# Alien and GSI

Marian Ivanov

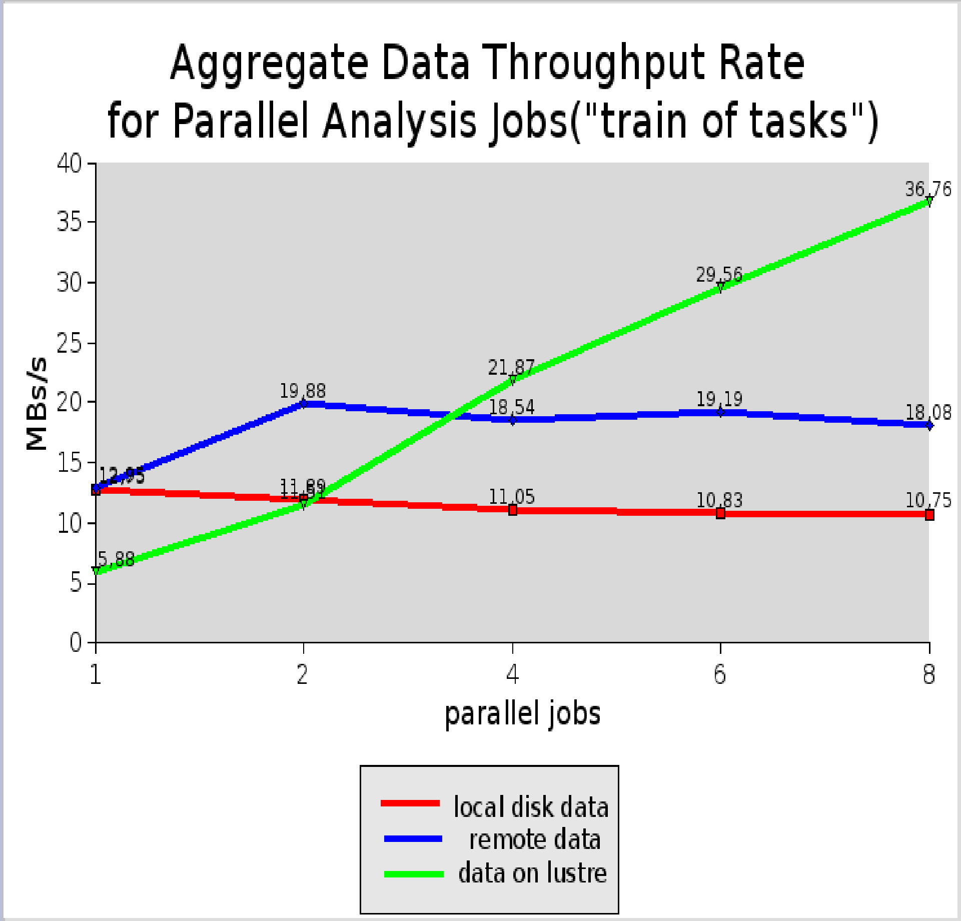
## Outlook

- GSI experience
- Alien experience
- Proposals for further improvement

## GSI - SE

- 1 Goal  - Central data transfer – mirroring
- Working ~ 20 Tby of data at GSI SE
- Missing part ?
- Central strategy of data cleaning
  - Current status (SE is almost full, 20 out of 27 Tby used)
- Central data management (on level of one SE)
  - How to check data volume – The data registered using cryptic physical names
  - Creating file list (per SE) – eg  find  -SE GSI::SE yyy xxx
  - Zombie files?
  - Quotas per user?

# GSI SE



- **Extensive test of IO configuration performed at GSI (Mischa Zynovyev, see next presentation)**
- **Test Example:** Difference in speed between parallel jobs (train of tasks) processing local data, parallel jobs processing data from remote fileserver, and parallel jobs processing data from Lustre cluster. One analysis job (“train of tasks”) processes 10000 files of 100 events each.

## GSI - CE

- GSI farm – biggest part 8 core machine
- Sharing:
  - GRID-ALIEN
  - Batch
  - Proof
- Optimal configuration
  - 1-2 slots per IO intensive jobs – currently PROOF
  - Other - CPU intensive – Simulation, reconstruction
- Analysis jobs (IO) vs Simulation and MC Reconstruction (CPU)
  - Is it possible to regulate ratio between IO and CPU intensive jobs ?
- Reconstruction job using of Raw data (IO or CPU)

## Grid experience

- What can go wrong  (with non zero probability) will go wrong
  - Input data (temporally) not accessible (staged)
  - Output SE (temporally) not accessible
  - Not proper environment on Computing elements (impossible  to compile macro, par files)
  - + Other run type errors
  - +

## Grid stability

- Input data access
  - Dynamic staging – Part of data available on “official” web site is not available  anymore
  - SE – temporary not available ( e.g. planned maintance)
  - Part of the SE not available (e.g. One file server not responding - GSI problem in February)
  - De synchronization in the File catalog (e.g. change of the File server name – current problem at GSI)
- Warning example
  - If jobs use the n input files and probability that file is not accessible is x, than probability that your job do not crash is
    - p(n,x) = (1-x)^n
  - 100 jobs, 1% ==> 36%

## Grid stability

- Condition changes dynamically in time
  - Goal – Get the status of components needed for jobs at the moment closest to the actual execution time
  - Optimally during job splitting
- Consequences
  - Gain – No waste of computing resources
  - More resources for Job splitter - negligible
    - The time to check orders of magnitude smaller than actual computation time
- This way the user-developer will be aware of possible problems (feedback time ~ minute(s)) and can react without waiting for the jobs to fail on the worker nodes.
  - The fraction of available data should be a parameter in a jdl file

## Proposal – Job submission algorithm (0)

- 1. Linear loop over file list
  - Check availability of files
  - Make a list of computing elements (CE) close to the available data
- 2. Linear loop over selected CE
  - Check the environment for given type of job
  - Select allowed, forbidden CE
- Split input list to the 2 list
  - Available, Non available
  - Optionally, provide both lists to the user
- Continue with default splitting

## Proposal – Job submission algorithm (1)

- How to develop an algorithm
- 1 phase
  - Use preprocessing scripts for JDL+xml (input data and output SE availability)
  - New information needed in jdl -pre validation script to check environment before submitting actual job
    - The pre validation scripts (without data input) have to have priority
    - At minimum try to compile and use general HalloWorld.par
- 2 phase
  - Integration of tested algorithm to the Job splitter