Overall quality optimization for DQM stage in High Energy Physics experiments

N Benekos[1], M Parra-Royon[2] and J M Benítez[3]

[1] Dept. of Experimental Physics - EP-NU, CERN, Geneva, Switzerland.
[2,3] Department of Computer Science and Artificial Intelligence,DiCITS, DaSCI, IMUDS. University of Granada. E.T.S. de Ingenierías Informática y de Telecomunicación, Granada, 18014, Spain.

E-mail: [1]nektarios.benekos@cern.ch, [2]j.m.benitez@decsai.ugr.es, [3]manuelparra@decsai.ugr.es

Abstract. Data Acquisition (DAQ) and Data Quality Monitoring (DQM) are key parts in the HEP data chain, where the data are processed and analyzed to obtain accurate monitoring quality indicators. Such stages are complex, including an intense processing work-flow and requiring a high degree of interoperability between software and hardware facilities. Data recorded by DAQ sensors and devices are sampled to perform live (and offline) DQM of the status of the detector during data collection providing to the system and scientists the ability to identify problems with extremely low latency, minimizing the amount of data that would otherwise be unsuitable for physical analysis. DQM stage performs a large set of operations (Fast Fourier Transform (FFT), clustering, classification algorithms, Region of Interest, particles tracking, etc.) involving the use of computing resources and time, depending on the number of events of the experiment, sampling data, complexity of the tasks or the quality performance. The objective of our work is to show a proposal with aim of developing a general optimization of the DQM stage considering all these elements. Techniques based on computational intelligence like EA can help improve the performance and therefore achieve an optimization of task scheduling in DQM.

1. Introduction

The experimental research of multi-messenger accelerator-driven particle physics, studying from the largest-scale structures in the observable, to the most fundamental particles, and accelerated and non-accelerated based High Energy Physics (HEP) facilities must be supported by sustainable software (s/w) and hardware (h/w) that address tremendous technical challenges. A fundamental part of the HEP data chain is related to Data Acquisition (DAQ) and Data Quality Monitoring (DQM). DAQ/DQM are key parts in this software chain where the data are acquired, processed and analyzed in the early stages of the HEP experiments. DQM aims to fill the gap between the fast, high-bandwidth on-line monitoring with a limited and fixed CPU budget, and off-line processing which has access to substantially more resources but is much less agile. Hence, DQM is an indispensable tool to identify problems with the experiments and to reduce data loss. In this sense, DQM provides an information interface for quality monitoring management through measurements, parameters, or histograms.

The tasks performed in DQM involve a very large computational load (related to time consumption and computing resources), so they must be carefully selected to provide results

with adequate performance constraints. The tasks performed in DQM can be abstracted as a flow of operations that are applied to the data coming from the DAQ, with the objective of delivering processed outputs and results of the state of the experiment. In this article we present a proposal for the optimization of the general data processing in the DQM stage that allows to improve the performance of the execution of workflows in aspects such as the use of computing resources or the processing time of operations. With this proposal, a decision making tool for DQM is obtained, through which it is possible to select how the processing will be done in the most optimal way considering all the objectives of the study.

2. Problem description

DQM is a key stage of information processing in the early phases of virtually all experiments in the HEP context. As shown in Figure 2, multiple algorithms are performed to compute results for scientists and engineers. The general idea is to execute a complete workflow consisting of smaller interconnected algorithms, grouped in tasks, that share data inputs and outputs (see Figure 1). The processing elements in DQM are as follows:

Algorithms - It refers to algorithms performed in the DQM workflow such as *FFT* or the identification of *Region of Interest*. Algorithms can be executed on different architectures and computing resources (CPUs, GPUs, FPGAs/ASICs [6], Cloud providers, ...).

Task - It corresponds with a set of algorithms that perform a function or operation, for example the validation of a procedure, the calculation of some parameters or the obtaining of graphical results of the DQM.

