Deploying a fault tolerant computing middleware over Grid'5000: performance analysis of CONFIIT and its integration with a quantum molecular docking

Luiz-Angelo Steffenel[1], Jean-Charles Boisson[1], Chantal Barberot[2], Stéphane Gérard[2], Éric Hénon[2], Christophe Jaillet[1], Olivier Flauzac[1] et Michael Krajecki[1].

[1] CReSTIC-SysCom – EA 3804

[2] Institut de Chimie Moléculaire de Reims – UMR CNRS 6229

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Outline

- Introduction
- CONFIIT
- Case study:
  - Langford's problem
  - Quantum molecular docking

Introduction

- **The FIIT class of problems ⇔**

Finite number of Independent and Irregular Tasks

- For each task:
  - Independency ⇔ no communications
  - Unpredictable execution time
  - Same algorithm

- FIIT programming:
  - How to divide the problem?
  - How to compute a given task?

CONFIIT Presentation

- *Computation Over a Network for FIIT*

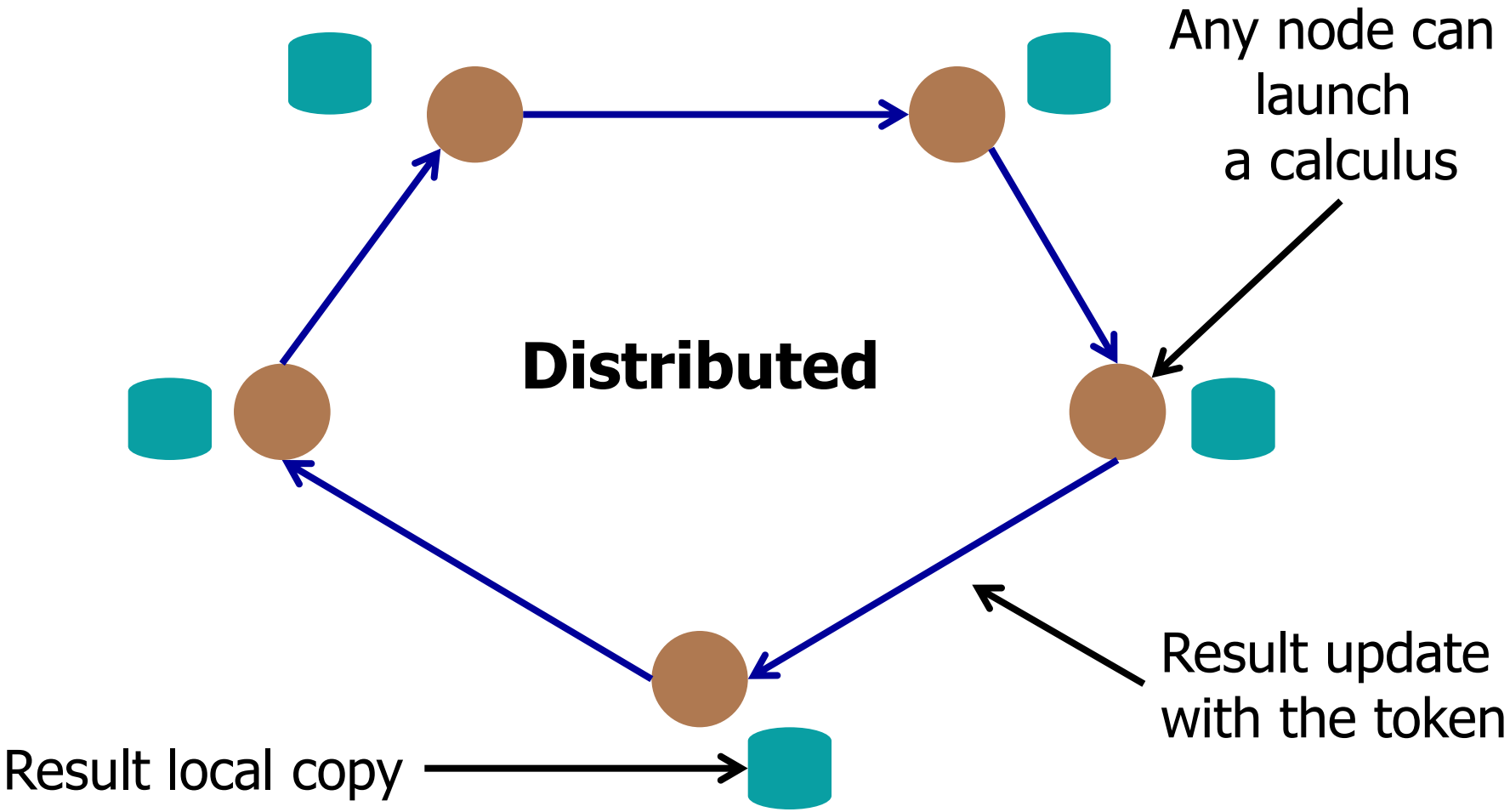
- Objectives:
  - Tasks distribution over the network
  - Tasks resolution
  - Results gathering

- Characteristics:
  - Fully distributed (peer-to-peer)
  - Heterogeneous (system, hardware, architecture)
  - Easy-to-use API

