Simulations and study of a new scheduling approach for distributed data production

Dzmitry Makatun$^{1,3}$, Jérôme Lauret$^{2}$, Hana Rudová$^{4}$ and Michal Šumbera$^{3}$

$^1$Faculty of Nuclear Physics and Physical Engineering, Czech Technical University in Prague
$^2$STAR, Brookhaven National Laboratory, USA
$^3$Nuclear Physics Institute, Academy of Sciences, Czech Republic
$^4$Masaryk University, Czech Republic

E-mail: dzmitry.makatun@fjfi.cvut.cz

Abstract. Distributed data processing has found its application in many fields of science (High Energy and Nuclear Physics (HENP), astronomy, biology to name only those). We have focused our research on distributed data production, an essential part of computations in HENP. Using our previous experience, we have recently proposed a new scheduling approach for distributed data production which is based on the network flow maximization model. It has a polynomial complexity providing required scalability with respect to the size of computations. Our approach improves the overall data production throughput due to three factors: transfer input files in advance before their processing (allows to decrease I/O latency); balancing of the network traffic (includes splitting the load between several alternative transfer paths); and transfer files sequentially in a coordinated manner (allows to reduce the influence of possible network bottlenecks). In this contribution, we present the results of our new simulations based on the GridSim framework which is one of the commonly used tools in the field of distributed computations. In these simulations we study the behavior of standard scheduling approaches compared to our recently proposed approach in a realistic environment relying on the data from the STAR and ATLAS experiments and considering the influence of the background traffic. The final goal of the research is to integrate the proposed scheduling approach into the real data production framework. In order to achieve this we are constantly moving our simulations towards real use cases, study scalability of the model and the influence of the scheduling parameters on the quality of the solution.

1. Introduction

Modern experiments in HENP heavily rely on distributed model of computations [1]. Data production (often called event reconstruction) is an essential part of such computations, while similar computational tasks can be found in many other fields. Despite the data production differs in many aspects from other types of computations (user analysis, simulations) it is often approached by general scheduling techniques, and that may lead to sub-optimal performance and increase the requirements for computational infrastructure (CPU, network, storage). For this reason, some experiments use custom setups in order to increase computational efficiency of data production for a given distributed infrastructure [2]. Such an approach raises a scalability issue, since reconfiguration of the setup in case of changes to the infrastructure (addition of new sites, resource fluctuations) requires an increased effort.
In our research we use typical properties of distributed data production in order to develop a scheduling approach that can optimize resource utilization and automatically adapt to the current state of the computational infrastructure. In this approach we use the following assumptions:

- A job duration and the size of its output can be estimated knowing the size of its input data, using statistics from previously finished jobs.
- Each input file has to be processed exactly once.
- All the input data are placed at a centrally located source site and the output has to be transferred back to that location. The underlying model also allows multiple data sources (data replication) and multiple output destinations, such cases will be studied in future research.

The task of the scheduler is to distribute input data between available computational sites considering the real network topology, bandwidth, storage space and computational throughput in order to complete processing as fast as possible. The detailed description of the developed approach can be found in [3] and [4], while here we provide a brief summary of the model (Section 2) and of the setup for simulations (Section 3). In Section 4 we present results of simulations of distributed data production studying an influence of background traffic. Section 5 provides results of simulations with Grid of a large scale (repeating the Tier-1 network of one of the largest HENP experiments) in order to verify scalability of the developed approach.

2. Model summary
Since the order of particular jobs and their assignment to a particular computational site is not important for data production, in our approach we plan the resource (CPU, storage, network) load only and then distribute input files accordingly. The planning is done cyclically, i.e. a plan for a limited planning time interval $\Delta T$ is produced at the beginning of this interval each time. The plan relies on the system state at the moment when it is created, but not on previously issued plans.

The computational Grid can be represented as a graph, where weights of links correspond to the maximal amount of data that can be transferred within $\Delta T$, calculated as $\text{bandwidth} \times \Delta T$. For planning of input transfers (Figure 1) we connect all the computational sites to a dummy sink and a dummy source to input storage(s) via dummy links. The capacities of dummy links represent constraints on storage, CPU throughput and data availability. For planning of output transfers (Figure 2) we swap the sink and the source and inverse directions of the dummy edges. In this case, their capacities impose constraints on the storage and amount of available/expected output data. We apply a network flow maximization approach [5] to both input/output problems and the solution is the amount of input/output data ($F_i$) to be transferred over each network link during $\Delta T$ in order to maximize the overall computational throughput.

