UNIVERSITÉ PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Quantum and **Distributed** Computer Science (**QDCS**) Master's Program

Master Informatique
Faculté des Sciences d’Orsay
Université Paris-Saclay

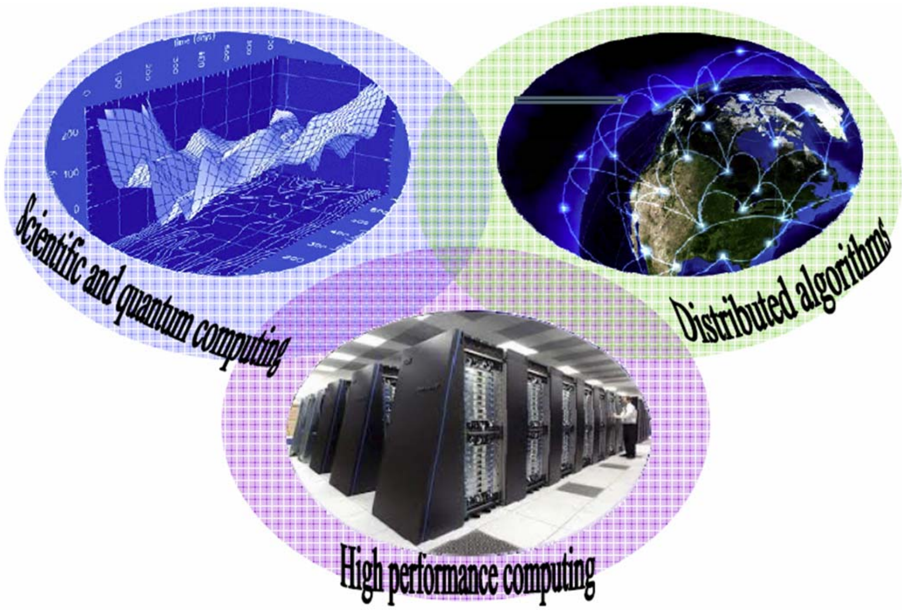
Janna BURMAN and Oguz KAYA

université PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

QDCS Master

will enable you to acquire deep knowledge in three major interconnected areas:

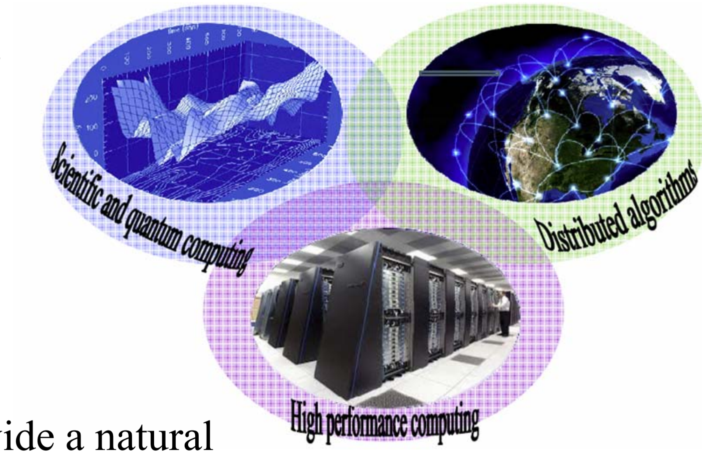
- **Distributed algorithms and systems**
- **High performance and parallel computing**
- **Quantum computing**



universite PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

QDCS Master

QDCS master's program focuses on the performance, robustness and optimization of **distributed**, **parallel** and **quantum** systems



Using advanced methods, algorithms, models, analysis and programing of parallel, distributed and quantum computing

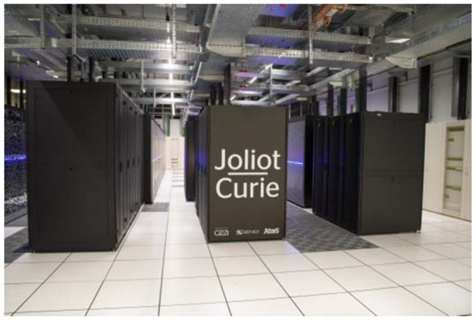
Quantum computing is a new compute paradigm and provide a natural extension to current **parallel** and **distributed** computing and systems by

- –introducing new classes of algorithms, protocols, and programming models, and providing radical performance gains with quantum parallelism
- –motivating hybrid supercomputing architectures involving classical and quantum processors in a large-scale distributed system