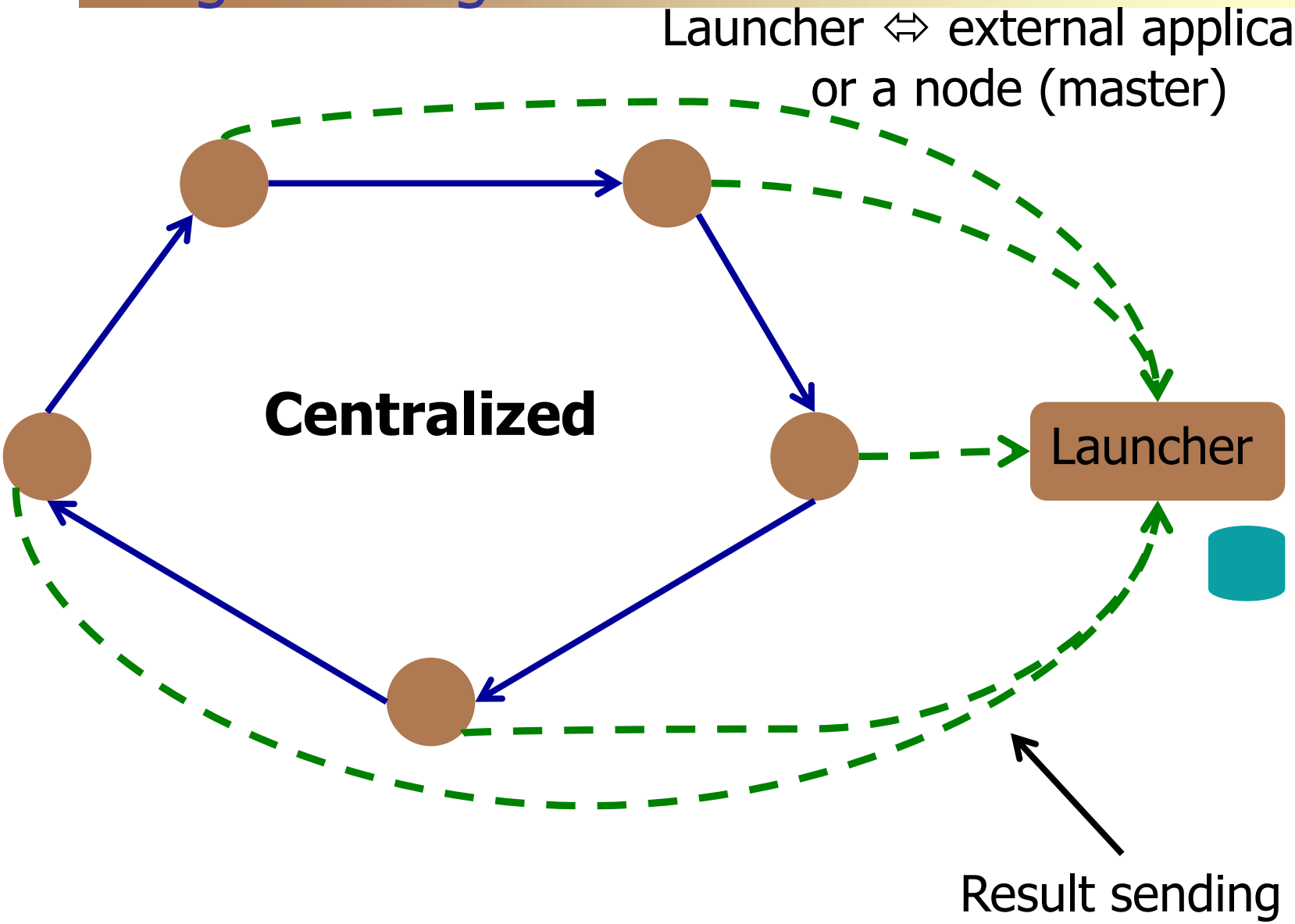
CONFIIT
Task management

- Information distribution ⇔ token circulation over a logical ring.
- Instance to solve ⇔ task array
- State of a task:
  - New ⇔ not computed
  - Locally running
  - Remotely running
  - Terminated

CONFIIT
Programming mode



CONFIIT
Programming mode



CONFIIT
Programming mode comparison

	Distributed	Centralized
Fault tolerance	++	+
Task restart	Yes	Yes (If launch node runs)
Result location	On any nodes	Launch node
Communication cost	++	+

CONFIIT
Generic application deployment

- Download the last CONFIIT version (jar file)
  http://cosy.univ-reims.fr/~lsteffenel/CONFIIT

- Start the initial daemon (community initialization):
  - java –cp Confiit-X.X.jar CDaemon –init –clean &

- Start the daemon on the other resources in order to join the community:
  - java –cp Confiit-X.X.jar CDaemon –clean –node node-01 &

- Start the instance to solve:
  - java –cp pathToJar/Confiit-X.X.jar MyApplication

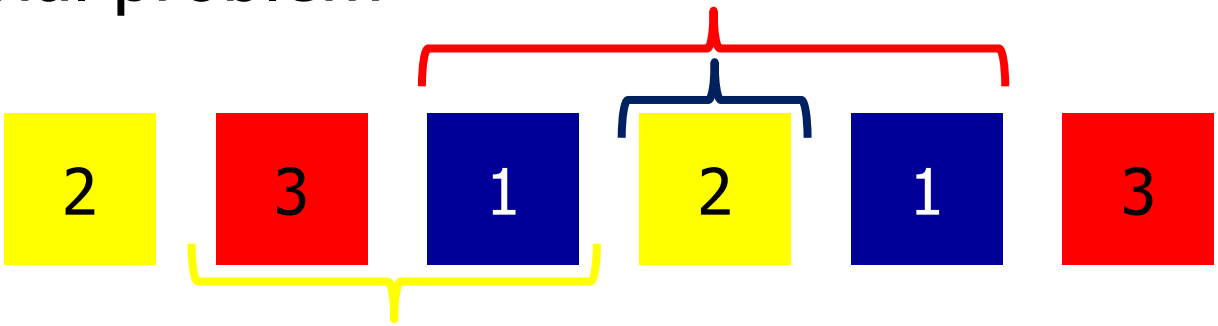
Case study 1: Langford's problem

➔ Grid use for solving large problem instances.

➔ Impact of node placement in the computation.

Case study 1: Langford's problem

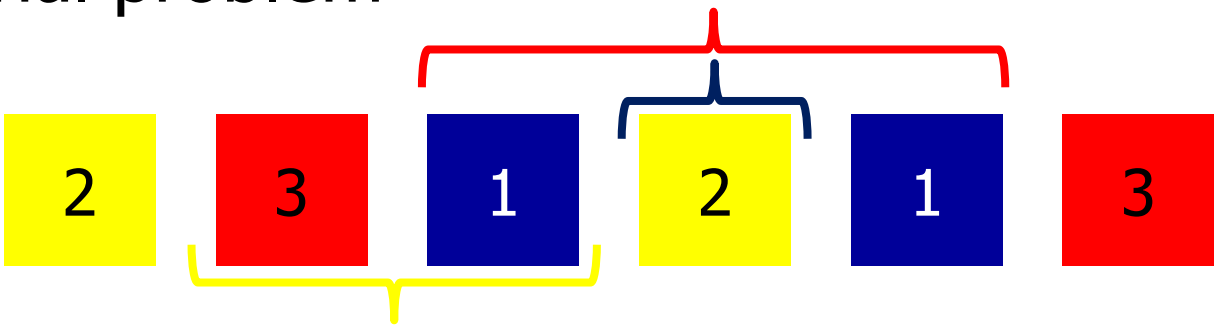
- Combinatorial problem
- L(2,3):



- Largest instances up to now:
  - L(2,19) solved in 2.5 years (1999)
  - L(2,20) solved in one week (2002)

