Popularity Prediction Tool for ATLAS Distributed Data Management

T Beermann[1,2], P Maettig[1], G Stewart[2, 3], M Lassnig[2], V Garonne[2], M Barisits[2], R Vigne[2], C Serfon[2], L Goossens[2], A Nairz[2] and A Molfetas[2,4] on behalf of the ATLAS collaboration

[1] University of Wuppertal, Wuppertal, Germany
[2] CERN, CH-1211, Geneva 23, Switzerland
[3] University of Glasgow, Glasgow G12 8QQ, UK
[4] University of Melbourne, Melbourne, Australia

E-mail: `thomas.beermann@cern.ch`

Abstract. This paper describes a popularity prediction tool for data-intensive data management systems, such as ATLAS distributed data management (DDM). It is fed by the DDM popularity system, which produces historical reports about ATLAS data usage, providing information about files, datasets, users and sites where data was accessed. The tool described in this contribution uses this historical information to make a prediction about the future popularity of data. It finds trends in the usage of data using a set of neural networks and a set of input parameters and predicts the number of accesses in the near term future. This information can then be used in a second step to improve the distribution of replicas at sites, taking into account the cost of creating new replicas (bandwidth and load on the storage system) compared to gain of having new ones (faster access of data for analysis). To evaluate the benefit of the redistribution a grid simulator is introduced that is able replay real workload on different data distributions. This article describes the popularity prediction method and the simulator that is used to evaluate the redistribution.

1. Introduction

The ATLAS[1] collaboration creates and manages vast amounts of data. Since the detector started data taking, Don Quijote 2 (DQ2)[2], the collaboration's distributed data management system, is responsible for managing petabytes of experiment data on over 750 storage end points in the Worldwide LHC Computing grid[3]. DQ2 organizes, transfers and manages not only the detector's RAW data, but also the entire life cycle of derived data products for the collaboration's physicists. This is done in accordance with the policies established in the ATLAS Computing Model. Users send their jobs to the workload management system, PanDA, which then schedules the job to run on a grid site.

In this article we describe a new, dynamic way of pro-actively replicating data, which is based on predictions about the future access of datasets by analysing the popularity of datasets in the past. This is especially interesting for datasets used in user analysis jobs. Unlike production jobs, where the requirements are predefined, it is harder to foresee future popularity, which makes efficent data placement more difficult. Furthermore a simple grid simulator is introduced that helps evaluating the benefit of this new system.

2. ATLAS Data Distribution

The distributed data management system (DDM) operates on files, which usually contain many events. Events taken under the same detector conditions can be distributed over multiple files and some kind of aggregation is needed. For this, DQ2 has the concept of datasets. Datasets are a logical unit and combine one or more files. They are the operational unit for DQ2. Only datasets can be transferred between sites, whereas files cannot. The actual copies of a dataset on a site are called replicas. To create a new replica all files in a datasets are copied to a site in the grid and registered as a new replica in the system. So a logical dataset in DQ2 can have multiple physical replicas.

The distribution of datasets is based on policies defined by the Computing Resource Management (CREM)[5], guided by the ATLAS computing model and operational constraints. For each type of dataset a minimum number of replicas is defined. Furthermore, where these replicas need to be is defined (Tier-1 or Tier-2, disk or tape). Additionally the PanDA Dynamic Data Placement (PD2P)[6] replicates datasets reactively when a certain threshold of jobs using this dataset is exceeded.

To run analysis jobs on this data send their jobs to the ATLAS workload management system called PanDA. PanDA takes the user jobs and schedules them to run on a sites that host a replica of the dataset needed for analysis, based on availability and the workload of the site and the jobs' priority. Each time PanDA accesses a file in dataset it sends a report to the DQ2 tracer system. This report includes information about the corresponding dataset, the site it was accessed on, the user, the starting and ending time and the download status.

The challenge is to use all these systems to effectively make best use of all available resources. The problem is complicated, because of the difficulty of predicting future accesses by users and the number of degrees of freedom when replicating data. This article introduces a system that can help with this problem.

3. Dataset Access Prediction

To distribute data better first knowledge of a future access pattern is needed. For this a prediction model is need, which is introduced here. The undelying assumption is that historic user behaviour can be used to make predictions about future dataset popularity.

3.1. Concept

The goal of this part of the system is to estimate the number of dataset accesses for a certain time in the near-term future by finding trends in the historic dataset accesses. To find these patterns an artificial neural network (ANN)[7] is used in this case. The idea is to aggregate the accesses for each dataset per week and feed this to the neural network to get an estimate for the next week. The first n weeks of accesses to each dataset ($A_{1,2,..,n}$) are taken and a prediction is made for week $n+1$. To do this a neural network is trained which has n input neurones and 1 output neurone that outputs A_{n+1}th. Selecting a week as the averaging period was done to smooth out weekday/weekend variations and matches the necessary time to redistribute data on the grid.