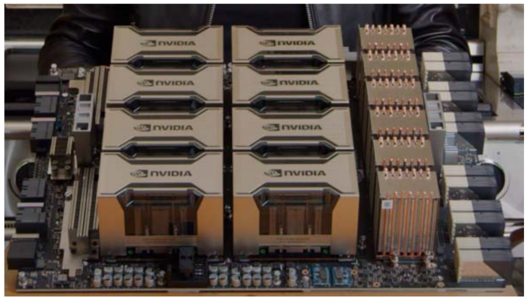
universite PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Parallel and High Performance Computing

- Big data analysis, machine learning, and large-scale scientific simulations require an immense compute power
- Processor frequency stagnates due to physical constraints (stuck at around 5-6GHz for quite a while)
- Using multiple connected machines (supercomputers) simultaneously is the only way forward for large-scale applications
- Programming, orchestrating, and exploiting the potential of such machines is poses challenges in terms of
  - algorithms
  - compute architecture
  - programming



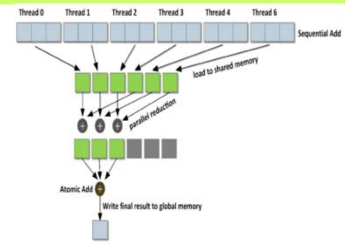
Irène Rome, 300K CPU and 655K GPU cores


Nvidia A100 GPUs in a compute node

Apple A14 chip with CPU, GPU, and accelerators

What we learn

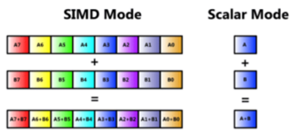
- Parallel algorithms and complexity
- Parallel computer architecture
- Multi-core parallel programming
- Distributed parallel programming
- GPU programming
- Low-level code optimization and tuning (memory, cache, vector units, ...)
- Other programmation skills (object-oriented, template metaprogramming using modern C++, compilers, scientific programming using Python, etc.)

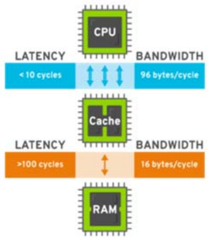


OpenMP

OPEN MPI

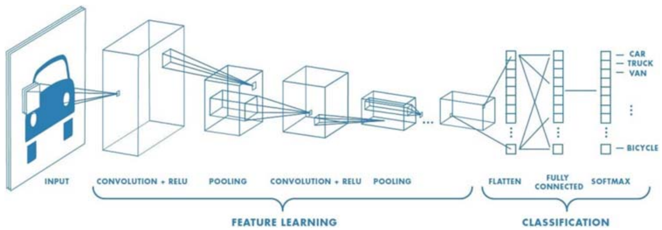
NVIDIA CUDA



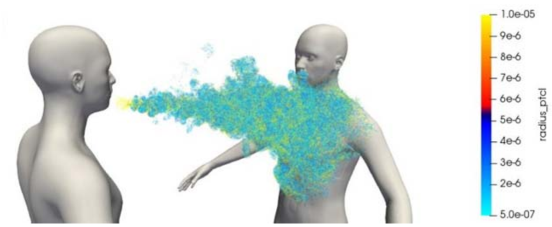


C++ 20 is here!

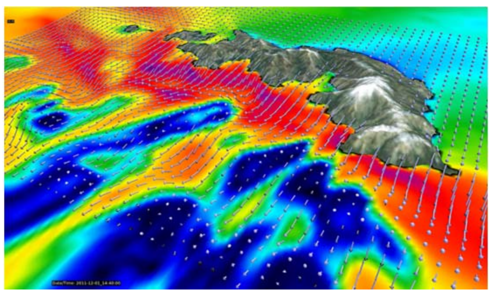
Applications



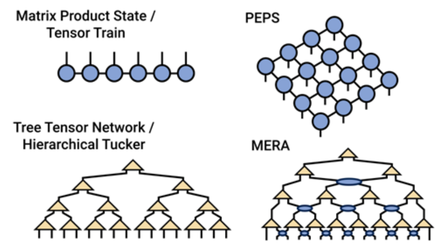
Deep neural network training + inference



