

Where to compute?



Where to do it: PRACE ☺ Tier-0, Tier-1:

BlueGene/P 5.87 Petaflop/s
PRACE@GCS@Jülich

Bull Cluster **Curie** 1.8 Petaflop/s
PRACE**@**GENCI@CEA

BlueGene/Q 1 Petaflop/s
PRACE@CINECA

CRAY HORNET
4 Petaflop/s
PRACE@GCS@HLRS

IBM SuperMUC
3 Petaflop/s
PRACE@GCS@LRZ

MareNostrum
1 Petaflop/s
PRACE@BSC

Common questions:

Why HPC and not Amazon / Google /other cloud

- Low latency network
- Dedicated vs shared issue
- HPC is SaaS (kind of) and Cloud is usually IaaS (a lot of effort to start)
- Support !!!!
- $$ ☺
- CLI is so 90s, is there anything else?

How can I start using supercomputer?

e.g.:

- https://hpc.man.poznan.pl
- http://www.plgrid.pl/oferta/zasoby_obliczeniowe/granty_obliczeniowe

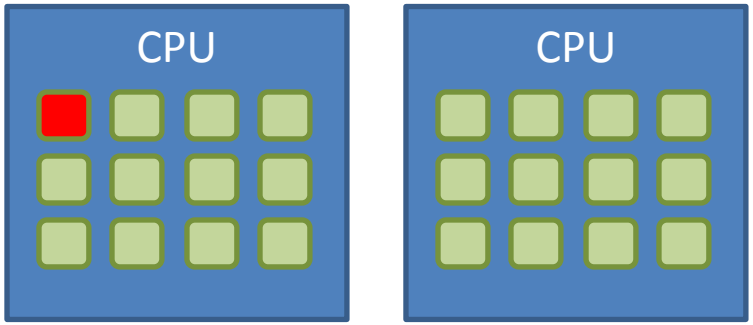
MULTI-PURPOSE IS NOT ALWAYS AN ADVANTAGE

Source: http://www.fastcarinvasion.com/must-see-moment-tractor-crosses-way-racing-car/

Questions?

Thank you

Jak wykorzystać potencjał klastra: jednowątkowość



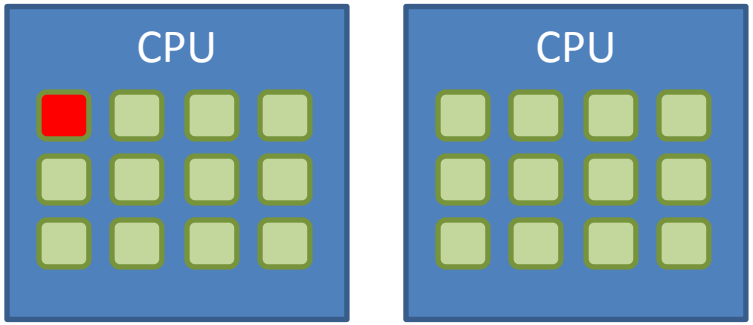
- Proste w użyciu
- Wymaga podziału problemu
- … i zebrania wyników
- Idealne równoleglenie ☺
- Zadania typu array