Workflow - DQM perform a set of computationally intensive operations that generally result in a complete workflow, containing different tasks (T) and algorithms (A) in each step.

In addition, all these elements are subject to parameterization, constraints and inter-dependencies among them. The computational cost of the overall process is high, which means that the workflow must be carefully scheduled in order to provide light processing time and balanced resource consumption. In this sense, an adequate planning of the execution of the set of tasks and algorithms, shaping a workflow, can improve the general performance. We are faced with a combinatorial problem of type NP-Complete [1], where there is a wide search space and to obtain an optimal global solution becomes a very complex task. To solve this type of problems different proposals for resource planning have been approached, particularly with great success the proposals based on Evolutionary Algorithms (EA) [3]. Thus, we think that a conveniently designed EA would also work well for the problem of concern. In the following sections we define all the necessary elements to consider.

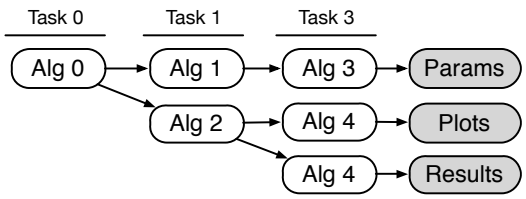

Figure 1. Example of a workflow composed of tasks and algorithms, which produce a set of results in the same way that in DQM stage.

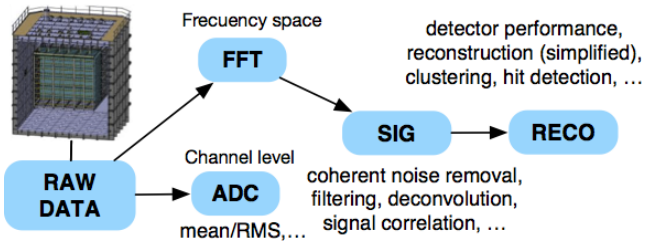

Figure 2. DQM processing stage in *protoDUNE* [8] where it comprises a group of operations that shape a data processing workflow.

3. Methodology

The goal of the proposal is to offer a decision-making tool that allows optimizing the processing workflow described in section 2.

3.1. Multi-objective optimization

To solve optimization problems, algorithms are used to find a solution that maximize or minimize the value of a function for a given problem. In many real problems and in particular resource planning problems, different objective functions are involved. To tackle this problem, multi-objective algorithms have been designed [5, 2], allowing several objectives to be optimized at the same time. The solutions to a multi-objective optimization problem are the set of non-dominated solutions called the *Pareto set* [11]

In this context, the goal of optimization is to determine how, when and where the processing for the DQM stage will be optimally executed. This depends on many criteria that can be contradictory or conflicting and in addition have associated constraints. The general mathematical model for specifying a multi-objective optimization problem consists in finding a vector $x^* = [x_1^*, x_2^*, \cdots, x_n^*]^T$ that satisfies m constraints ($i = 1, 2, \cdots, m$) of $g_i(x) \geq 0$, and p constraints ($i = 1, 2, \cdots, p$) of $h_i(x) = 0$ and finally optimize the vector function $f(x) = [f_1(x), f_2(x), \cdots, f_k(x)]^T$ where $x = [x_1, x_2, \cdots, x_n]^T$ is the decision variable vector.

3.2. Evolutionary algorithms

Meta-heuristics based on EA are widely used in resources scheduling with constraints [10] offering a good balance between promising solutions and computational performance compared to other traditional proposals [5]. A proposal based on the multi-objective optimization with EA for the problem described in section 2 has been developed.

The basic principles of EA, as well as the general working scheme of operation and algorithms are defined in different published works [4, 5, 2]. In this type of problems an individual (or chromosome) encodes a solution to the optimization problem where all the details are specified about what, when, how and where each operation will be performed. The algorithm manages a set of solutions, called population. The overall quality of the solutions is expected to improve along successive generation of populations through the operations of selection, crossover and mutation.

