IceProd 2 Usage Experience

D Delventhal¹, D Schultz¹, J C Diaz Velez¹

¹ Wisconsin IceCube Particle Astrophysics Center, University of Wisconsin-Madison, 222 W Washington Ave, Suite 500, Madison, WI 53703, USA

E-mail: david.delventhal@icecube.wisc.edu, david.schultz@icecube.wisc.edu, juancarlos@icecube.wisc.edu

Abstract. IceProd is a data processing and management framework developed by the IceCube Neutrino Observatory for processing of Monte Carlo simulations, detector data, and data driven analysis. It runs as a separate layer on top of grid and batch systems. This is accomplished by a set of daemons which process job workflow, maintaining configuration and status information on the job before, during, and after processing. IceProd can also manage complex workflow DAGs across distributed computing grids in order to optimize usage of resources.

IceProd has recently been rewritten to increase its scaling capabilities, handle user analysis workflows together with simulation production, and facilitate the integration with 3rd party scheduling tools. IceProd 2, the second generation of IceProd, has been running in production for several months now. We share our experience setting up the system and things we’ve learned along the way.

1. Introduction

The IceCube detector is located at the geographic South Pole and was completed at the end of 2010 [1]. It consists of 5160 optical sensors buried between 1450 and 2450 meters below the surface of the South Pole ice sheet and is designed to detect interactions of neutrinos of astrophysical origin. The IceProd [2] software framework was developed by IceCube in 2006 to manage distributed simulation workloads. It has been used in production for 11 years to handle the data processing and simulation needs for the IceCube collaboration. It has run thousands of CPU years and produced over three petabytes of data. In order to keep up with the increasing demands in computing resources, a second version has been in development over the last few years. This article describes the experience developing, testing, and deploying IceProd 2 during the last year.

2. IceProd 2

The second version of IceProd is a complete rewrite of the codebase, focusing on current and future usage and eliminating old usage patterns [3]. It promotes better software design practices, including unit tests and code coverage. IceProd 2 includes many new features and improvements in both the server and the job wrapper. A pilot system has been put in place to reduce job failures and run multiple jobs on a single worker. Database load is now spread out by having a separate database instance at each site. The web server that handles all external communications is now significantly more scalable, easily handling 10 thousand connections per minute. Support for user accounts, groups, and priority-based scheduling is being developed. IceProd 2 is also web focused — its actions can all be controlled via a full JSON API and includes a web-based interface for performing common actions and queries.
3. Recent work

Over the past year many improvements have been made to the reliability and performance of IceProd.

2. While several improvements were long-term goals, most were to aid in debugging and monitoring as testing increased in scale and complexity.

3.1. Removal of internal RPC

Previously different modules communicated with each other using an RPC (Remote Procedure Call) protocol. This allowed for asynchronous requests for potentially high latency operations, such as database queries. The RPC code has now been completely removed and replaced using Python futures. As a result, the number of lines of code has been decreased by about 15 percent. Such a substantial decrease in code base size has the benefit of increasing reliability and maintenance. The removal of RPC has also made tracing exceptions much easier.

3.2. Job pilot

Improvements have also been made to the pilot. A common issue is that jobs are evicted for using more memory than they have requested. The pilot now allocates extra memory for the job to use in order to decrease the likelihood of eviction. The new pilot can also run multiple jobs on the same node. This can greatly increase the efficiency, packing more jobs into the pilot resource limits.

3.3. Monitoring

For IceProd 2, a new monitoring system is being deployed and is currently in testing. Information is collected and sent over to Graphite [4] for aggregation and display. This collected information includes JSON RPC messages, including failures; website requests; queue status of both pilots and jobs; and dataset progress, including number of jobs that fail or are reset. More measurements are added as needed.

3.4. Improved logging

All of the logs produced by running jobs are stored in the database. These logs are crucial for diagnosis of jobs running on the cluster, and can be conveniently viewed via the web interface. For each job, there is a list of log categories (e.g. stdout, stderr) followed by the last few lines of the log information. The entire log can be viewed by clicking on a link.

