PAPER • OPEN ACCESS

Leveraging the checkpoint-restart technique for optimizing CPU efficiency of ATLAS production applications on opportunistic platforms

To cite this article: D Cameron *et al* 2018 *J. Phys.: Conf. Ser.* **1085** 032028

Leveraging the checkpoint-restart technique for optimizing CPU efficiency of ATLAS production applications on opportunistic platforms

D Cameron[1], J Elmsheuser[2], L Heinrich[3], W Lavrijsen[4], P Nilsson[2], V Tsulaia[4] and M Vogel[5] on behalf of the ATLAS Collaboration
[1]University of Oslo, P.b. 1048 Blindern, 0316 Oslo, Norway
[2]Brookhaven National Laboratory, PO Box 5000, Upton, NY 11973, USA
[3]New York University, 70 Washington Square South, New York, NY 10012, USA
[4]Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720, USA
[5]University of Wuppertal, Gausstrasse 20, 42119 Wuppertal, Germany

E-mail: VTsulaia@lbl.gov

Abstract. Data processing applications of the ATLAS experiment, such as event simulation and reconstruction, spend considerable amount of time in the initialization phase. This phase includes loading a large number of shared libraries, reading detector geometry and condition data from external databases, building a transient representation of the detector geometry and initializing various algorithms and services. In some cases the initialization step can take as long as 10-15 minutes. Such slow initialization has a significant negative impact on overall CPU efficiency of the production job, especially when the job is executed on opportunistic, often short-lived, resources such as commercial clouds or volunteer computing. In order to improve this situation, we can take advantage of the fact that ATLAS runs large numbers of production jobs with similar configuration parameters (e.g. jobs within the same production task). This allows us to checkpoint one job at the end of its configuration step and then use the generated checkpoint image for rapid startup of thousands of production jobs. By applying this technique we can bring the initialization time of a job from tens of minutes down to just a few seconds. In addition to that we can leverage container technology for restarting checkpointed applications on the variety of computing platforms, in particular of platforms different from the one on which the checkpoint image was created.

We will describe the mechanism of creating checkpoint images of Geant4 simulation jobs with AthenaMP (the multi-process version of the ATLAS data simulation, reconstruction and analysis framework Athena) and the usage of these images for running ATLAS Simulation production jobs on volunteer computing resources (ATLAS@Home) and on Supercomputers.

1. Introduction

The ATLAS Experiment [1] processes its data at over 140 computing centers around the world using more than 4 million CPU-hours/day. Even at such a massive scale, the data processing of the experiment is resource-limited. The ATLAS physics program will strongly benefit from applying more compute resources to Monte Carlo simulation, and over the next decade the situation will become even more critical with the LHC [2] and ATLAS upgrade programs bringing an order of magnitude increase in computing requirements. In view of the steady demand for new computing resources, it becomes very important for the experiment not only to efficiently use all CPU power available to it, but also to pro-actively leverage computing resources it does not own, such as commercial clouds, supercomputers and volunteer computing.

In order to effectively use all available CPU-cores on a compute node, ATLAS is running the majority of its production workflows using AthenaMP [3] - a process-parallel version of the ATLAS data processing framework Athena. An AthenaMP job starts as a serial process (the master), which first goes through the application initialization phase, initializes and allocates as much memory as possible and then forks several event processors (workers). The workers run in parallel and process all events assigned to the given job. They share a substantial fraction of process memory and this memory sharing is provided solely by the Linux kernel copy-on-write (COW) mechanism. The serial part of an AthenaMP job (initialization in the master process) includes loading a large number of shared libraries, reading detector geometry and conditions data from external databases, building a transient representation of the detector geometry and initializing various components of the framework. In some cases, e.g. a slow connection to an external database, such an initialization step can take as long as 10-15 minutes. This may have a visible negative effect on CPU efficiency of the entire AthenaMP job, especially when the job is executed on an opportunistic resource with a maximum allowed wall time of 1-2 hours.

For speeding up the initialization step of AthenaMP jobs, we can benefit from the fact that in ATLAS production workflows a very large number of jobs can have almost identical job configuration. For example, all jobs within the same simulation production task have the same geometry and physics process configuration and only very few configuration parameters (e.g. input file name) change from job to job. This allows us to run one sample job from such a task, checkpoint it just after the initialization step and then use the generated checkpoint image for fast startup of all jobs within the same task, with only minimal modifications of the configuration parameters after the restart.

There are a number of use-cases where HEP software in general and ATLAS software in particular can benefit from leveraging the process checkpoint-restart technique [4]. In our studies so far we investigated how we can apply this technique to avoid the time-consuming initialization step of ATLAS Geant4 Simulation [5, 6] production jobs. In this paper we describe preliminary results obtained on two platforms: Volunteer Computing (ATLAS@Home [7]) and the Intel Knights Landing (KNL) supercomputer Cori at the National Energy Research Scientific Computing Center (NERSC), Berkeley, USA.

