Resource Utilization by the ATLAS High Level Trigger during 2010 and 2011 LHC running

Elliot Lipeles, Rustem Ospanov and Doug Schaefer, on behalf of the ATLAS collaboration

E-mail: schae@cern.ch
University of Pennsylvania, USA

Abstract. Since starting in 2010, the Large Hadron Collider (LHC) has produced collisions at an ever increasing rate. The ATLAS experiment successfully recorded the collision data with high efficiency and excellent data quality. Events were selected using a three-level trigger system, where each level made a more refined selection. The Level 1 (L1) trigger consisted of a custom-designed hardware trigger which seeded two higher software based trigger levels. Over 300 triggers composed a trigger menu which selected physics signatures such as electrons, muons, particle jets, etc. Each trigger consumed computing resources of the ATLAS Trigger system and offline storage. The LHC instantaneous luminosity conditions, desired physics goals of the collaboration, and the limits of the trigger infrastructure determined the composition of the ATLAS Trigger menu. We describe a trigger monitoring framework called the Cost Monitoring Framework for computing the costs of individual trigger algorithms such as data request rates and CPU consumption. This framework was used to prepare the ATLAS Trigger for data taking during increases of more than six orders of magnitude in the LHC luminosity and has been influential in guiding ATLAS Trigger computing upgrades.

1. Introduction

In March of 2010, the Large Hadron Collider (LHC) started colliding proton-proton beams at 7 TeV center-of-mass energy. Initially, each beam contained a single proton bunch with low proton density generating fewer than one collision per beam crossing. Since then the LHC has increased the peak luminosity by more than six orders of magnitude. By October of 2011, the peak luminosity reached $\mathcal{L} = 3.65 \times 10^{33} cm^{-2}s^{-1}$, which corresponds to approximately 255 million inelastic collisions per second. At this time, the ATLAS [1] event recording rate has averaged 300-400 Hz [2].

Increasing the LHC luminosity presented unique challenges for the ATLAS trigger system. Events were selected using a three-level trigger system, where each level made a more refined selection. The Level 1 (L1) trigger consisted of a custom-designed hardware trigger which seeded the Higher Level Trigger (HLT) consisting of the two software based trigger levels. Before the first collisions, extensive detector and software tests were carried out with Monte Carlo (MC) simulation, test beams and cosmic muon data. Initially, only very loose selections using L1 triggers were applied online. Data collected in either special runs or parasitically from normal runs, which provided a sufficiently balanced data sample to work with high and low rate triggers, was used to validate more selective HLT algorithms. For this ”Enhanced Bias” (EB) data more detailed timing information was collected on an event by event basis including events that
were rejected. The collected data was used to validate more selective HLT algorithms. These algorithms were later deployed online before a next step in LHC luminosity. During this rapid luminosity ramp up, it was critical that a deployed trigger menu did not exceed limits imposed by the Data Acquisition system (DAQ) and the online computing resources. To maintain constant rates, trigger selections were optimized before each significant step in the peak luminosity.

An execution of a trigger signature incurs computing and data storage costs. This paper describes a detailed trigger monitoring framework, which will be referred to as the Cost Monitoring Framework for the rest of this paper, developed by the ATLAS collaboration to quantify computing costs and network bandwidth usage associated with trigger signatures and entire trigger menus. These costs can be extrapolated to higher luminosity and increased number of collisions per bunch crossing called pileup using specially collected EB data. In 2012 the luminosity at the LHC has already exceeded the 2011 records with plans for further increase of the current luminosity of \( L = 5.54 \times 10^{33} \text{cm}^{-2} \text{s}^{-1} \) in the near future. The Cost Monitoring Framework [3] has already played a key role in preparing for the new data.

2. ATLAS Trigger

The ATLAS Trigger searched for interesting physics signatures and reduced the data recording rate down to a manageable size that would allow for multiple years of data taking. The ATLAS Trigger was built as a three level system [5] consisting of a custom-designed hardware decision at the Level 1 (L1) and two software based trigger levels called the Level 2 (L2) and the Event Filter (EF) as seen in figure 1. Each level had progressively more time to make a decision allowing for more refined selections. The decisions were made by feature extraction and hypothesis testing.