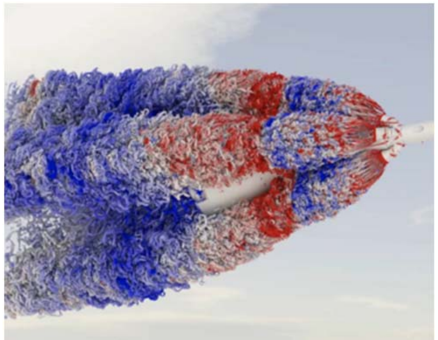
Droplet simulations



Climate and weather simulations



Tensor networks for quantum states



NASA Orion takeoff

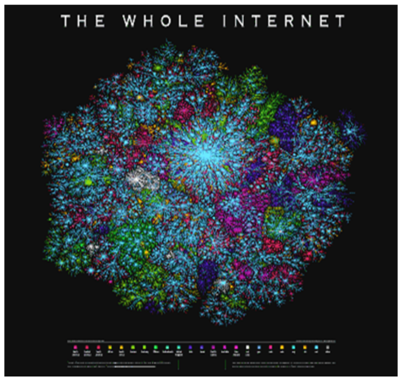
Other HPC applications

université PARIS-SACLAY

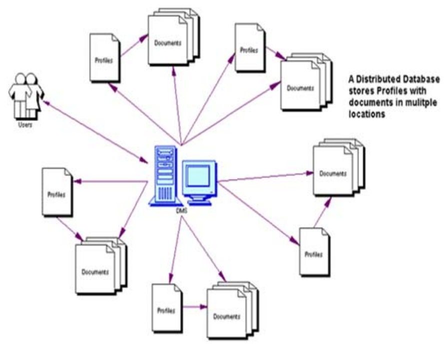
FACULTÉ DES SCIENCES D'ORSAY

Distributed Systems and Algorithms

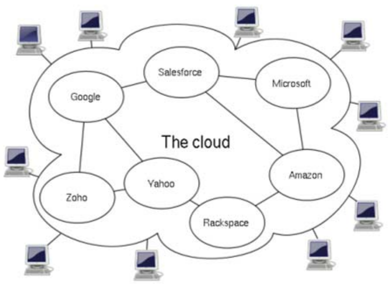
Internet



Distributed DB



Cloud



BitTorrent

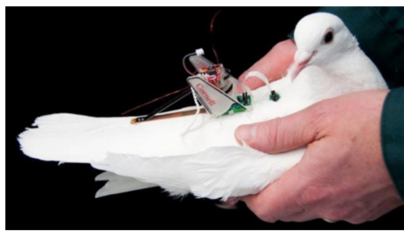
Bitcoin Net

Parallel computing

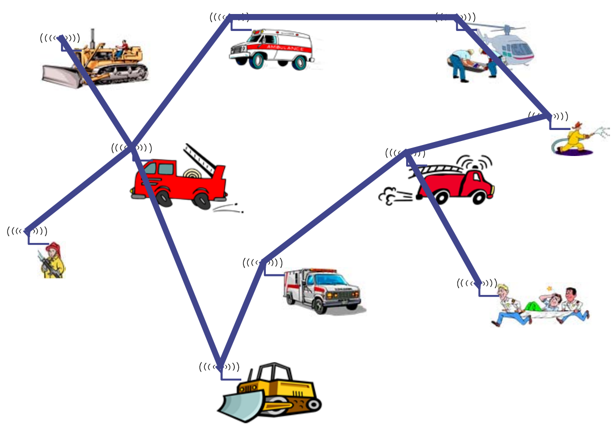
universite PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Algorithms for Distributed Systems