The network flow maximization problem is known to have a polynomial complexity on the network size. It is an advantage since, in the general case, scheduling problems are NP-hard.

In order to execute the plan, there is a dedicated service running at each computational site, called "handler", also responsible for sending statistics and status data to the planner. As the handler receives a plan, it knows the amount of input/output data $F_i$ to be transferred over each outgoing link to the neighboring nodes. Each time a new input file arrives over one of the incoming links, the planner either submits it for processing (if there are free CPUs, see Figure 3) or, otherwise, forwards it over one of the outgoing links with a remaining flow $F_i > 0$ and then decreases this flow by the size of the file (see Figure 4). If there are no outgoing links with $F_i > 0$ the file is kept at the local queue until a CPU becomes free or a new plan is issued. In this manner the handler maintains a local queue of input files, so that a CPU can start the next job as soon as a previous one is finished, therefore, a transfer overhead is eliminated.
3. Simulation setup

For testing the proposed scheduling approach, we did perform simulations of distributed data production in GridSim [6], a commonly used tool for simulation of computational Grids. As input parameters, we have also used data collected during and extracted from data production for the STAR experiment at the KISTI computational facility [7]. The data production was performed during three months in 2014 and records of 77,000 jobs were collected. For larger simulations a larger set of jobs can be generated using Monte-Carlo selection from the original one.

In addition to the proposed approach (called PLANNER), two other standard job scheduling policies were simulated for comparison. Both of them use the same approach for allocation of CPUs to jobs – “when a CPU is free, send an input file from the Tier-0 for processing”, but data transfers are modeled differently:

**SEQ**: A network link is modeled as a space-shared resource, i.e. if multiple files are submitted for a transfer over the same link simultaneously, they are dispatched one by one in the First-In-First-Out order.

**PAR**: A network link is modeled as a time-shared resource, i.e. multiple transfers can be performed simultaneously, sharing the bandwidth of the link.
4. Simulations with background traffic

Network infrastructure is often shared between several experiments, as a result transfer latency can increase when several network activities are ongoing simultaneously. We have studied the influence of the background traffic on the efficiency of data production comparing different scheduling approaches with the help of simulations. The simulated Grid consisted of a single remote computational site with 1,000 CPUs and 14 TB disc cache connected via 1 Gbps network link to the central storage (Tier-0). Both sites were sending unrelated files of 12 GB size every 1,000 seconds to each other, the number of those files to be sent at once was changing in different simulations. The performance of the simulated scheduling approaches is compared in Figure 5. As one can see in the plot, our PLANNER shows better performance in all the simulations and the gain in performance increases as the level of background traffic grows. The PLANNER’s makespan increases just by 0.17% when changing from 0 to 0.8 Gbps of the concurrent transfer submission rate while the makespan for the SEQ and PAR is much higher corresponding to 24% and 120%, respectively. This is achieved due to file transfers in advance before computation, so that the jobs are not delayed by the network latency.

It is important to consider the maximal possible network latency when setting up the planning time interval $\Delta T$. If $\Delta T$ is too small, the transfers started at one planner cycle may be completed at the next one and that can compromise the preciseness of the plan execution. The simulations have shown that $\Delta T$ set to 6–12 hours is large enough to avoid such issues.

In future we plan to perform simulations with background traffic in more complex Grid topologies (many remote sites) and implement an additional reasoning into the planner such that it can identify network links with high background traffic and decrease the load on those links by utilizing alternative transfer paths as illustrated in Figure 6.

5. Scalability test

In order to test the scalability of the planner we have performed simulations using data on Tier-1 sites of one of the currently largest experiments in HENP. We have used statistics of 10 computational sites obtained from online monitoring tools. However, the designed parameters of such a network are well balanced and a transfer latency does not influence the
computational performance significantly. Under these conditions all the simulated approaches provide similar performance. For a better illustration of the planner abilities, we have added a dummy site called "PLUTO" to the initial Grid. The site has a significant amount of CPUs but a limited connection (1 Gbps) to the Tier-0 while it has a reasonable connection to two other Tier-1 sites. Such situation can often emerge in real life, when there are computational resources provided at sites outside of the CERN network infrastructure. The final Grid setup used in the simulations is provided in Figure 7. The amount of disk cache at sites was set proportional to the number of CPUs since we assume that only a part of the overall storage can be dedicated to data production.