The L1 used a set of hardware triggers that relied on data from the muon, calorimeter, and minimum bias detectors running at the collision frequency of 40MHz [2]. Small localized Regions of Interest (RoI) were identified as possibly containing a physics signature such as: muon, electron, photon, tau or jet. Each physics signature had a set of criteria to pass, and the accepted events were stored on ReadOut-Subsystems (ROS) which contained many ReadOut Buffers (ROB) that buffered the events for the remaining two trigger levels as seen in figure 1.

The L2 was seeded by the L1 RoI, and around 2-4% of the detector data was collected as needed from the ROS PCs by specialized algorithms. The algorithms were executed sequentially on commodity PCs to quickly reconstruct interesting features in \( \approx 37 \text{ms} \) online during 2011 as you can see in figure 3. At the end of each sequential step, thresholds were applied to accept or reject the event, and if passed, the successive steps built upon the results of the previous step. This allowed for quicker decisions because full reconstruction was not required to reject an event. An example of the stepwise reconstruction of a muon trigger can be seen in figure 2.

If accepted, the full detector data on the ROS PCs was moved to the Event Builder. The EF ran a series of complicated reconstruction algorithms with the full detector data from the Event Builder to make the final decision. The EF made decisions faster than design with results in around 0.7s during 2011 data taking as seen in figure 3, and accepted events were written to permanent storage.

A sophisticated software framework (steering framework) configured and executed the High Level Trigger (HLT) [7], which consisted of the L2 and EF. The steering framework recorded cost monitoring data using histograms, which were accessible in real time in the control room and archived for later analysis. Also the steering framework buffered more detailed information on the HLT algorithm execution for each event, which was collected using dedicated triggers to attach the buffers for readout. Detailed buffers were attached to about 1 out of 10 dedicated triggers at the L2 and 1 out of 100 at the EF. All other data for these events was stripped to reduce bandwidth use and storage size, which was a few hundred bytes per event. The cost monitoring data was extracted by the Cost Monitoring Framework using standard ATLAS data.
**Figure 1.** Trigger and DAQ design architecture

**Figure 2.** Example of stepwise execution for a muon trigger. Each step built off the results of the prior step allowing for more refined selection.
processing systems. The trigger cost data consisted of L1, L2 and EF trigger decisions, unique event id, algorithm run time, and data requests size and location. The L2 and EF processing times were monitored for many different collision conditions. As an example figure 3 shows L2/EF event processing times and L2 data retrieval times.

![Figure 3.](left) Level 2 (L2) processing and data retrieval time (right) Event Filter (EF) processing time [4]

3. Predict Trigger Rates
Cost monitoring data was used from special runs called Enhanced Bias (EB) runs to predict trigger rates. The EB cost monitoring data was collected to get a large statistics sample of high and low rate triggers. It was important to predict low rate triggers accurately because a lot of them were supporting triggers that were used to understand the higher rate triggers. In order to predict trigger rates accurately, the EB cost monitoring data must be treated carefully. Two random triggers were used to calibrate the high rate and low rate triggers. Overlap between the selection of these two trigger seedings was calculated using:

\[ \text{Prob}(\text{seeding}) = p_1 + p_2 - p_1 \times p_2, \]  

where \( p_1 \) was the probability for the first random trigger and \( p_2 \) was the probability for the second random trigger.

Many triggers are run with prescales, which select events with a probability of 1/prescale. To predict the rate of a trigger for higher luminosity, the following formula was used:

\[ \text{Rate}(\text{trigger}) = \frac{\mathcal{L}_\text{predicted}}{\mathcal{L}_\text{EB}} \times \frac{PS_{\text{EB}}}{PS_{\text{new}}} \times \text{Prob}(\text{seeding}), \]  

where \( \mathcal{L}_\text{predicted} \) was the new collision luminosity, \( \mathcal{L}_\text{EB} \) was the enhanced bias luminosity, \( PS_{\text{EB}} \) was the prescale of the online EB data, \( PS_{\text{new}} \) was the new prescale defined by the user, and \( \text{Prob}(\text{seeding}) \) is the above defined overlap probability of the overlap.