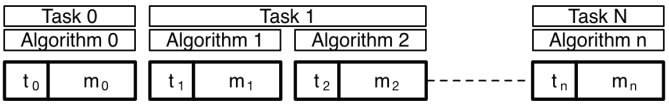

Figure 3. Solution proposal for DQM workflow scheduling with time (t) and mode (m).

4. A multi-objective Evolutionary Algorithm for DQM optimization

The modeling of this type of algorithms for DQM optimization involves the aspects developed in the following subsections.

4.1. Representation of the solution

It is necessary to represent the DQM workflow (tasks, algorithms, dependencies and resources) (Figure 1) as a chromosome (individual or solution) that is manageable by the EA. The solution

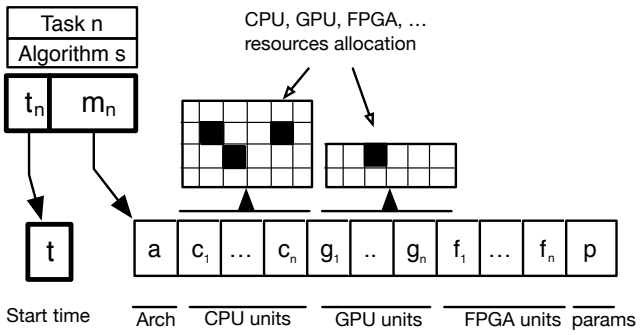

Figure 4. Detail of the components into which the execution mode is divided. *Arch* is the target architecture used. *Units* refers to the set of the infrastructure available from *PilotJobs* [7], *virtual nodes* o *bare-metal*. Then, *params* corresponds to groups of hyper-parameters of the algorithm.

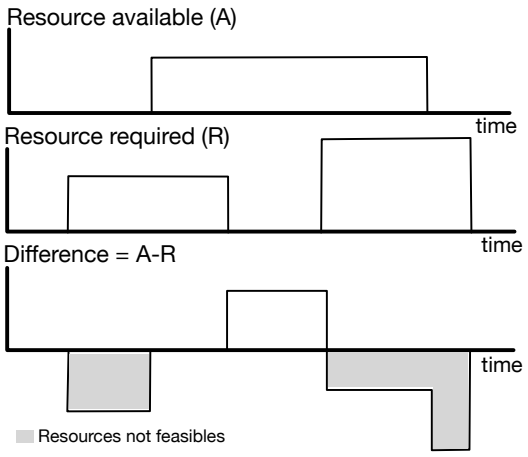

Figure 5. Example of resources availability, requirement, and feasibility profiles. Negative segments indicates parts of schedule is not feasible.

representation proposal is composed of two elements, start time (t) and mode of execution (m) as shown in the Figure 3. The time t corresponds to the start of the task or algorithm with respect to its predecessor in the hierarchy of the workflow scheme (see Figure 1), and the mode m is the set of resources used/required for each algorithm. The resource set (m) is composed of a vector (r) with: a) the type of architecture where the algorithm is executed (CPU, GPU, FGPA, Cloud provider and others), b) the number of computing units of the type of architecture available, c) the group of parameters of the algorithm execution. The details of the mode components within a solution are shown in Figure 4.

4.2. Genetic operators

Genetic operators are functions used to generate new solutions from the current population. Operators for initialization (IO), crossover (CO), mutation (MO), repair (RO) and selection (SO) have been considered. IO makes a random allocation of t and m values. The CO combines two solutions to give a new one, merging a part of one solution with another part of the other solution from one random point. The MO changes a part of the solution randomly, taking valid values within the range of resources for each part of m and values of t. RO is necessary when solutions violate integrity restrictions on resource allocation or use, in this way the OR changes the solution to meet all constraints, and it is applied to a percentage of the population in each iteration. SO selects which of the solutions will remain in the next generation [9].

4.3. Objective functions

The measurement of the performance of the solutions corresponds to two parts: a) the satisfaction of the constraints, and b) the performance of the scheduling in terms of time and use of resources of the DQM workflow.