3.2. Design

The system has one neural network per dataset project and datatype as well as the week to predict. The reason different neural networks are used for each project and datatype is that access patterns vary significantly for these different core data types and lumping them all together makes it hard for neural network to make a good prediction. Also one further step is introduced before the data is actually fed to the neural network. There are a lot of datasets stored on the grid that are not accessed at all or just very little. These datasets could be filtered when training the ANN and only datasets that are actually accessed would be taken for training.

Prediction method	RMS
Neural Network	24.48
Static	82.47
Linear	207.7

Table 1. RMS value for different prediction methods

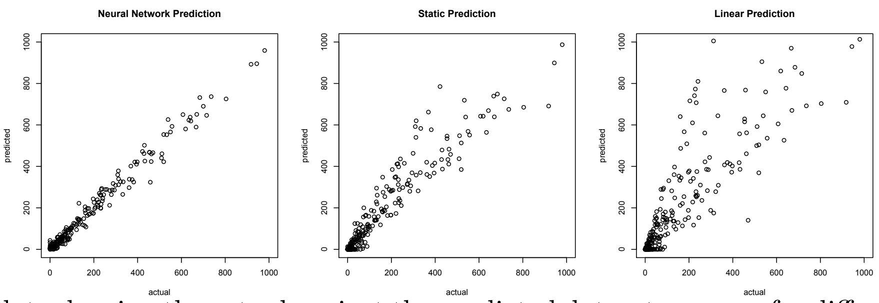

Figure 1. Plots showing the actual against the predicted dataset accesses for different prediction methods

This can be done by looking at the past accesses. If the accesses in the last weeks were below a certain threshold the chances are very little that it will be accessed in the future.

3.3. Evaluation

To evaluate the quality of the prediction made by the neural network we compared the results to two different simple predictions methods:

- **Static Prediction:** The static prediction method uses the nth value of the input vector as the prediction for the $n + 1$th week, i.e., $A_{n+1} = A_n$.
- **Linear Prediction:** The linear prediction predicts the $n + 1$th week as the nth week plus the difference between A_n and A_{n-1}, i.e. $A_{n+1} = 2A_n - A_{n-1}$.

Figure 1 shows an example of how the different prediction methods behave and table 1 shows the RMS of the difference between predicted and actual values for each method. To generate the training and evaluation sets a common input set was split in half. The results show the prediction of the evaluation set. The linear prediction is clearly the worst. The static prediction can give better than the linear prediction and can give in some cases also give better results than the neural network prediction. However, overall, the neural network prediction gives the best results of all methods.

Another result of the exercise is that neural networks are good at picking up the trend of dataset accesses in the future, i.e. predicting if the popularity of a certain dataset goes up, down or stays the same from one week to the next. In the example from figure 1 the neural network predicts the right trend in 77% of the cases.

4. Data Redistribution and Simulation

Predictions can be used to guide data redistribution on the grid, but some points have to be considered when doing this: 1. how many replicas are already available for this dataset; 2. how many times will the dataset be accessed; 3. how expensive would it be to create a new replica (bandwidth). Furthermore, there are boundaries that have to be considered: 1. there is only a limited amount of disk space, i.e. when creating new replicas this limit always has to be considered and unused replicas may have to be removed first; 2. when removing a minimum number of replicas should be kept per dataset depending of the type of the dataset.

A way is need to be able to evaluate if the additional replicas have a positive impact on analysis jobs run on the grid. For that reason in this section a simplified grid simulator is introduced that will be able to replay the same workload on different data distributions over the grid and measure the waiting time per job and the make span of the whole workload.

4.1. Design

The simulator consists of three main components:

- **Sites:** Manages each site's available and used disk space, as well as the number of job slots and the waiting queue.
- **Distributed Data Management:** The DDM system provides a mechanism to add and remove replicas to a central catalogue and to the sites, as well as providing information about replicas for datasets.
- **Workload Management System:** The WMS takes workload and schedules it to the sites, interfacing both with DDM, for data locality, as well as the sites for job slot information. The workload itself consists of a tuple with the job submission time, the execution time, the used datasets, the user and the number of files to be processed.

