



CentraleSupélec

Feedback from a data center for education at CentraleSupélec engineering school

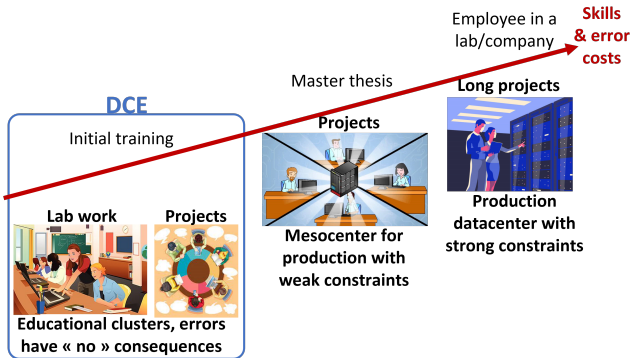
J. Fix, S. Vialle, R. Hellequin, C. Mercier, P. Mercier, J.-B. Tavernier

jeremy.fix@centralesupelec.fr

IPDPS - Edupar Workshop 2022



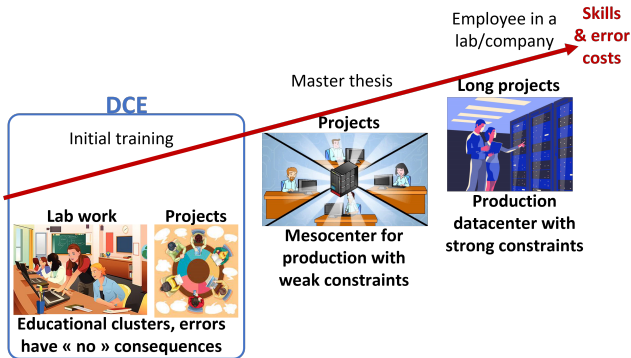
Motivations



For teaching, for beginners

- powerfull shared computing resources

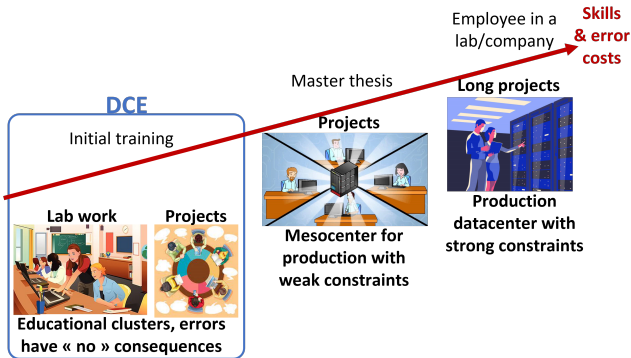
Motivations



For teaching, for beginners

- powerfull shared computing resources
- no bill to the end users

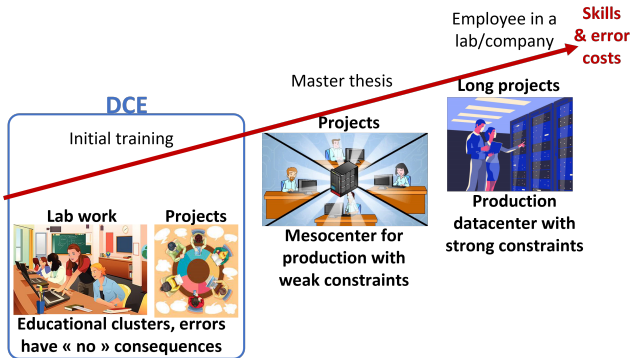
Motivations



For teaching, for beginners

- powerfull shared computing resources
- no bill to the end users
- errors without consequences

Motivations



For teaching, for beginners

- powerfull shared computing resources
- no bill to the end users
- errors without consequences
- access as simple and ergonomic as possible

Where is the Data Center for Education (DCE)





What is the Data Center for Education (DCE)

CPU clusters

- Kyle : 68×16 physical cores, 64 GB
- Sarah : 32×8 physical cores, 32 GB

Eth 10Gb/s

AVX, AVX2, AVX-512





What is the Data Center for Education (DCE)

CPU clusters

- Kyle : 68×16 physical cores, 64 GB
- Sarah : 32×8 physical cores, 32 GB

Eth 10Gb/s

AVX, AVX2, AVX-512

GPU servers

- Cameron : 17 GPU servers (1080)
- Tx: 17 GPU servers (2080Ti)
- Sh: 17 GPU servers (3090)



What is the Data Center for Education (DCE)

CPU clusters

- Kyle : 68×16 physical cores, 64 GB
- Sarah : 32×8 physical cores, 32 GB

Eth 10Gb/s

AVX, AVX2, AVX-512

GPU servers

- Cameron : 17 GPU servers (1080)
- Tx: 17 GPU servers (2080Ti)
- Sh: 17 GPU servers (3090)



Admin machines

1(+1)×Gateway, 1(+1) frontal SLURM, NFS server + 1 storage bay



Administration

- resources are scheduled with **slurm**
- the installations are synchronized with **ansible**
- the nodes are monitored with **Zabbix**
- labs are planed on website `www.dce-cs.fr`
- users access the nodes with home made software `dcejs`