The plot in Figure 8 shows how the total CPU utilization in the simulated Grid changes with time for SEQ and PLANNER scheduling approaches. Unfortunately, the results for PAR approach are not presented here as the underlying GridSim model for the parallel network mode appeared to be too computationally demanding. However, this approach has shown the worst performance in all previous simulations [3]. As one can observe in Figure 8, the PLANNER has reached 100% of the overall CPU performance since it was able to utilize all the CPUs at the "PLUTO" site using alternative (indirect) transfer paths. Such approach allows to use more computational resources outside of the primary network infrastructure. The makespan improvement in the considered case is 21%, an increase in throughput which would not have been easily possible without reasoning and a planner.

During the simulations with the realistic size of Grid an average runtime of the planner was measured. The result shows that planning for 12 hours of data production is done within 7 milliseconds. This allows to apply this approach for an online planning of data production in the real environment.

6. Conclusion
In previous work [3] we have proposed a new approach to scheduling of data production on distributed resources. GridSim simulations and data from the real infrastructure were used to test the approach and its implementation. The simulations have shown that the proposed approach systematically provides better makespan and can optimize over a wide range of infrastructure configurations in {CPU, disk, network bandwidth} phase-space compared to classic job scheduling approaches.
Figure 8. Results of simulations of the large-scale Grid. The planner achieves a higher CPU utilization due to network load balancing.

The recent research has shown that the approach can provide a significant performance improvement when a shared network infrastructure is used. It can better adapt to background traffic, since planning is done in advance and data are transferred before computations. It can also provide a network load balancing, including usage of network paths which are not always intuitive (optimization on secondary network paths), that allows to utilize sites with poor connectivity to the Tier-0 site more efficiently. Such optimization is automated based on current Grid topology and state. Simulations of data production in a Tier-1 Grid of one of the largest HENP experiments has shown that the proposed approach is scalable and can be implemented for online planning in real infrastructures.

In future we plan to continue the work on the proposed approach, conduct simulations with multiple data sources/destinations (Tier-0s), simulate data production of more experiments, implement advanced reasoning about background traffic and study an influence of internal planner parameters (such as $\Delta T$). The final goal is to integrate the planner into existing Resource Management Systems and deploy it into the data production framework of the STAR experiment.

Acknowledgments
This work has been supported by the Czech Science Foundation (13-20841S, P202/12/0306), the MEYS grant LG15001 and the Office of Nuclear Physics within the U.S. Department of Energy. Furthermore, the authors would like to acknowledge the Institute of Physics of the Czech Academy of Sciences for kindly providing access to the GOLIAS computer farm for running simulations.

References
[1] I Bird, P Buncic, F Carminati, M Cattaneo, P Clarke, I Fisk, M Girone, J Harvey, B Kersevan, P Mato, R Mount, and B Panzer-Steindel. Update of the Computing Models of the WLCG and the LHC Experiments. Technical Report CERN-LHCC-2014-014. LCG-TDR-002, CERN, Geneva, Apr 2014.
[2] Jan Balewski, Jerome Laurent, Doug Olson, Iwona Sakrejda, Dmitry Arkhipkin, et al. Offloading peak processing to virtual farm by STAR experiment at RHIC. Journal of Physics: Conference Series, 368(012011), 2012.
[3] Dzmitry Makatun, Jérôme Laurent, Hana Rudová, and Michal Šumbera. Distributed data production planning for High Energy and Nuclear Physics. Journal of Scheduling, 2015. (Submitted for publication).
[4] Dzmitry Makatun, Jérôme Laurent, Hana Rudová, and Michal Šumbera. Model for planning of distributed data production. In Proceedings of the 7th Multidisciplinary International Scheduling Conference (MISTA), pages 699–703, 2015.
[5] R K Ahuja, T L Magnati, and J B Orlin. Network flows: theory, algorithms, and applications. Prentice Hall, 1993.
[6] Rajkumar Buyya and Manzur Murshed. GridSim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. *The Journal of Concurrency and Computation: Practice and Experience (CCPE)*, 14, 2002.

[7] Levente Hajdu, Lidia Didenko, Jerome Lauret, J Amol, W Betts, H J Jang, and S Y Noh. Automated finite state workflow for distributed data production. In *ACAT*, 2016. (Submitted for publication).