For each workload element the available replicas for the corresponding dataset are requested from DDM. In the next step the currently available job slots are requested for each hosting site. Then the site with the biggest available capacity is chosen to run the job. If no hosting site has a job slot available then the waiting queue sizes are considered and the site with the smallest waiting queue is chosen. If a job slot is available for this job it is run for the given execution time. If the job is put in the waiting queue it will wait until it becomes the first in the queue and then take the a job slot when one becomes available. The simulator then will run until no workload is left anymore.

4.2. Metrics

It is important to have clear metrics that assess whether a certain distribution is better than another one. Two important metrics are introduced and measured by the simulator:

- **Make span:** The make span measures how long it takes until the simulator has finished running all the workload.
- **Waiting time per job:** The average time it takes from the scheduling of a job until it actual starts running.

The make span already gives a good indicator about the benefit of a distribution, but it is strongly biased by the last jobs that are submitted and their duration. The waiting time per job is a very important metric as directly shows the positive or negative impact of a certain distribution on the time it takes for users' jobs to complete. Furthermore more metrics like the maximum waiting queue size and the maximum number of used job slots are recorded. But overall the average waiting time is the best metric to measure a possible gain with the distribution as it directly affects the user.

4.3. Evaluation

The simulator now gives the possibility to evaluate different data distribution strategies. The strategies that are tested here are:

- **Actual Data Distribution:** The data distribution as extracted from DQ2.
- **One extra replica:** Based on the actual data distribution one extra replica is added for all datasets that will be accessed in the future, without taking the available disk space into account.

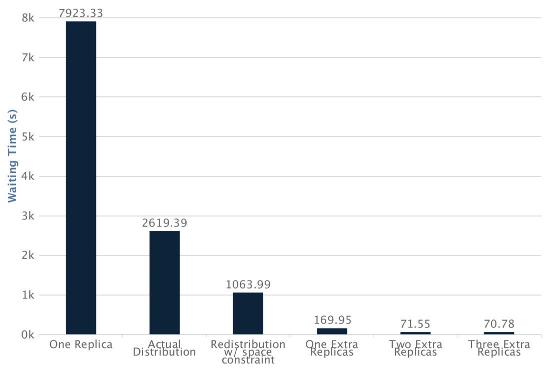

Figure 2. Plot showing the average job waiting time for different distribution methods for a workload of one day.

- **Two extra replicas:** As above, but with two extra replicas.
- **One extra replica with space constraints:** First remove all additional replicas for dataset that are not accessed and then fill up this space with replicas for accessed datasets.
- **All extra replicas removed:** Base on the actual data distribution all extra replicas are removed, i.e. only the minimum number of replicas is kept per dataset.

Simulating only the DE cloud and using data and Monte Carlo datasets a first simulation run was performed. For this the workload and the data distribution was extracted from the PanDA WMS and DDM systems and replayed with this simulator. The data was distributed as described before and the same workload was run for all of them. For each run the average waiting time per jobs was measured. Preliminary results for a simulation run with one day of workload can be seen in figure 2. The results show that the worst possibility is with no additional replicas, where the waiting times go up a lot. On the other hand it can be seen that at some point additional replicas only bring a only a small gain. The actual distribution gives already a good result compared to the worst case, but it can be better as can be seen in the results for an additional replica.

5. Conclusion and Future Work

In the currently running system data distribution is either a manual process or proceeds by creating new replicas reactively. This article shows how future access patterns can be predicted. With this knowledge a better data distribution can be achieved. To evaluate the benefits a simulator was described and first results where presented. Based on this work different redistribution strategies will be developed and evaluated.

References

[1] Jones R and Barberis D 2008 The atlas computing model *Journal of Physics: Conference Series* vol 119 (IOP Publishing) p 072020
[2] Branco M, Zaluska E, De Roure D, Lassnig M and Garonne V 2010 *Concurrency and Computation: Practice and Experience* 22 1338-1364 ISSN 1532-0634
[3] Bird I, Bos K, Brook N, Duellmann D, Eck C, Fisk I, Foster D, Gibbard B, Girone M, Grandi C *et al.* 2008 *EGEE, Mar* 18
[4] Maeno T, De K, Wenaus T, Nilsson P, Walker R, Stradling A, Fine V, Potekhin M, Panitkin S, Compostella G 2012 *Journal of Physics: Conferences Series* vol 396 (IOP Publishing) p 032071
[5] Computing Resource Management accessed on 23.09.2013 URL `https://twiki.cern.ch/twiki/bin/viewauth/Atlas/ComputingResources`
[6] Maeno T, De K, Panitkhin S 2012 *Journal of Physics: Conferences Series* vol 396 (IOP Publishing) p 032070
[7] Artificial Neural Networks accessed on 23.09.2013 URL `http://en.wikipedia.org/wiki/Artificial_neural_network`