Case study 1: Langford's problem

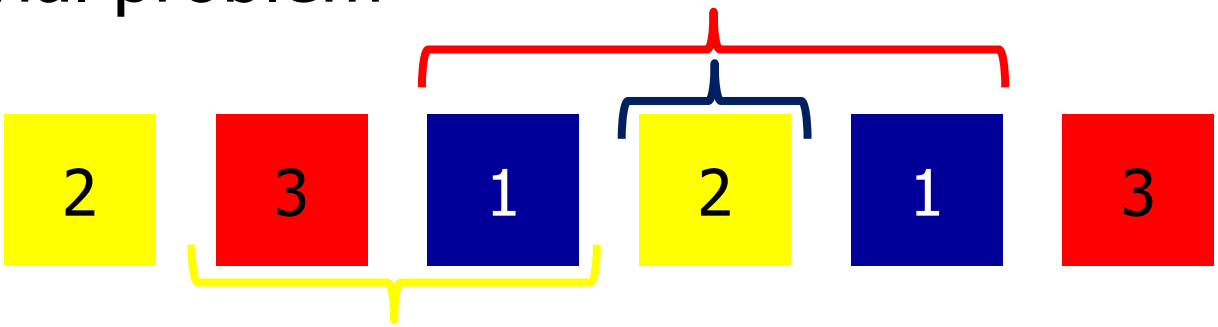
- Combinatorial problem
- L(2,3):



- Largest instances up to now:
  - L(2,19) solved in 2.5 years (1999)
  - L(2,20) solved in one week (2002)
  - L(2,23) solved in 4 days with 63 machines (2004)
    - Equivalent to ~320 days of sequential computation

Case study 1: Langford's problem

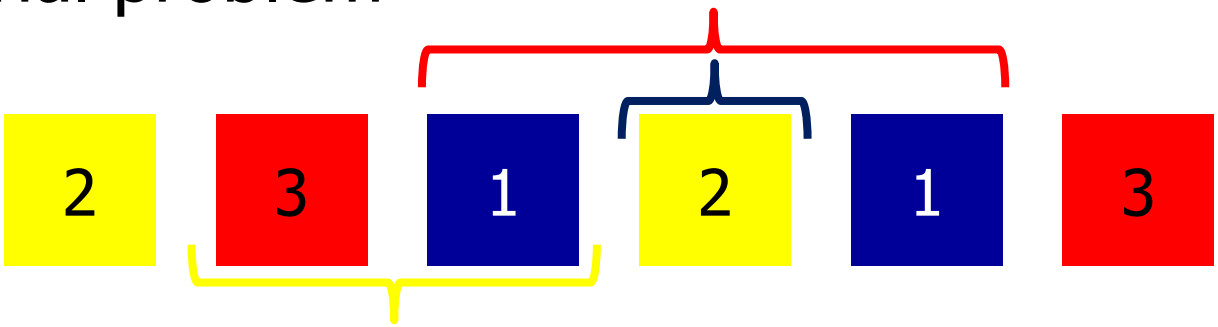
- Combinatorial problem
- L(2,3):



- Largest instances up to now:
  - L(2,19) solved in 2.5 years (1999)
  - L(2,20) solved in one week (2002)
  - L(2,23) solved in 4 days with 63 machines (2004)
    - Equivalent to ~320 days of sequential computation
  - L(2,24) solved in 94 days in a variable set of machines (2005)
    - (about 850 days of sequential computation)

Case study 1: Langford's problem

- Combinatorial problem
- L(2,3):



- Largest instances up to now:
  - L(2,19) solved in 2.5 years (1999)
  - L(2,20) solved in one week (2002)

SYSCOM team

  - L(2,23) solved in 4 days with 63 machines (2004)
    - Equivalent to ~320 days of sequential computation
  - L(2,24) solved in 94 days in a variable set of machines (2005)
    - (about 850 days of sequential computation)

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 1: Langford's problem

- Ratio between level **n** and **n-1** ≈ 5x
- There is no solution for n=25 and n=26
- So for n=27...
  - 850 days of seq. computing$_{24}$ x 5$_{25}$ x 5$_{26}$ x 5$_{27}$ =
  - **106 250 days (291 years)**

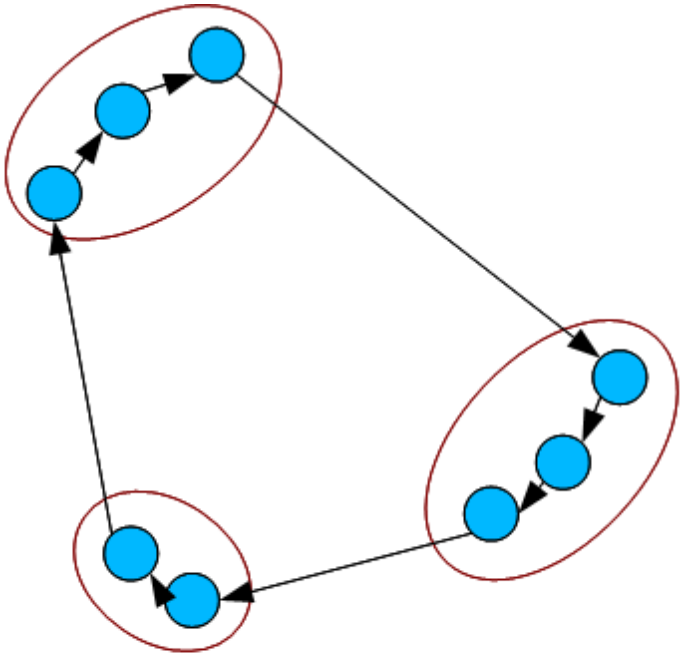
- Only way: distributed computing (grid, cloud...)
  - More than 3 months in a grid with 1000 nodes
  - In a so long run, we must be sure that the node's placement doesn't affect the overall performance

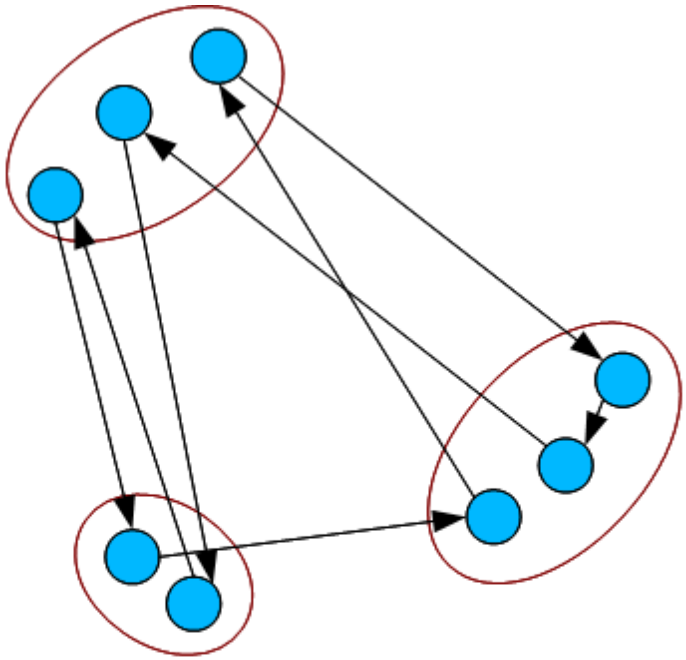
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 1: Langford's problem