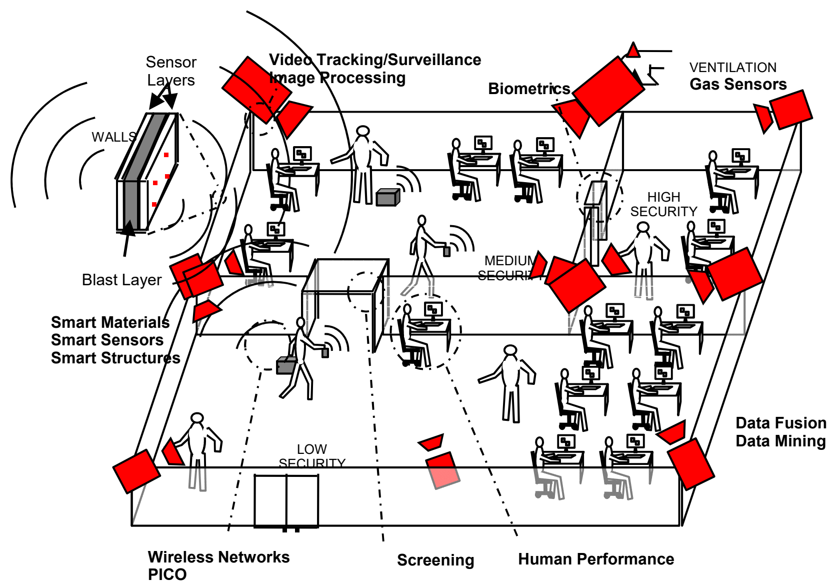
Mobile or fixed sensor networks, IoT



ZebraNet - wildlife tracking



EMMA pollution monitoring

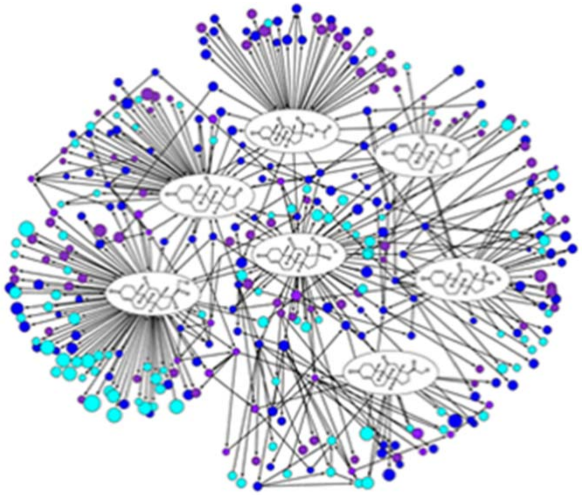


Secure Area Monitoring

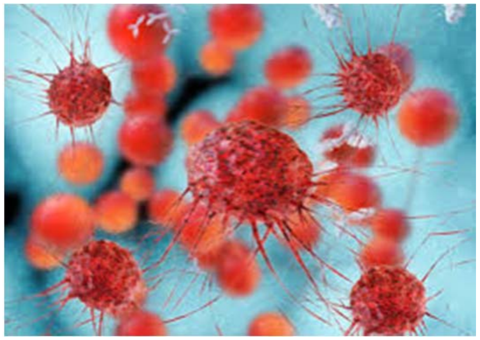
universite PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Algorithms for Distributed Systems

In Nature



Small fraction of the Organic Chemistry Network (~0.001%). Here, the nodes represent chemical compounds, which are connected by directed arrows representing chemical reactions.



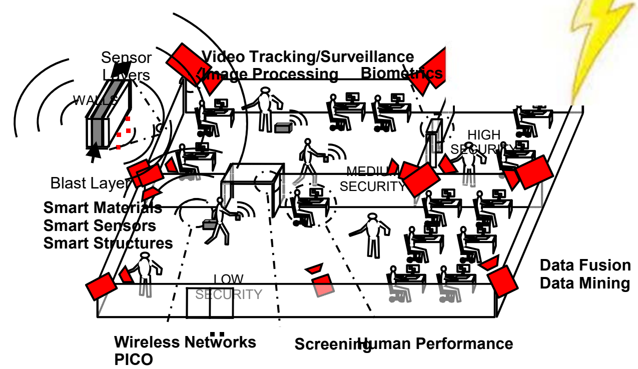
Distributed system characteristics

- The system is composed of **several computational entities**, called **processes**
- Processes are **remotely distributed and independent**
- Frequently even without any common shared memory
- The **communication is thus non-instantaneous**, done either by shared memory, or by messages (routing of the messages takes time)
- Processes must collaborate to **realize a common task**