Administration

- resources are scheduled with **slurm**
- the installations are synchronized with **ansible**
- the nodes are monitored with **Zabbix**
- labs are planed on website `www.dce-cs.fr`
- users access the nodes with home made software `dcejs`

Teaching

- Hadoop / Spark pre-installed distributed servers (on Sarah)
- Intel OneAPI, gcc, python, OpenMP, OpenMPI, OpenBLAS, CUDA, cublas, cudnn, ...
- Tensorflow, PyTorch, pandas, numpy, scikit-learn
- Robot Operating System (ROS) ecosystem

Custom envs. with conda (soon with module/singularity)

Hardware costs

- Kyle (68 nodes): 160k€
- Tx (17 nodes + 2080Ti) : 50k€



Hardware costs

- Kyle (68 nodes): 160k€
- Tx (17 nodes + 2080Ti) : 50k€

Energy costs

- CPUs clusters : $\approx 103MWh/y$
- GPUs servers : ≈ 80 120MWh/y (Cam,Tx + Sh)

Total estimated : ~~182~~ 222MWh/y ($\approx 25K€$)

Not accounted : the free cooling with large fans

Hardware costs

- Kyle (68 nodes): 160k€
- Tx (17 nodes + 2080Ti) : 50k€

Energy costs

- CPUs clusters : $\approx 103MWh/y$
- GPUs servers : $\approx 120MWh/y$ (Cam, Tx + Sh)

Total estimated : ~~182~~ 222MWh/y ($\approx 25K€$)

Not accounted : the free cooling with large fans

Humans

3 engineers (40%, 5%, 5%) and 2 professors (10%, 5%)

≈ 7.8 (french-)man months/year

Using the DCE



CentraleSupélec



Registering a lab

Entering your TP request

Submit your request

Description of the lab

Planning lab sessions

November 2021						
Mon	Tue	Wed	Thu	Fri	Sat	Sun
01 Nov TP 01 08:00-10:00				02 Nov TP 02 08:00-10:00		
03 Nov TP 03 08:00-10:00	04 Nov TP 04 08:00-10:00	05 Nov TP 05 08:00-10:00	06 Nov TP 06 08:00-10:00	07 Nov TP 07 08:00-10:00	08 Nov TP 08 08:00-10:00	09 Nov TP 09 08:00-10:00

Session date * beginning hour Duration ⓘ +Available ⓘ Type de TP * Cluster ⓘ * Type of session

04/01/2021

08:00

4

00:30

Machine Learning

GPUe

Lab session

To delete

To add

Teacher requests are validated by the admin team.



Registering a lab

Entering your TP request

Submit your request

Description of the lab: Planning lab sessions

Session date	beginning hour	Duration	+Available	Type de TP	Cluster	Type of session
04/01/2022	08:00	4	00:30	Machine Learning	GPUe	Lab session

To delete

To add

Teacher requests are validated by the admin team.

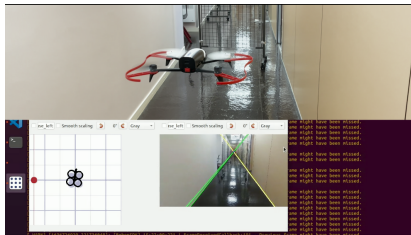
Co-hosting labs and projects

Partition	Max time	# nodes	Purpose
gpu_tp_resa	4h30	34	Scheduled lab works
gpu_tp	2h	34	Lab works live coding
gpu_prod_night	12h	16	Batch jobs, night& weekends
gpu_inter	2h	42	Project live coding
gpu_prod_long	48h	17	Long batch jobs

Description of the use case

The students experiment algorithms on robotic platforms:

- written in ROS with multiple interacting processes,
- computationnally intensive processes (MonteCarto / Particle filters for SLAM, planning and navigation, Gazebo simulator),
- requires an interactive session



Also labs with on-site platforms during Covid-19.

Description of the use case

The student train deep neural networks (pytorch/tensorflow):

- during ML labs, or projects/challenges (see Kaggle 3MD4040)
- both interactive and batch mode (long running jobs)

The screenshot displays a multi-panel interface for machine learning on GPU servers. The left panel shows a 'DCR Connection tool' with fields for Cluster (GPU), Login (gpu), and a 'List of your allocations' table. The middle panel shows a code editor with Python code for training a neural network. The right panel shows TensorBoard with training metrics graphs.

DCR Connection tool

Cluster:

Login:

List of your allocations

Job ID	Partitions	Time	Nodes	VC	Hostname	Actions
2002.0	gpu_1node	3:20	g40	--	--	<input type="button" value="Actions"/>
2002.1	gpu_1node	3:20	g40	--	--	<input type="button" value="Actions"/>
2002.2	gpu_1node	3:20	g40	--	--	<input type="button" value="Actions"/>
2002.3	gpu_1node	3:20	g40	--	--	<input type="button" value="Actions"/>
2001	gpu_1node	8:56	g40	localhost:5000	--	<input type="button" value="Actions"/>

log: VNC done. Please start your views.