- Interfacing with CONFIIT:
  - Stand-alone application designed to work with CONFIIT.
  - Distributed programming mode.
  - Two types of topology:
    - Geographical placement
    - Unordered placement

Case study 1: Langford's problem



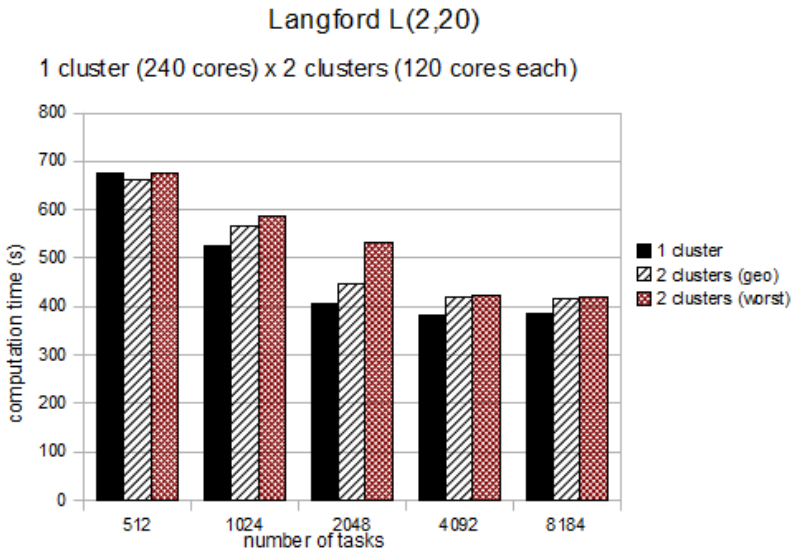


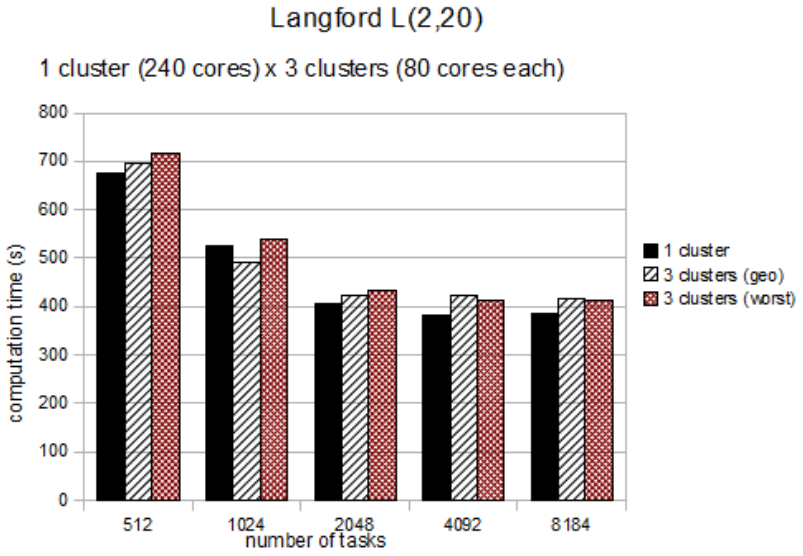
Geographical placement
Token circulation ++
Fault tolerance +

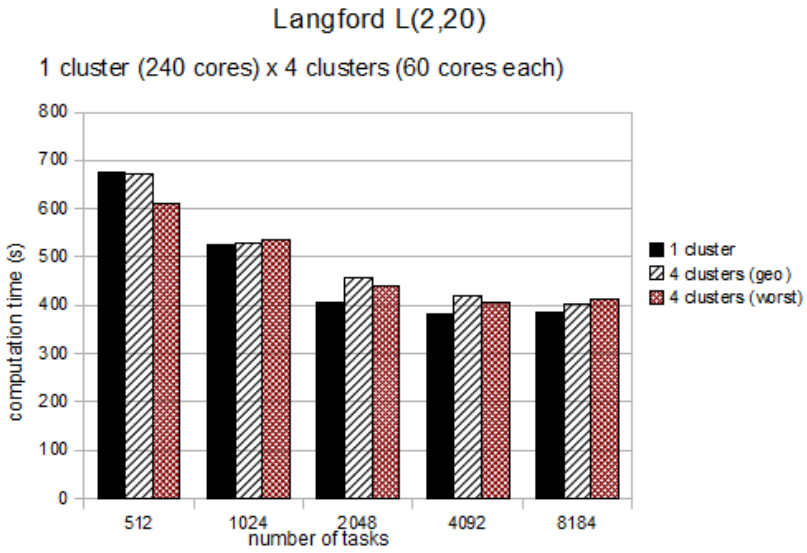
Unordered placement
Token circulation +
Fault tolerance ++

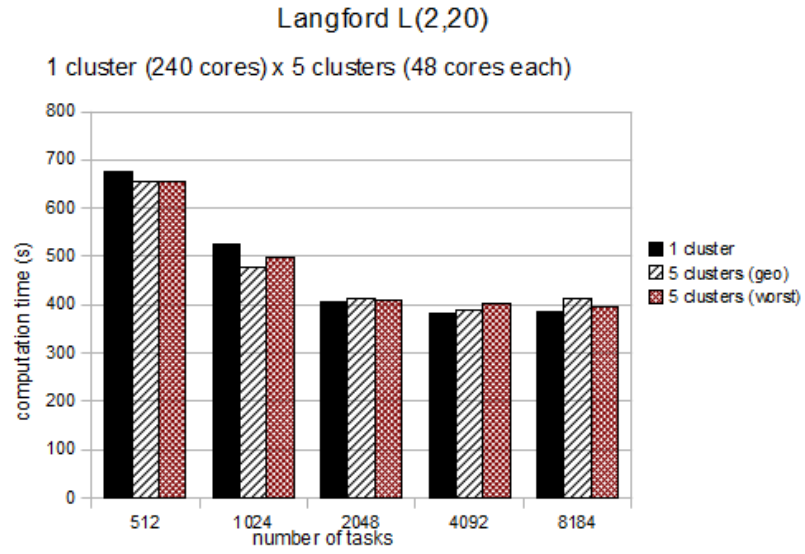
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 1: Langford's problem