Menu experts used the Cost Monitoring Framework and the EB cost monitoring data to iterate quickly (10min) over a trigger menu and adjusted the prescales for the 300-400 triggers in a menu with the help of the predicted trigger and stream rates from the Cost Monitoring Framework. The procedure allowed for predictions of newly proposed triggers that have never been run online, and the resulting predictions were very accurate as can be seen in figure 4. An extension of the current framework will soon be available that allows dynamic predictions for new multi-object triggers, which will facilitate the rapid exploration of possible configurations.
Figure 4. (left) Online trigger rates and predictions for a few primary EF triggers (right) Online trigger rates and predictions for different physics streams and trigger levels [4].

4. Monitor and Predict ROS Rates
Detailed data request information was stored in the cost monitoring data, which could monitor requests to the ROS PCs. ROS request rates were monitored online, and the Cost Monitoring Framework automatically produced detailed summaries of these requests including: the requesting algorithm name, the number of requests, and the data size requested. The summaries were regularly consulted by trigger menu experts. The L2 menu was optimized in such a way that the caching and reuse of retrieved Read Out Buffers was maximized. This allowed a menu expert to design trigger menus that will reduce the load on very specific regions of the detector. Also during stops in data taking, the ROS PCs with the highest rates were replaced with faster machines whenever possible to avoid dead time.

In 2011, an optimization of the readout buffer assignment was performed. The optimization spread the amount of requests more evenly among the ROS PCs. The cost monitoring data played a major role in identifying ROS PCs with a high access rates and in testing new configurations with a procedure similar to the trigger rate predictions in section 3. If ROS request limits were exceeded in just one ROS, then data for sections of the detector were lost for that event. The predictions allowed ROS experts to iterate through many configurations without collecting new data. The result was a much more uniform request pattern as can be seen in figure 5. In addition to predicting one set of beam conditions, the EB cost monitoring data was also used to predict ROS access rates for changing collision conditions. The projections ensured that the ROS requests were run at the highest rate possible without exceeding bandwidth or retrieval rate limits, which maximized the amount of physics data that was collected.

5. Monitor CPU Usage
The cost monitoring data recorded algorithm processing time as well as event information like the mean number of collisions. The CPU usage for RoI based triggers increased linearly over a large range of pileup. Examples of resource usage for electron and tau signature trigger algorithms can be seen in figure 6. With changing beam conditions, the input from the cost monitoring data such as RoI based trigger timing was used to make predictions of CPU requirements for higher pileup. The predictions gave feedback to trigger algorithm developers about their algorithm’s response to changing collision conditions. In addition, HLT CPUs were purchased as needed, according to the resource projections obtained from direct CPU monitoring and the Cost Monitoring Framework.
Figure 5. (left) Online ROS request rates before equal load distribution and predictions from Cost Monitoring after the load was distributed evenly. (right) Online request rates and predictions for ROS rates with five times the collision rate and more uniform rates per subdetector [4]. Note the spike on ends of the LAR are reduced after recabling the ROS PCs.

Figure 6. (left) Example of online tau trigger algorithm CPU usage versus the mean number of collisions (right) Example of online electron algorithm CPU usage versus the mean number of collisions [4]

6. Conclusion
In 2010 and 2011, the LHC rapidly increased its luminosity by orders of magnitude, and the ATLAS Trigger could seamlessly adapt to these quite dramatic changes with the help of the Cost Monitoring Framework [6]. The Cost Monitoring system stored detailed information for a few events and used that information to extrapolate toward new collision conditions. Predicting the upcoming trigger rates and resource allocation for new beam conditions made the Cost Monitoring system a vital tool for understanding the resource usage of the ATLAS Trigger.

7. References