./moja_aplikacja.exe dane.txt

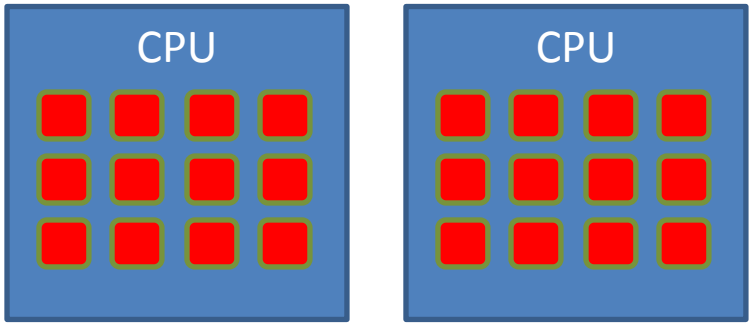
Setki lub tysiące zadań…

Multithreading

Punkt wyjścia: symulacja trwa miesiąc



./moja_aplikacja.exe dane.txt



./export omp_num_threads=28
./moja_aplikacja.exe dane.txt
28x szybciej ☺

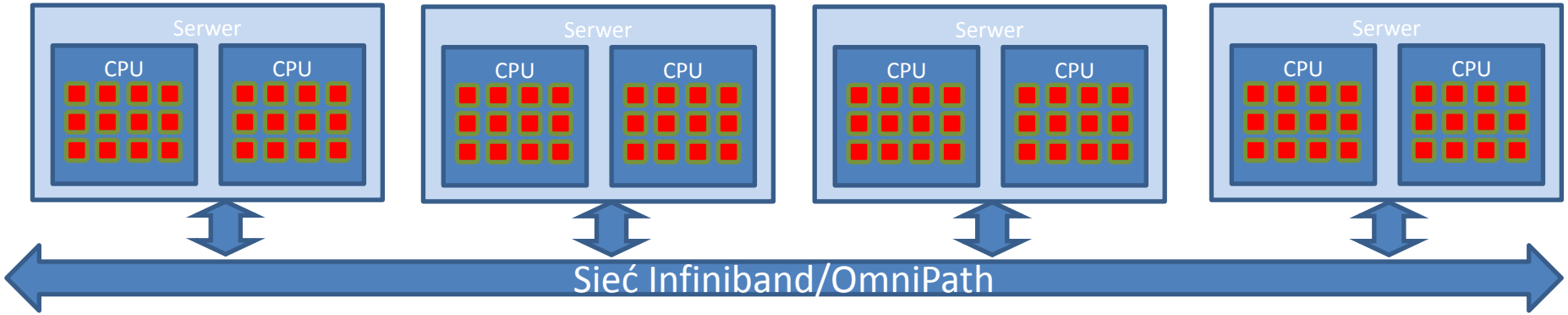
Programowanie rozproszone: wątki

- Relatywnie proste w użyciu i programowaniu
- Sporo narzędzi ułatwiających równoleglenie kodu
- Wsparcie (prawdopodobnie) we wszystkich językach programowania
- Limit równoległości: jeden serwer

- Większość aplikacji zainstalowanych na klastrach potrafi wykorzystać ten machanizm