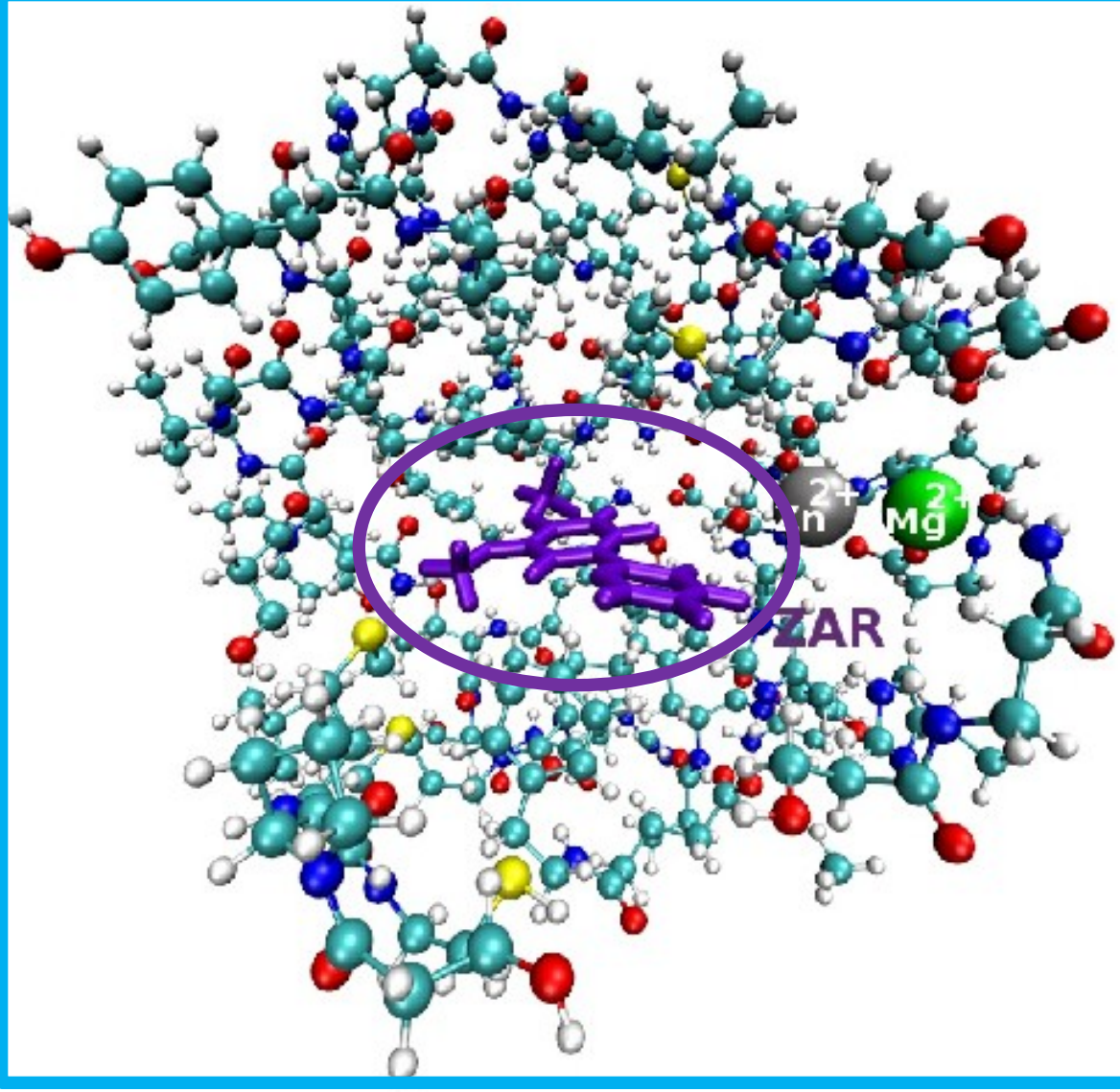


Case study 2: Quantum molecular docking
(preliminary work)

➔ Extreme resource consuming problem.

➔ Easy deployment for a non-specialist end user

Case study 2: Quantum molecular docking



Docking

||

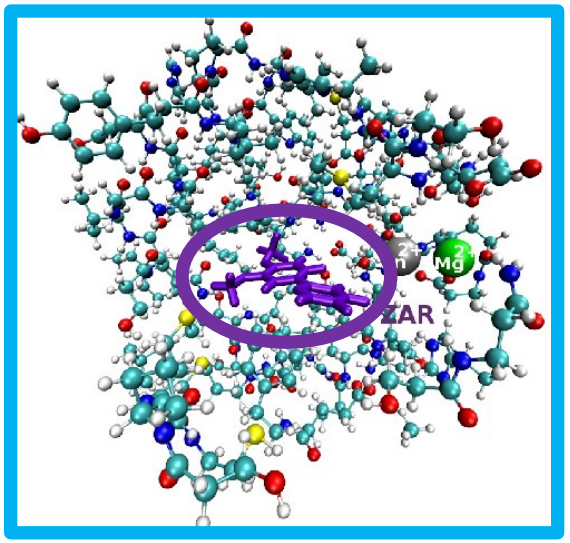
Macromolecule

+

Ligand

Case study 2: Quantum molecular docking

- The complexity of the problem relies on:
  - The flexibility of the molecules:
    - The both are rigid ⇔ rigid docking
    - The ligand is flexible ⇔ semi-flexible docking
    - The both are flexible ⇔ full flexible docking
  - The type of energy evaluation:
    - Empirical force fields
    - Quantum methods:
      - Semi-empirical
      - Ab initio



UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- The complexity of the problem relies on:
  - The flexibility of the molecules:

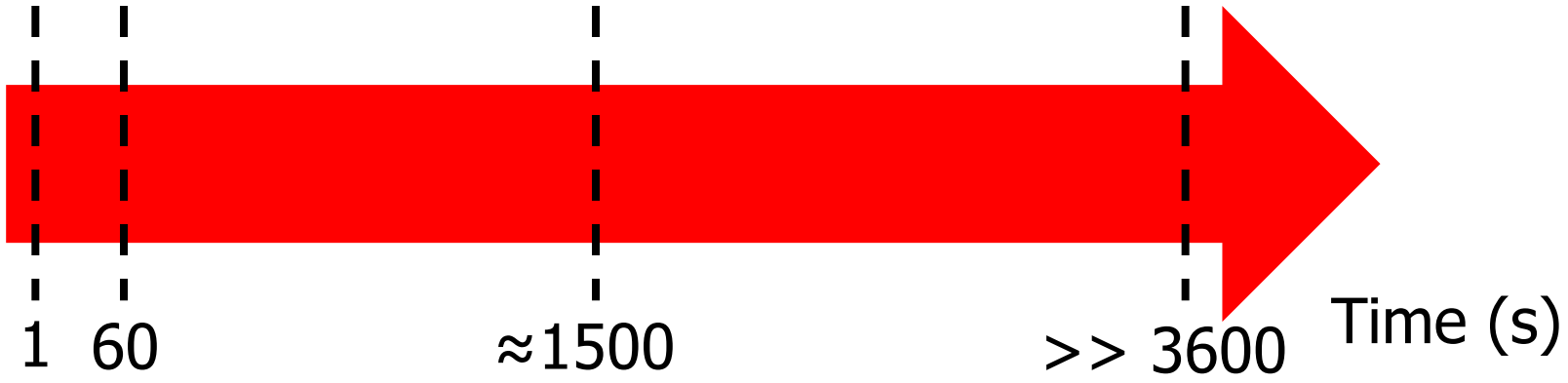
➔ impact on the search space size

  - The type of energy evaluation

➔ impact on the evaluation cost of a solution

Case study 2: Quantum molecular docking

- Evaluation cost scale for one solution:



