MANAGING LARGE SCALE COMPUTING FOR MPD EXPERIMENT*

Krzysztof Dygnarowicz

Faculty of Electronics and Information Technology
Warsaw University of Technology, Warsaw, Poland
kdygnaro@gmail.com

Jakub Zieliński, Łukasz Sawicki

Faculty of Physics, Warsaw University of Technology, Warsaw, Poland

(Received February 1, 2021)

As the part of the TeFeNICA, we received tasks associated with physical analysis, large scale computing and data management. The main goal of our project was to create scripts in Bash or programs in C/C++ and generate events which could be used to analyse. The next step was launching our work on other clusters. The article shortly describes what tools can be used to achieve these targets.

DOI:10.5506/APhysPolBSupp.14.621

1. Introduction

One of the challenges we must face during the NICA project is the amount of data we want to collect, process and storage. Data should be accessible to all members of collaboration and would allow physical analyses. With many clusters available, we have lots of different Linux distributions (CentOS, Ubuntu), queue systems (SLURM, Sun Grid Engine) or the same software but in different versions from the ones used during computing. At some point, we must unify it and give a tool to run tasks independently from the platform.

2. Big data problem

During creating the technical design of NICA network and computing infrastructure at JINR, there had been an estimated volume of data that will be created during the experiments.

* Presented at NICA Days 2019 and IV MPD Collaboration Meeting, Warsaw, Poland, October 21–25, 2019.

TABLE I

Plans in amount of data produced by NICA [1].

NICA subsystem	Technical data rate [GB/s]	Event rate [kHz]	Event size [MB]	Full event size [GB]	Mean data transfer rate [GB/s]	Data volume [TB/24 hours]
Accelerators						
2019–2020	0.5				0.1	4
> 2020	1.5				0.3	10
BM@N						
2019–2020		30	0.5	15	20	100
> 2020		50	0.7	35	100	300
MPD						
2021–2022		0.1	1	0.1	10	200
>2022		6	2	12	100	600
SPD						
>2023		50	0.5	25	100	1000

Assumed 1 PB of data in 2023 can be only achieved with the usage of proper tools which should be tested first (see Table I). At this moment, we have the software version of MPD detector, called MPDRoot, and can simulate impacts and reconstruction of events of colliding gold ions. This software can be used now to generate a large amount of data.

We are doing this now because we want:

- To find MPDRoot errors.
  During physical analysis of this data, the model can be easily verified and show errors.
- To make reconstruction programs.
  When MPD will start to work, we should have some basic reconstruction programs.
- To know what to implement in FPGA.
  Some analyses could show us which data is most important and where we should improve the frequency or accuracy of measurements.

3. Producing data

For this task, our group created a script to automate the job with the following outline:

- Get information of current JOBID in a queue system, use it to name the folder.

— Link generator of event (particles collision) and MPDRoot.
— Generate data (`runqmd.bash` for UrQMD generator).
— Run them by detector (`runMC.c`).
— Reconstruct the event from detector data (`reco.c`).
— Copy data to safe storage.
— Erase unnecessary data.

The script was firstly running on NICA cluster [2] which at the beginning had limitations of 200 jobs per user. The next step was to perform the script on other clusters like HybryLIT. For this, it had been modified to use with the SLURM job system. The basic commands are described in the instruction [4]. Further improvements can be achieved with grid computing.

Supercomputers can be connected in a grid where they share their resources. An example of this system is RDIG (Russian Data Intensive Grid) [3] which manages jobs using DIRAC. "DIRAC is a software framework for distributed computing providing a complete solution to one (or more) user community requiring an access to distributed resources" [5]. It allows to integrate different clusters, manage hardware and run tasks on them. Tests show that our first approach to the problem will not work in this case.

4. Conclusion

During our work, we proposed a system which uses JOBID to identify the output of jobs. After this, we decide that name of the output of the file should give full information of used generator, energy, centrality, number of events, *etc*.

This pattern is currently used on the HybryLIT cluster by A. Moshkin:

```
vHLLE-UrQMD_AuAu_11.5GeV-06.6-10.4fm_200ev_1.root
```

During generation on the HybryLIT and NICA cluster, it has been estimated that one job should work for 200 events to optimize usage of the CPU time and RAM storage. Moreover, clusters have a limitation, for example, one job could not take more than one day or access to some storage spaces can be done only indirectly through the working script. DIRAC at this point did not have standardized generators (UrQMD, vHLLE) and the user needed to install them on his own.

The instructions we prepared should be filled with basic workflow patterns and information on available modules. The next step will be receiving feedback from people who will use our instructions.

Links to programs and instructions:

https://github.com/jzielins97/NICA
https://github.com/Lukaszz99/work
https://gitlab.com/kdygnaro/tefenica_mpd_lit

REFERENCES

[1] http://mpd.jinr.ru/wp-content/uploads/2018/06/NICA_computing_TDR_1.03.pdf, access 15.03.2020.

[2] http://mpd.jinr.ru/howto-work-with-nica-cluster/, access 15.03.2020.

[3] http://ca.grid.kiae.ru/RDIG/, access 15.03.2020.

[4] http://hlit.jinr.ru/en/user_guide_eng/#_3en, access 15.03.2020.

[5] http://diracgrid.org/, access 15.03.2020.