4. Status and future work

IceProd 2 started production-scale testing in June 2016. Many issues, including those previously mentioned, were fixed. Official simulation datasets started in December 2016, running in parallel with IceProd 1. We plan to switch all of simulation and experimental data production over to IceProd 2 by the end of spring 2017. At that time, we will also invite analysis users to migrate to IceProd 2 instead of hand-written submissions.
4.1. Distributed intermediary storage
One of the more important long-term goals is the distributed storage of intermediary files. Currently all intermediary files are saved on one server at the main datacenter in Madison. At some point, using a single server will not scale very easily, even with 10Gb/s connectivity. It is also possible to saturate a network link closer to the compute nodes. We hope within the next year to be able to use multiple servers to store temporary files. Initially this will be random selection, but we’d like over time to be able to regionalize the network of sites and optimize available network bandwidth.

4.2. Pilot file staging
A significant fraction of time for each job is spent staging files in and out. This leads to an inefficiency, since most resources (except disk) are not being used during transfers. This is especially inefficient for GPUs, as these jobs are generally short yet read/write a large amount of data. The solution is to make the pilot do the input/output file staging in the background, while another job runs.

4.3. MongoDB
Because of the large amount of job statistics and logs being accumulated, we are investigating the use of MongoDB or other storage engines. On startup, IceProd can check for the presence of MongoDB, and if not found, a regular SQL database will be used as a fallback. This is still in the planning stages, as a long-term goal.

4.4. Web interface design
IceProd contains a web-based interface that allows users to submit jobs, change the configuration, and query dataset statistics. Currently, this interface is fully functional, but its design needs significant improvement. As tablets and phones are very popular internet devices, the interface must work well on smaller screens and touch-enabled devices. Areas of focus for improvement include layout, sizing, styling, and general usability.

5. Conclusions
IceProd is an easy to install and manage user-level middleware designed to monitor large sets of distributed jobs. The IceCube collaboration uses it for simulation, detector data, and analysis level processing. Significant progress has been made in developing IceProd 2, a complete rewrite compared to IceProd 1. Many new features have been added and performance improvements have been made. IceProd 2 is currently being actively used for production.

Acknowledgements
We acknowledge the support from the following agencies: U.S. National Science Foundation-Office of Polar Programs, U.S. National Science Foundation-Physics Division, University of Wisconsin Alumni Research Foundation, the Center for High Throughput Computing (CHTC) at the University of Wisconsin–Madison, the Open Science Grid (OSG) grid infrastructure; U.S. Department of Energy, and National Energy Research Scientific Computing Center; Natural Sciences and Engineering Research Council of Canada, WestGrid and Compute/Calcul Canada; Swedish Research Council, Swedish Polar Research Secretariat, Swedish National Infrastructure for Computing(SNIC), and Knut and Alice Wallenberg Foundation, Sweden; German Ministry for Education and Research (BMBF), Deutsche Forschungsgemeinschaft (DFG), Helmholtz Alliance for Astroparticle Physics (HAP), Research Department of Plasmas with Complex Interactions (Bochum), Germany; Fund for Scientific Research (FNRS-FWO), FWO Odysseus programme, Flanders Institute to encourage scientific and technological research in industry (IWT), Belgian Federal Science Policy Office (Belspo); University of Oxford, United Kingdom; Marsden Fund, New Zealand; Australian Research Council; Japan Society for Promotion of Science (JSPS); the Swiss National Science Foundation (SNSF), Switzerland; National Research Foundation of Korea (NRF); Danish National Research Foundation, Denmark (DNRF).
References

[1] Halzen F 2003 IceCube A Kilometer-Scale Neutrino Observatory at the South Pole
    IAU XXV General Assembly, Sydney, Australia, 13-26 July 2003, ASP Conf. Series vol 13
    volume 13, pages 13-16, July 2003

[2] Aartsen M G et al 2015 The IceProd framework: Distributed data processing for the
    IceCube neutrino observatory J. Parallel Distrib. Comput. 75 198–211

[3] Schultz D 2015 IceProd 2: A next generation data analysis framework for the
    IceCube neutrino observatory J. Phys.: Conf. Ser. 664 062056

[4] Graphite project, https://graphite.readthedocs.io