Semi empirical quantum method

Empirical force fields:
<1s with pre-computation
<60s with live computation

Ab initio quantum method

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- Current approaches:
  - Semi-flexible docking with empirical force field
  - Pros:
    - The energy evaluation is very quick
    - Using metaheuristics allows to gain good quality solutions without exploring the whole search space.
  - Cons:
    - The force field is calibrated for specific molecular families

    ➔ A lot of system cannot be analyzed with these methods.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- Our approach:
  - rigid docking with a quantum semi-empirical method
  - Pros:
    - All the systems can be studied.
    - Using metaheuristics allows to gain good quality solutions without exploring the whole search space.
  - Cons:
    - The energy evaluation is $(very)^n$ time expensive.
      - 1000 atom system (only one solution)
        - ⇔ more than 20 min on a standard processor

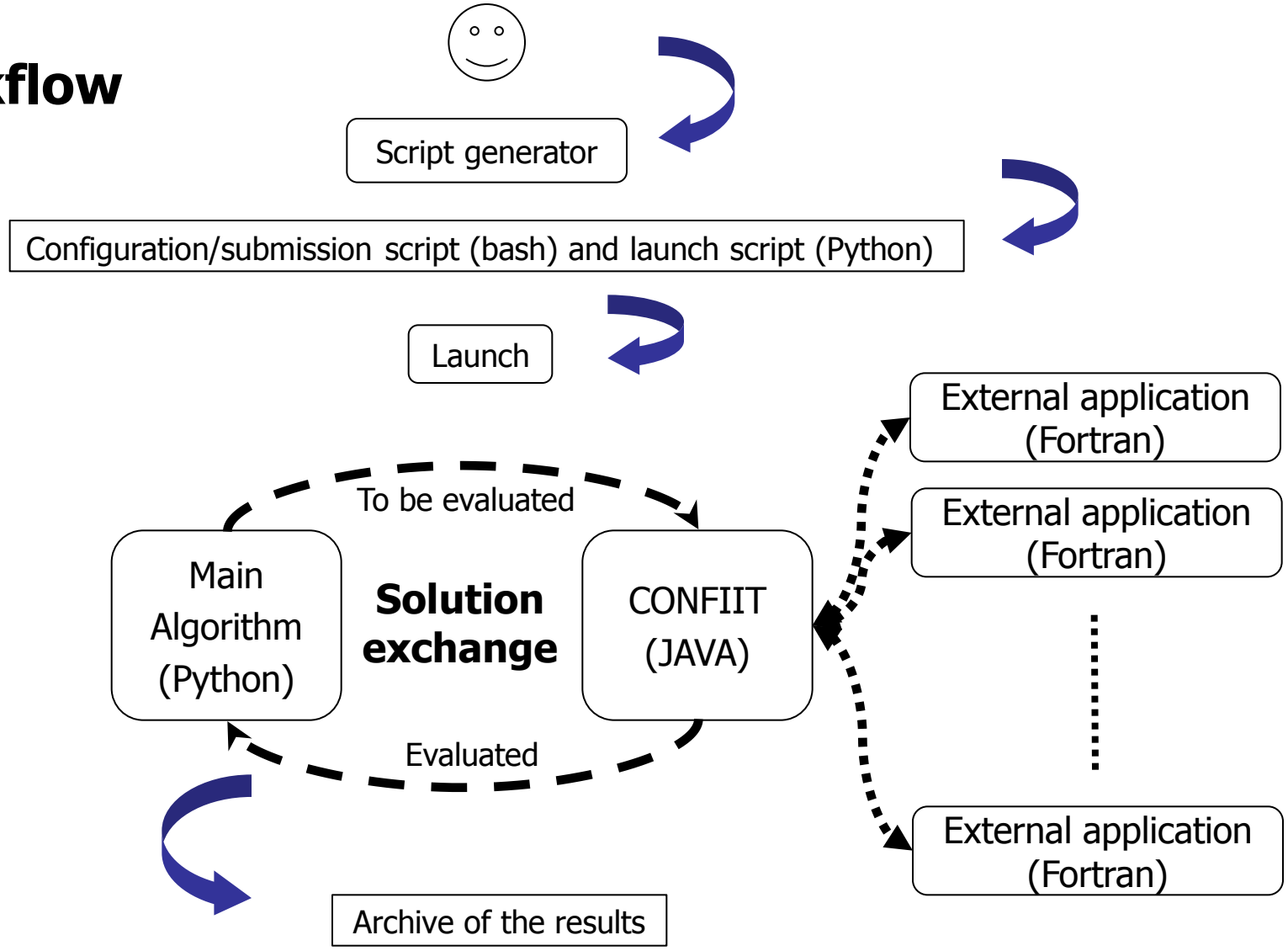
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- Interfacing with CONFIIT:
  - CONFIIT manages the evaluation of the solutions.
  - The main algorithm does not need to know how the solutions are evaluated.
  - Centralized programming mode.
  - No pre-defined node placement.

Case study 2: Quantum molecular docking

Workflow



UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- System of interest:
  - PDE4: phosphodiesterase enzyme of type 4
  - PDE4 + cAMP (cyclic adenosine monophosphate)
    ➔ AMP which has inflammatory effects
  - Stop this reaction (inhibits it) allows to relax the muscles of the breath ⇔ drug for healing:
    - Asthma
    - COPD (chronic obstructive pulmonary disease)

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- System of interest:
  - Replace the cAMP in the PDE4.
  - Existing inhibitors (4):
    - Good impact on the diseases
    - A lot of secondary effects
  - Aim of the chemists ⇔ find a new inhibitor
    - With qualities equivalent to others existing inhibitors.
    - Without their drawbacks