4.3.1. Constraints. Resource scheduling in DQM is subject to different constraints, such as the number of computing resources, data dependency between tasks, or their availability at any given time, including the provision of algorithm implementations on different infrastructures. In Figure 5, an example of modeling of constraints due to resource and time availability can be seen. The degree to which constraints are violated determines how feasible the schedule is.

4.3.2. Objectives. Two objectives have been considered. They are to be minimized:

(i) Time of the workflow execution (T_w): it is the total estimated time of a complete workflow. Seeing the DQM task scheduling as an acyclic directed graph (DAG) $G = (V, E)$, the total time is the largest path cost in G, calculated as: for each operation $v \in V$ in linearized order $T_{ops}(u) = max_{(u,v)\in E}\{T_{ops}(u) + T_{op}(u, v))\}$, being T_{op_s} the start time with respect to the preceding task in the workflow, T_{op} the set of operations to be performed in the workflow such as $T_{op} = T_{e_i} + T_{l_i} + T_{t_i}$, where T_e the estimated execution time of the algorithm, T_l the estimated latency time and T_t the time to transfer data.

(ii) Resource utilization (R_u). It is the indicator of the total resources used by the execution of the algorithms, calculated as follows: $R_u = \sum_{i=0}^{n} R_{op_i} = \sum_{i=0}^{n}(R_{cpu} + R_{gpu} + R_{fpga})$, being n the total number of operations, R_{cpu}, R_{gpu} and R_{fpga} the resource use indicators of the execution of the algorithm on the different selected architectures.

5. Conclusions

Data processing in DQM is a critical stage in identifying problems and reducing data loss in early processing stages of HEP experimentation. In order to mitigate this problem, the proposal developed allows an optimization of the entire workflow in DQM, with two clear objectives, on the one hand to reduce the execution time of DQM processing tasks, offering better planning of them, and on the other hand, the minimization of the use of computing resources used by the workflow. With this proposal, is possible to carry out a global optimization of the workflow data processing in DQM, allowing to use it as a decision making tool for scientists to decide more efficiently the DQM processing stage.

Acknowledgments

This work was supported by the Research Projects *P12-TIC-2958* and *TIN2016-81113-R* (MINECO - Gov. of Spain).

References

[1] B. H. Arabi, Solving NP-complete Problems Using Genetic Algorithms, 2016 UKSim-AMSS 18th International Conference on Computer Modelling and Simulation (UKSim), Cambridge, 2016, pp. 43-48

[2] Coello, Carlos A. Coello, Gary B. Lamont, and David A. Van Veldhuizen. Evolutionary algorithms for solving multi-objective problems. Vol. 5. New York: Springer, 2007.

[3] Goldberg, D. E., & Holland, J. H. (1988). Genetic algorithms and Machine Learning. Machine Learning, 3(2), 95-99.

[4] Holland, J. H. (1992). Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press.

[5] Marler, R. Timothy, and Jasbir S. Arora. Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization 26.6 (2004): 369-395.

[6] Musa, L. (2008, September). FPGAS in high energy physics experiments at CERN. In 2008 International Conference on Field Programmable Logic and Applications (pp. 2-2). IEEE.

[7] Nilsson, P. (2008). Experience from a pilot based system for ATLAS. In Journal of Physics: Conference Series (Vol. 119, No. 6, p. 062038). IOP Publishing.

[8] The DUNE Collaboration, The Single-Phase ProtoDUNE Technical Design Report, [arXiV: 1706.07081].

[9] Deb, Kalyanmoy, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6.2 (2002): 182-197.

[10] Reeves, C., & Rowe, J. E. (2002). Genetic algorithms: principles and perspectives: a guide to GA theory (Vol. 20). Springer Science & Business Media.

[11] Zitzler, Eckart, and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Transactions on Evolutionary Computation 3.4 (1999): 257-271.