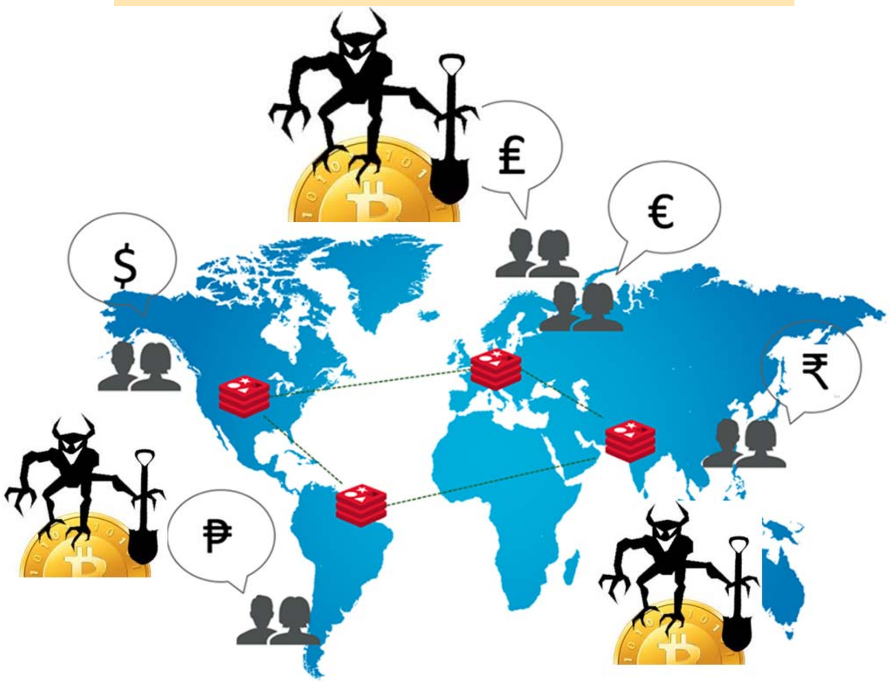
universite PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Real systems are subject to failures

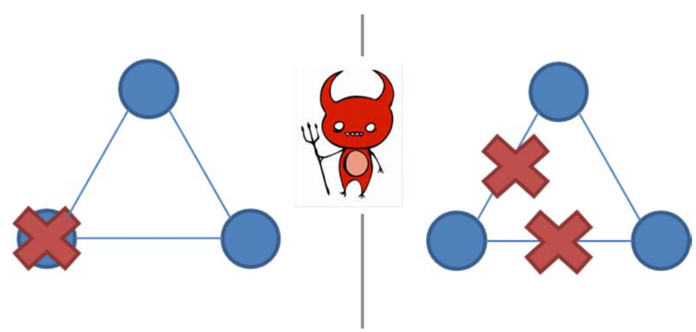
Memory, communication or program corruptions



Malicious participants

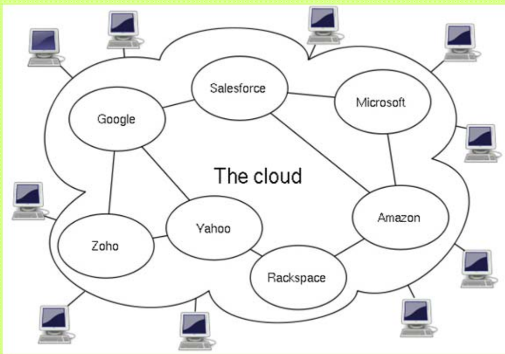


Nodes or link crashes

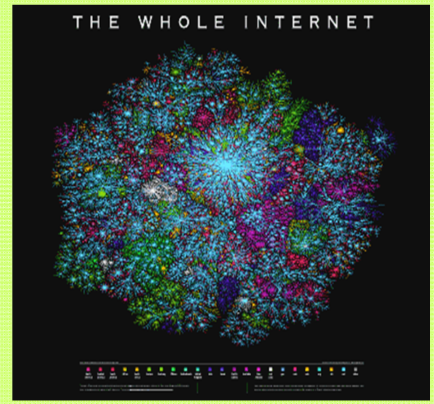


université PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Distributed Protocols/Applications



Parallel computing

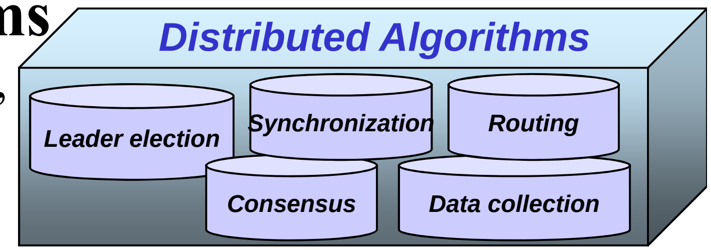


Distributed Algorithms

- *Leader election*
- *Synchronization*
- *Routing*
- *Consensus*
- *Data collection and broadcast*

Distributed algorithms

- **Distributed applications are based on a common set of fundamental problems**
  - If we can solve these problems efficiently, implement these applications efficiently.