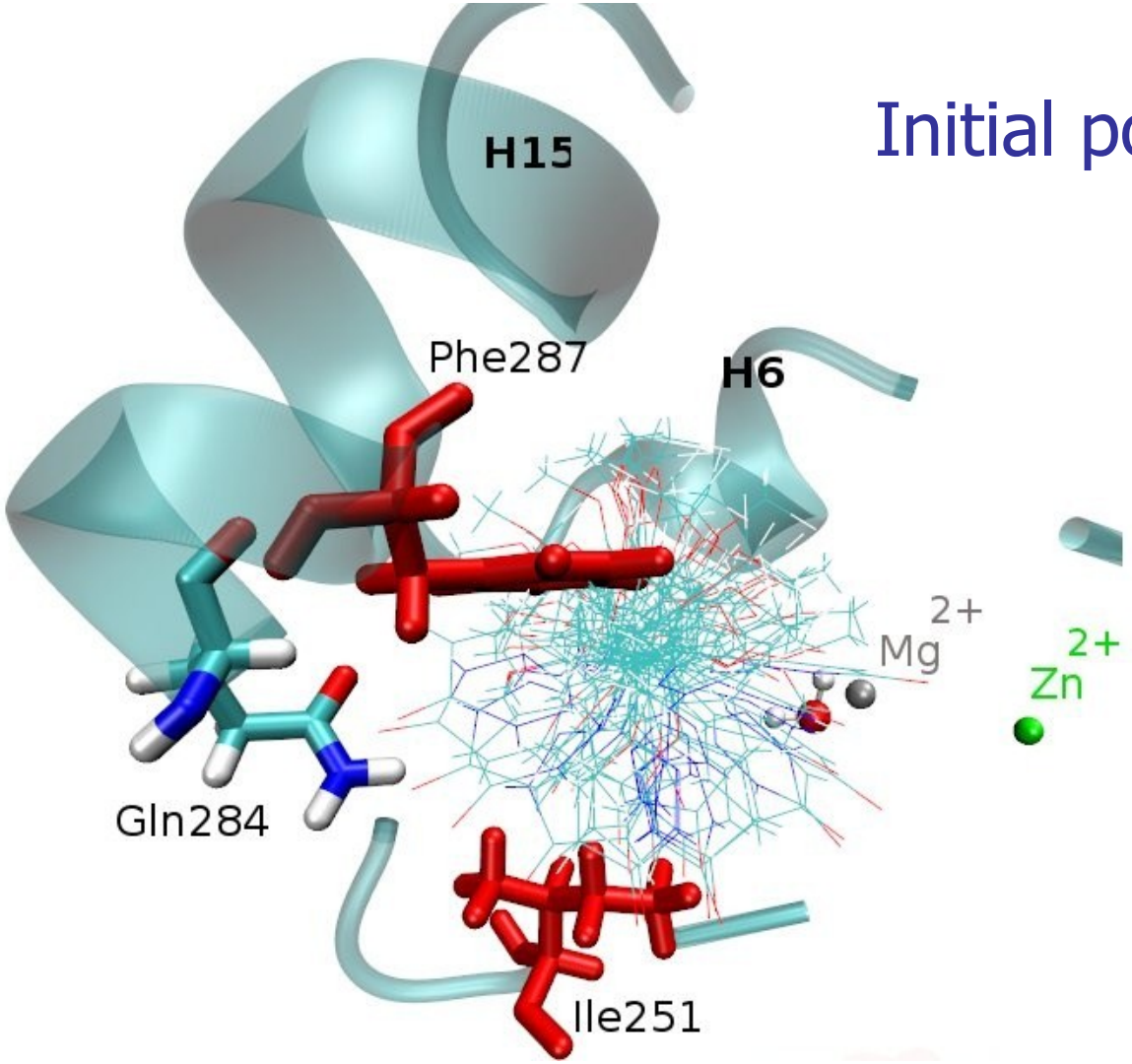
Case study 2: Quantum molecular docking

- First test ⇔ approach validation
  - The main algorithm is a genetic algorithm.
  - One level of paralleling ⇔ a pool of solutions is evaluated by a set of cores.
  - 10 runs with a population of 32 solutions during 500 generations:

➔ 3 days of computation with high performance processors (Xeon® CPU X5650 2,67GHz) on the ROMEO platform for **each** run.