Code Editor

```

import torch
import torch.nn as nn
import torch.nn.functional as F

class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 16, kernel_size=5)
        self.conv2 = nn.Conv2d(16, 16, kernel_size=5)
        self.fc1 = nn.Linear(16 * 5 * 5, 120)
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        x = F.torch.flatten(torch.unsqueeze(x, dim=0), start_dim=1)
        x = F.relu(self.conv1(x))
        x = F.relu(self.conv2(x))
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return F.softmax(x, dim=0)

net = Net()
optimizer = optim.Adam(net.parameters())

# Training loop
for epoch in range(10):
    net.train()
    for data_loader in data_loader_loader:
        for data, target in data_loader_loader:
            optimizer.zero_grad()
            output = net(data)
            loss = F.cross_entropy(output, target)
            loss.backward()
            optimizer.step()
    net.eval()
    for data_loader in data_loader_loader:
        for data, target in data_loader_loader:
            output = net(data)
            loss = F.cross_entropy(output, target)
            print('epoch: %d, loss: %f' % (epoch, loss))
    
```

TensorBoard

metrics

loss

0 2 4 6 8 10 12 14 16

0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

metrics/loss

metrics/loss/mean

0 2 4 6 8 10 12 14 16

0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4

metrics/loss/mean/1

0 2 4 6 8 10 12 14 16

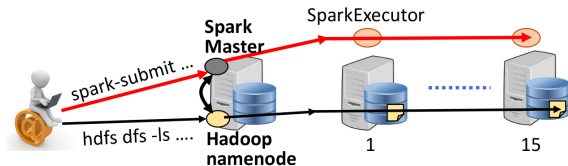
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4

metrics/loss/mean/1

0 2 4 6 8 10 12 14 16

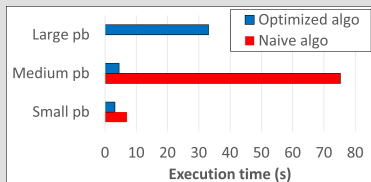
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4

metrics/loss/mean/1



Description of the use case

Design of a Map-Reduce with limited network traffic

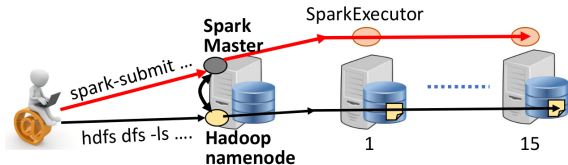


Bad algorithms on real distributed systems \Rightarrow huge network cost



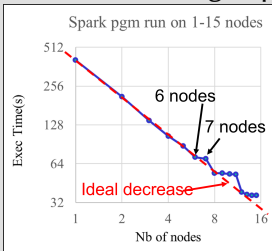
Teaching use cases

III. Distributed Hadoop/Spark on CPU cluster[2/2]

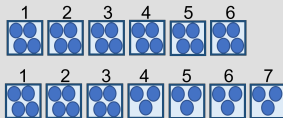


Description of the use case

Observe load balancing impact on speedup of Spark programs.



Ex: Load balancing of a **24-task** Spark pgm:



$$T(7 \text{ nodes}) = T(6 \text{ nodes}) = T(4 \text{ tasks})$$



Description of the use case

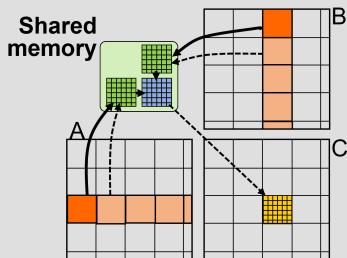
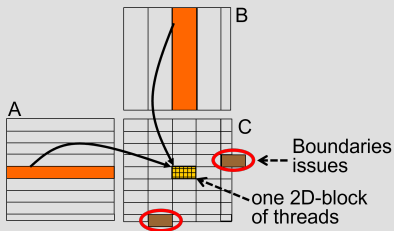
First steps with CUDA on GPU: dense matrix product

Issues:

- Grid of blocks of threads
- First CUDA kernels
- Coalescent mem. accesses

Issues:

- Use the shared memory
- Increase the coalescence
- Decrease the number of global memory accesses



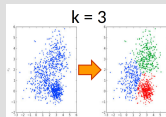


Description of the HPDA use case

K-means clustering on GPU with CUDA

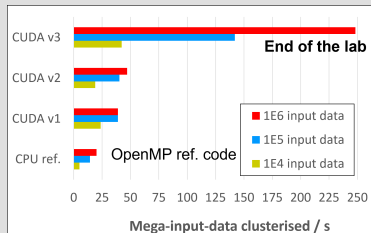
Issues adressed:

- Coalescence
- Shared memory
- Reduction
- Atomic functions



Lab: Design the right data structures and associated computations to obtain optimized kernels ... many combinations exist !

After 2 lab sessions :



Many CUDA optimizations succeed.

Thank you for your attention !

We are grateful for the support of



More details on

<https://dce.pages.centralesupelec.fr/>.

Feedback from a data center for education at CentraleSupélec engineering school, J. Fix, S. Vialle, R. Hellequin, C. Mercier, P. Mercier, J.-B. Tavernier
jeremy.fix@centralesupelec.fr