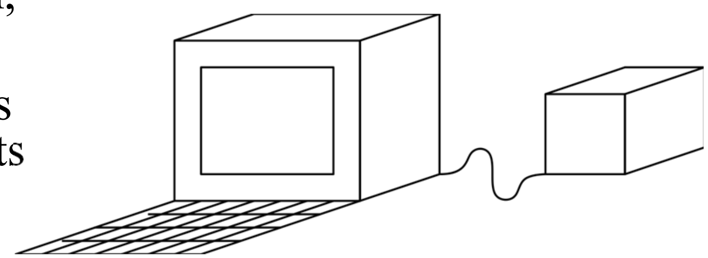


- **The goal of the domain of distributed algorithms is :**
  - to identify (extract) these fundamental problems
  - describe them formally
  - design and analyze efficient solutions/algorithms for them
  - prove their correctness and performance
  - study the limits and capabilities of distributed systems
    - impossibilities to solve a problem
    - lower bounds on time/messages/energy ...

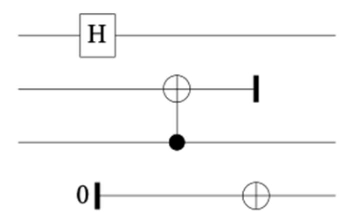
université PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Quantum computing

- Quantum computing leverages specific properties of quantum mechanics (superposition, entanglement) to carry out operations
- Potentially exponential speedup in computations (an operation on n qubits modifies 2^n coefficients simultaneously
- Exaflop supercomputers (10^{18} flop/s) will be probably beaten by a 60 qubit quantum computer, and in a more energy efficient manner (10^{-4} MWh vs 21 MWh).
- Main target domains
  - quantum chemistry, biology
  - artificial intelligence
  - cryptography
  - optimizaiton
  - computational finance
  - scientific simulations, ...



Hybrid CPU + QPU computing



Quantum circuit model

université PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Obligatory QDCS courses

distributed, HPC, quantum, optimization, programming

M1

1. [M1QDCS] Robust distributed algorithms
2. [M1QDCS] Self-stabilizing distributed algorithms
3. [M1QDCS] Parallel algorithms
4. [M1QDCS] High performance computing
5. [M1IoT] MPI programming
6. [M1QDCS] Scientific computing
7. [M1QDCS] Introduction to quantum algorithms and programming
8. [M1MPRI] Foundations of quantum information
9. [M1QDCS] Modeling et optimization of discrete systems
10. [M1QDCS] Games, learning, and optimization of complex systems
11. [M1QDCS] Object-oriented C++ programming
12. [M1QDCS] Advanced C++ programming

M2

1. [M2QDCS] Natural algorithms
2. [M2QDCS] Distributed computing by mobilg agents
3. [M2QDCS] GPU programming
4. [M2QDCS] Data parallel computing in C++
5. [M2QDCS] Tensor computations
6. [M2QDCS] Advanced quantum computing and error correction
7. [M2QDCS] Simulation of quantum processors
8. [M2QDCS] Frontiers of parallel, distributed, and quantum computing
9. [M2QDCS] Stochastic optimization
10. [M1/M2QDCS] TER (Task-parallel GPU+CPU matrix multiplication)

QDCS track generalities

- **The teaching is in English**

- Disciplinary courses block
  - **Obligatory QDCS courses** (slight personalization is possible)
  - **Optional courses** : Courses **from other computer master tracks at Paris-Saclay (AI, DS, IoT, MPRI, SETI, ArteQ)**

- Soft skill courses block: **complementary professional skill courses**
  - languages (English, French, other), communication, life in an enterprise, research training, conferences, sports.

- **Internship**/TER block:
  - **In M1** : 1+ month internship (or a summer school / training) and a TER project (Travaux d'Etude et de Recherche) in a research laboratory
  - **In M2** : **6 month internship in a laboratory or company**

université PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

Questions ?

- **M1 QDCS coordinator:** Janna Burman (janna.burman@universite-paris-saclay.fr)
- **M2 QDCS coordinator:** Oguz Kaya (oguz.kaya@universite-paris-saclay.fr)
- **QDCS Secretary:** Eva Perin (eva.perin@universite-paris-saclay.fr)
- **Distributed computing theme referent:** Janna Burman
- **High performance computing theme referent:** Oguz Kaya
- **Quantum computing theme referent:** Renaud Vilmart (renaud.vilmart@inria.fr)