ROMEO
CENTRE DE CALCUL
CHAMPAGNE-ARDENNE

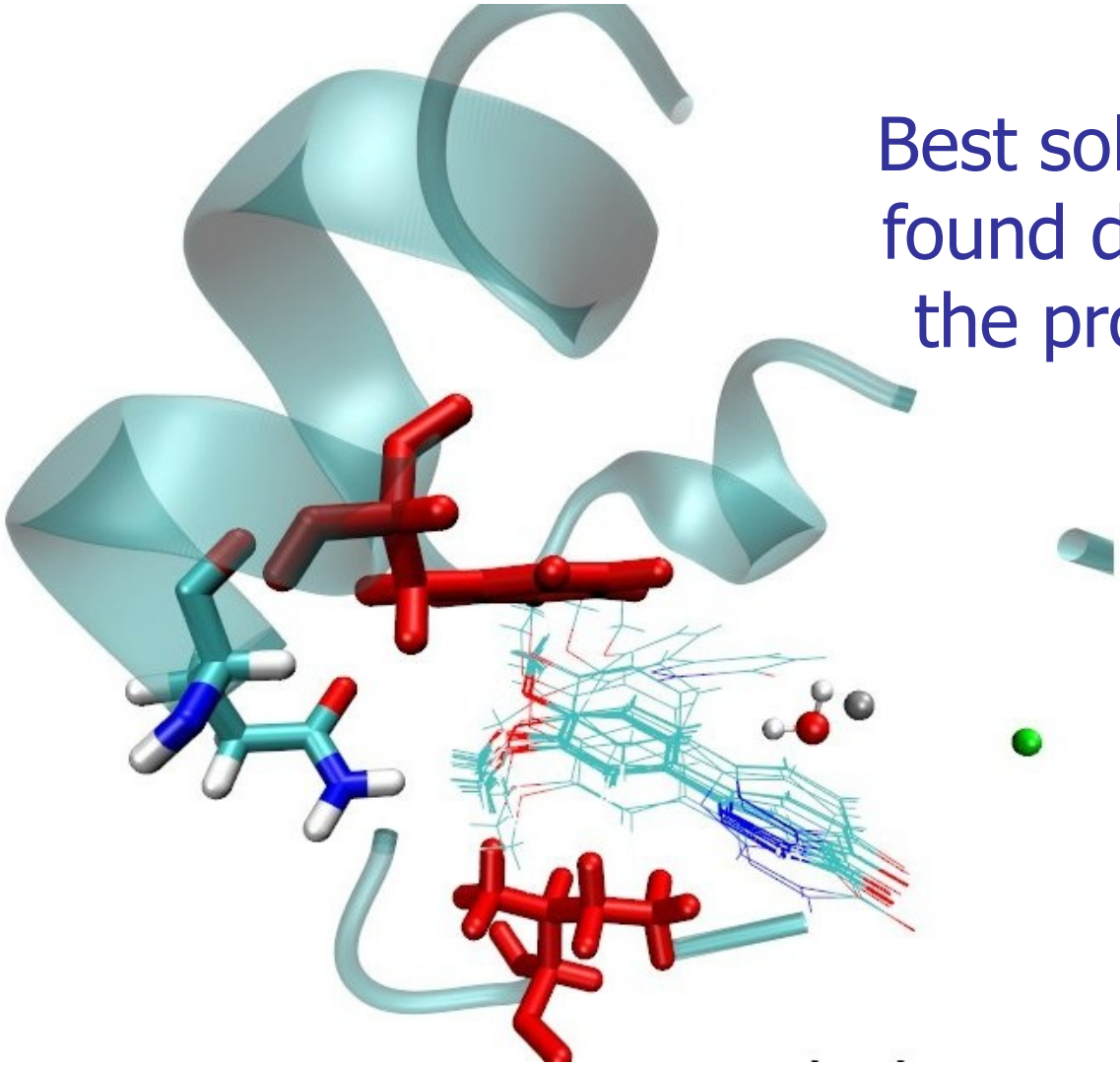
Case study 2: Quantum molecular docking



Initial population

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

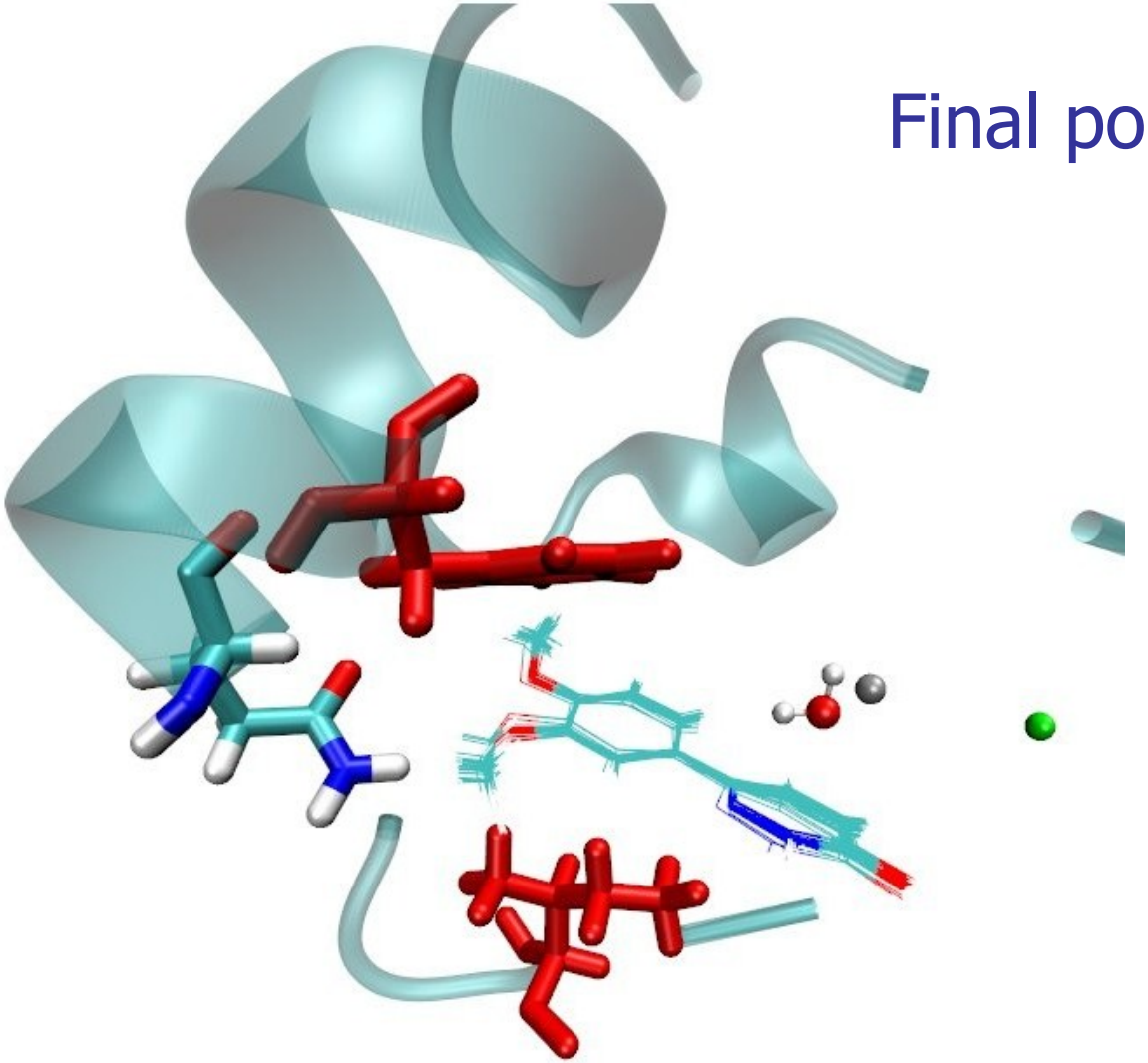
Case study 2: Quantum molecular docking



Best solutions found during the process

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking



Final population

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- Second test ⇔ scalability test
  - Deploying the docking with a large population (>500).
  - Use of Reims Grid'5000 node:
    - 44 nodes of 24 AMD Opteron™ Processor 265
      ⇔ 1056 computational cores
  - Stopping criterion ⇔ a number of iterations without improvement (with a maximum of 10 000 iterations).
  - One week of computation ➔ valid results.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- First test:
  - Valid results.
  - Too restrictive for a real docking study.
  - Too long.
- Second test:
  - Valid results.
  - Slower execution due to the master node: genetic operators + I/O for backup between the evaluations.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- Real tests:
  - Bigger number of solutions …
  - … with two levels of paralleling:
    - Population-level: a population of solutions is distributed on clusters of computational resources (geographical distant or not).
    - Solution-level: a solution is evaluated by a set of computational resources.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- Real tests:
  - Bigger number of solutions …
  - … with two levels of paralleling:
    - Population-level: a population of solutions is distributed on clusters of computational resources (geographical distant or not).
    - Solution-level: a solution is evaluated by a set of computational resources.

Large scale grid computing required

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Case study 2: Quantum molecular docking

- Next tests: why using Grid'5000 ?
  - To evaluate the dimensional aspects between the two levels of paralleling:
    - Size of the clusters
    - Communication costs
  - To verify the good interaction between the two levels of FIIT instances to solve.
  - To validate an auto-adaptive approach for an efficient charge-balancing according to the available architectures.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Conclusions

- CONFIIT is a working middleware:
  - Fully distributed, robust.
  - Heterogeneous (system, hardware, architecture);
  - With a rich programming API.
  - That can be used for real-life complex problems.
- Current improvements:
  - communication between tasks.
  - security / confidentiality.
- CONFIIT current version:

  ➔ **http://cosy.univ-reims.fr/~lsteffenel/CONFIIT**