Jak wykorzystać potencjał klastra: zadania rozproszone

Aplikacja musi korzystać / zostać skompilowana z bibliotekami **MPI**



*28*4*

mpirun -np 112 moja_aplikacja.exe dane.txt

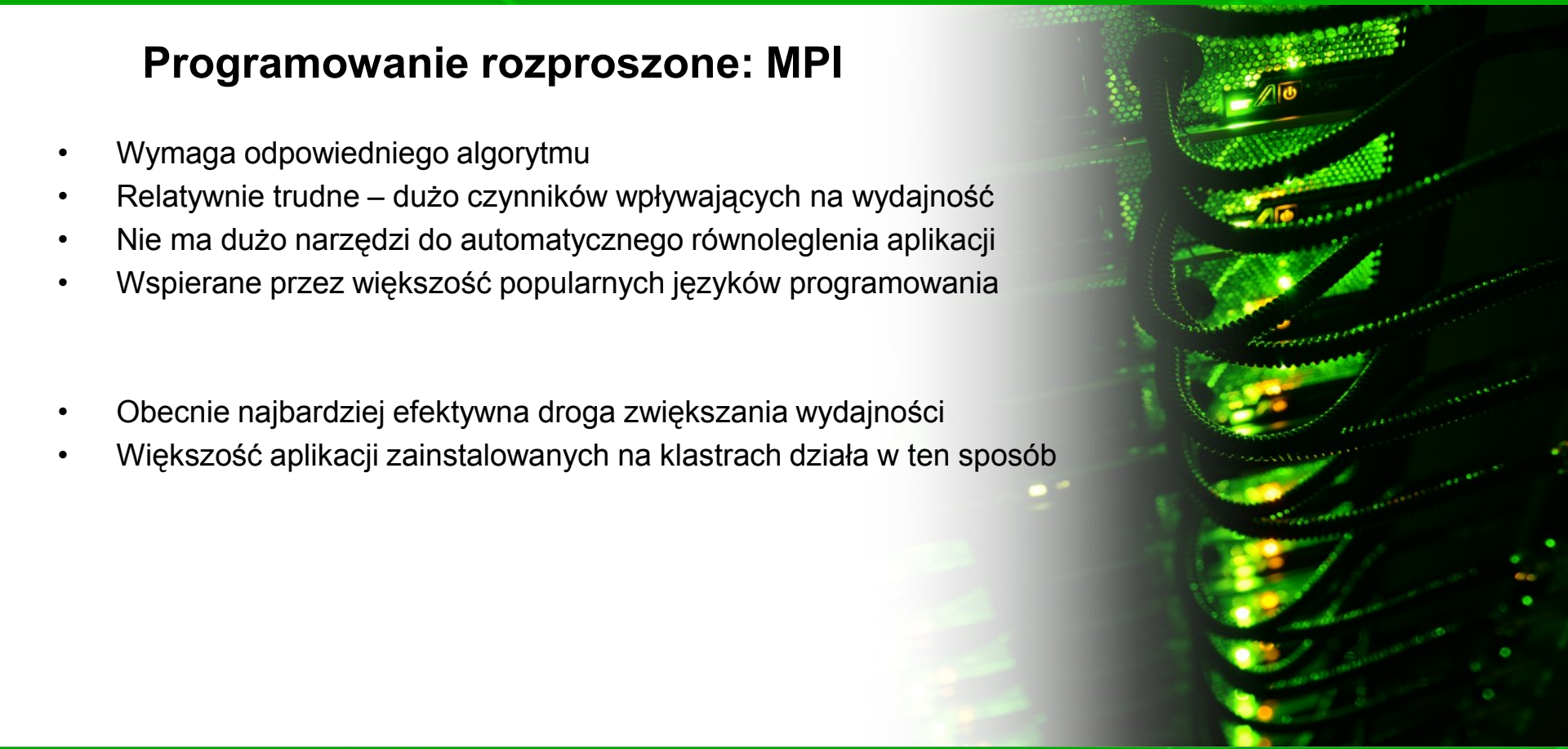
Zakończenie symulacji: 6,5 godziny,

Teoretycznie można zmniejszyć np. do 1 min

Programowanie rozproszone: MPI

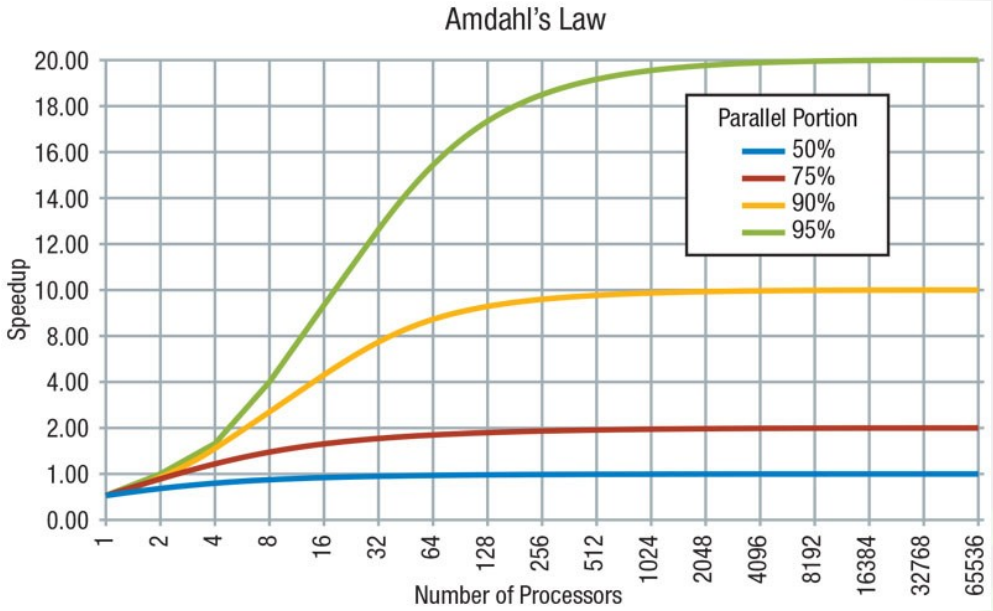
- Wymaga odpowiedniego algorytmu
- Relatywnie trudne – dużo czynników wpływających na wydajność
- Nie ma dużo narzędzi do automatycznego równoleglenia aplikacji
- Wspierane przez większość popularnych języków programowania

