Creatis LRMN

ThIS on the Grid

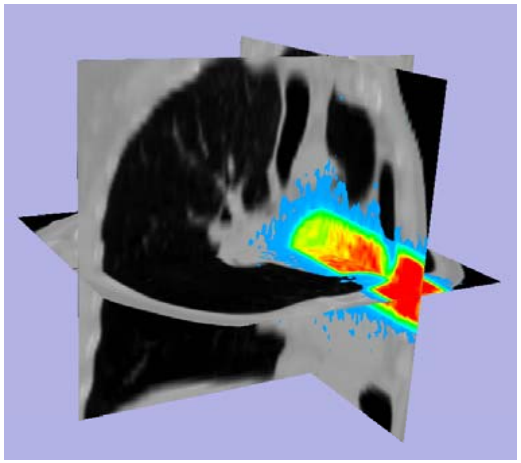
Sorina CAMARASU

Summary

- **What is ThIS ?**
- **ThIS on the Grid – The Gridification Process**
  - Why Bring ThIS on the Grid?
  - Implementation
  - Results
- **Perspectives**
- **Conclusion**

What is ThIS ?

- **A Therapeutic Irradiation Simulator**
  - Cancer treatment by irradiation of patient with beams of photons, protons or carbons
  - Simulation of the interaction between particles and matter
    - Matter ⇔ patient tumor
    - Particles ⇔ photons, carbon, protons
    - Interaction ⇔ dose distribution near tumor
  - Based on Geant4
  - Developed by: David Sarrut and Laurent Guigues
  - https://www.creatis.insa-lyon.fr/rio/ThIS



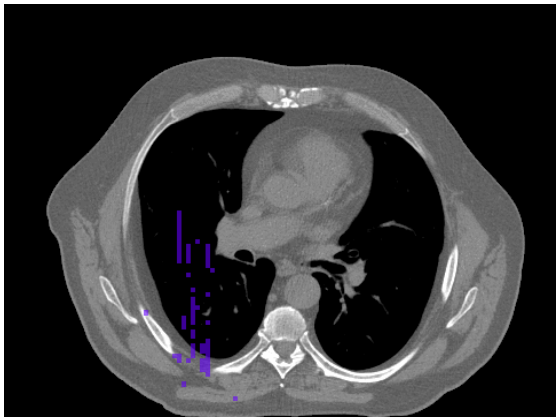
- **Aims**
  - Offer an open platform to researchers for Monte Carlo simulations optimization
  - Offer a fast and reliable simulation tool for researchers in medical physics (treatment planning) and medical imaging for treatment control
  - Produce a reference dataset (energy deposit, positron emitters distributions, ...) for non-conventional therapies (hadrontherapy).

Creatis LRMN

ThIS on the Grid
The Gridification Process

Why bring ThIS on the Grid?

- **Intensive computing**
  - Monte Carlo simulation (1 to 900 h cpu)
  - Stochastic simulation algorithm
  - 3D image pre-processing: segmentation



3D dose distribution, 700h CPU

CT image (482x360x141)

- **Data management**
  - Important input and output data
  - Input: patient images (20 to 50 Mb in 3D and up to 600 Mb in 4D), configuration and script files
  - Output: the images with the dose distribution (up to 50 Mb) and/or the phase space (up to 1G)

Implementation (I)

- **Geant4, CLHEP and shared libraries**
  - Static linking and build
- **Job submission**
  - Split the simulation into independent sub-jobs
    - e.g. a 50 M particles simulation is divided into 50 sub-jobs, each with 1 M particles to simulate
- **Parametric jobs**
  - Very similar jobs generated from one JDL file – exist with glite-wms
  - ThIS sub-jobs are parametric jobs
  - However, the current implementation does not use them
    - Problems with status and cancelation
- **The JJS Tool**
  - Developed at IN2P3 by Pascal Calvat
  - Java Job Submission – a Java application for job submission and management

Implementation (II)

- **Data management**
  - The executable and G4EMLOW libraries are distributed on SE
  - Configuration and input specific files are provided at submission
  - Output files/results are copied from the WN on the IN2P3 SRM and then on the user's machine
- **A typical script example**
  - Copy this_on_egee executable and G4EMLOW from SE on the local node
  - Copy input data from gsiftp server on the local node
  - Export G4LDATA
  - Execute this_on_egee
  - Copy output on gsiftp server
  - Clean (rm all copied files)

Results

- **Execution and waiting time vary significantly**
  - Execution time for a same type of job may vary depending on WN (computing power) from 1 to 4 hours
  - For 50 jobs submitted simultaneously, often at least one waiting job one hour after submission
  - A certain number of aborted jobs
- **Resubmission**
  - Resubmission must be taken into account
    - JJS integrates a job submission manager that resubmits aborted jobs or jobs that have been waiting for too long
    - The new WMS with glite-wms also integrates a ShallowRetryCount JDL attribute
- **ThIS has already been ported on the IN2P3 cluster**
  - For few simulations, results are retrieved faster
  - For many simulations, there may be a scalability problem

Perspectives

- **A ‘probabilistic’ implementation**
  - Currently 2 problems
    - No exact prior knowledge of the necessary number of simulations
    - Non optimal job repartition due to significant execution and waiting time variations
  - Solution
    - Jobs write their results periodically
    - A statistical uncertainty is calculated in real time
    - Jobs are done when the uncertainty threshold (or the max default number of simulated particles) is reached
    - No re-submission, no job cancelation
- **Testing and improving the scalability of our implementation**
  - Data management is currently rather centralized
- **Web portal**
  - A friendly-user web portal would ease the execution of ThIS on the grid for people who are not familiar with grid technologies

Conclusion

- **ThIS is a typical application that can benefit from grid computing technologies**
  - Its parallelization is natural, as simulations can be run independently
- **Everything is not perfect yet, work is to be done**
  - Data management
  - Job submission optimization
  - User access – web portal
- **Running ThIS on EGEE allows for a considerable speed-up in computation time**
  - Brings simulations lasting weeks (too much for a dose estimation) on one PC to only a few hours on EGEE

Creatis LRMN

Thank You for Your Attention!

Questions?