- Obecnie najbardziej efektywna droga zwiększania wydajności
- Większość aplikacji zainstalowanych na klastrach działa w ten sposób

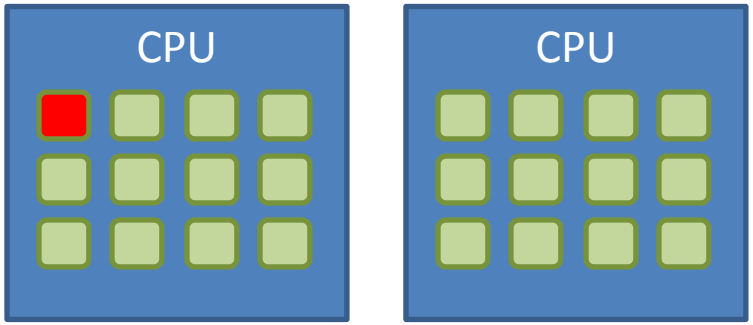


Problemy z przyspieszeniem:

Problemy z przyspieszeniem



Jak wykorzystać potencjał klastra: jednowątkowość



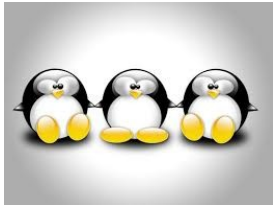
- Proste w użyciu
- Wymaga podziału problemu
- … i zebrania wyników
- Idealne równoleglenie ☺
- Zadania typu array

./moja_aplikacja.exe dane.txt

Setki lub tysiące zadań…

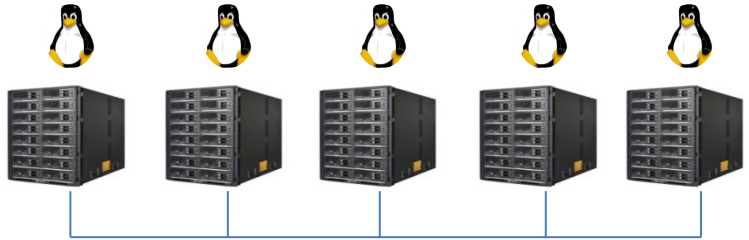
Chmurę też mamy!

Aplikacje



SYSTEM KOLEJKOWY slurm

Cloud engine openstack



InfiniBand

10 GEth

It is all about large problems… or many

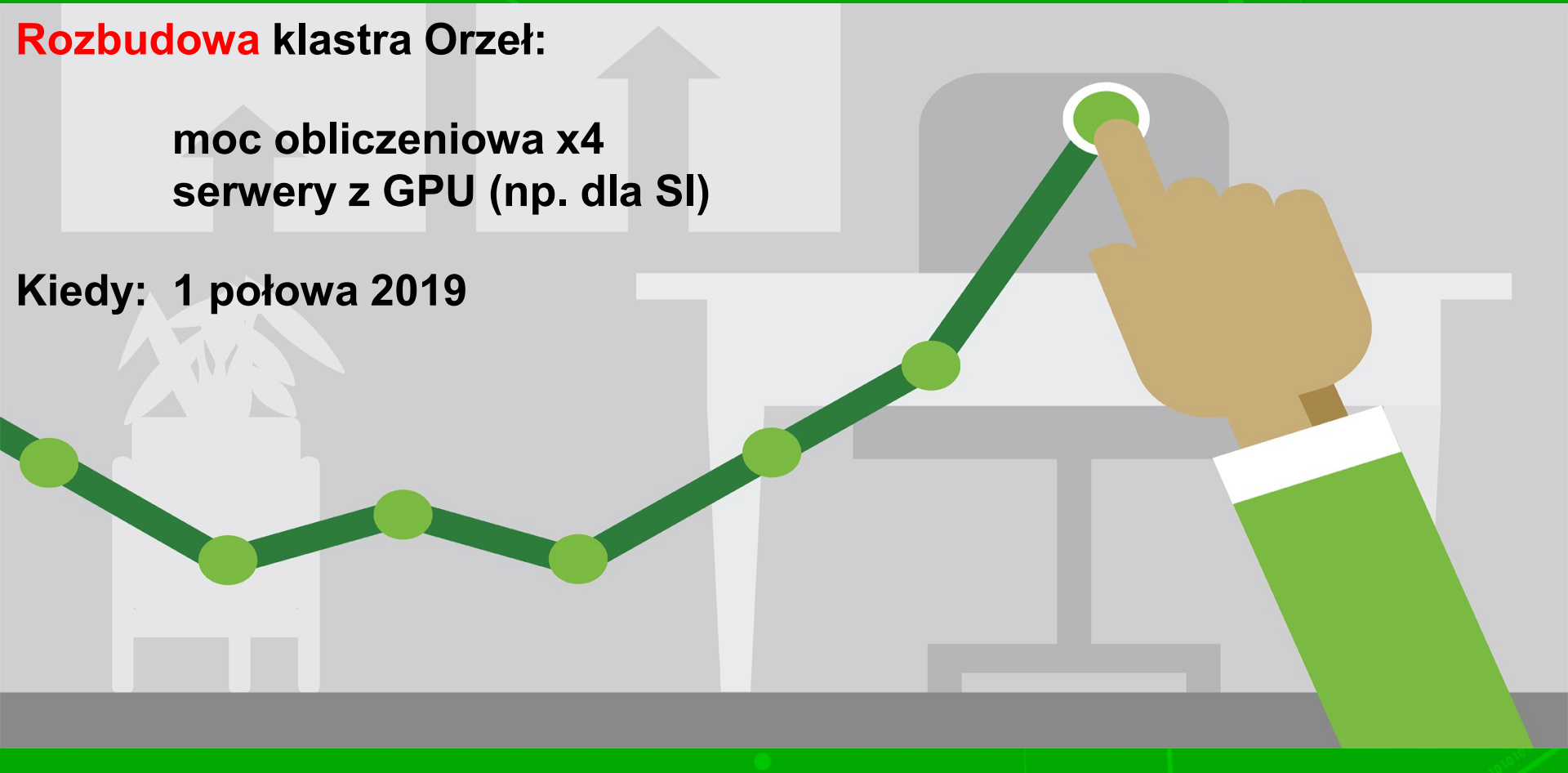
- HPC
  - Kind of IaaS
  - Kind of SaaS..

Najbliższa przyszłość

Rozbudowa klastra Orzeł:

moc obliczeniowa x4
serwery z GPU (np. dla SI)

Kiedy: 1 połowa 2019



Nowy portal dla użytkowników – koniec 2019:
HPC via WWW

Piszemy to dla uż lkie uwagi mile widziane