2. DMTCP - a tool for process checkpoint-restart

DMTCP (Distributed MultiThreaded Checkpointing) [8] is a free, open source tool which implements a process checkpoint-restart on the library level. It supports single-threaded, multi-threaded and multi-process applications running either on a single host or in a distributed environment. One of the main advantages of DMTCP is that it implements transparent checkpointing in user space with no modifications required either to user code or to the OS.

DMTCP supports two strategies for creating checkpoint images: a checkpoint can be triggered by an external process (dmtcp coordinator) or by the application itself. In the latter case the application decides when it is a good time to checkpoint itself and does so by calling the DMTCP API functions.

By default, when a checkpoint image is generated, DMTCP uses gzip for compressing the image file. If necessary, users can disable this feature via the DMTCP command-line interface. The compressed images use considerably less space on disk (a factor of 3 reduction was observed for ATLAS simulation images), but take a somewhat longer time to restart.

For our studies of AthenaMP checkpoint-restart, we used DMTCP version 2.4.5, which has been integrated into the ATLAS offline software release as an external package.

3. Checkpoint-Restart of AthenaMP

Athena/AthenaMP Monte Carlo production jobs are managed by the ATLAS workload management system Panda [9]. These jobs use specialized python wrappers, so called job transformations. One of the main purposes of a job transformation is to collect various configuration parameters via command-line options and to pass them over to the underlying Athena job. In order to support the process checkpoint-restart mechanism we introduced two new command-line options to the job transformation code: `--checkpoint` and `--restart`.

When the `--checkpoint` option is used, AthenaMP checkpoints itself at the end of the initialization step by calling the corresponding DMTCP API functions and then exits immediately. This operation generates a checkpoint image file as well as some auxiliary scripts, which are needed by DMTCP for restarting from this image later on. All these files are stored into an archive file for later usage.

In order to restart AthenaMP from a checkpoint image, the location of the checkpoint archive file is passed to the job transformation as a value of the `--restart` command-line option. In this case, instead of starting AthenaMP the normal way, the transformation first unpacks the checkpoint archive into the job's run directory and then sources the auxiliary DMTCP script, which restarts AthenaMP from the checkpoint image. The first thing AthenaMP does after the restart is to update some of the configuration parameters, e.g. the input file name and the number of events to process. AthenaMP reads this information from a special file created by the job transformation, which contains the current job configuration. After that AthenaMP proceeds with processing events in the usual way.

4. Portability of checkpoint images

Today ATLAS is running its production jobs in a heterogeneous computing environment. Jobs from the same production task can be executed at different production sites with potentially different computing platforms. Hence, from the perspective of the utilization of process checkpoint-restart, it is desirable to generate one checkpoint image once per task and to be able to use this image for launching jobs on heterogeneous platforms.

This is a challenging task, since DMTCP expects the same or very similar environment on the restart host compared to the checkpoint host. In order to work around this limitation we can leverage either virtual machine (VM) or container technologies. In particular, by using the same VM/container image for checkpoints and restarts the above requirement can be met.

Another issue, which we encountered during our tests and which can affect the portability of the checkpoint images, is the access to external resources (e.g. databases). For example, if a checkpointed job was configured to read data from a database which is accessible only for machines within a local network, then an attempt to restart such application at a remote site will lead to database connection errors. In order to deal with these kind of problems, the restarted application must be able to properly modify its configuration parameters (e.g. database connection string) required for accessing external resources.

5. Testing with production workloads

5.1. Volunteer Computing (ATLAS@Home)

ATLAS@Home is a research project which uses volunteer computing to execute detector simulation jobs with Geant4. On one hand it helps the experiment in finding free CPU cycles in a time of limited funding increases, and on the other hand it is an outreach tool to get the public involved in ATLAS. Like the majority of volunteer computing projects, ATLAS@Home is based on the BOINC [10] platform, which has built-in support for virtualization using VirtualBox [11]. Virtualization is required mainly because the majority of volunteers run the Windows operating system and Athena requires a Linux environment. ATLAS@Home is fully integrated into the

ATLAS workflow management system, such that from outside it looks like a regular grid site, and over the past 3 years ATLAS has simulated almost 200 million events using ATLAS@Home.

ATLAS@Home makes a good choice for a prototype platform to test the process checkpoint-restart technique for speeding up AthenaMP initialization time. Since jobs run inside a virtual machine, the environment can be fully controlled, i.e. the same VM image can be used for making checkpoint images and for restarting from them. There are several reasons which make fast initialization on ATLAS@Home rather important. Jobs in ATLAS@Home usually run for 1-2 hours, which increases the fraction of the initialization step in the overall job wall time with respect to "standard" production jobs running on the Grid. In addition volunteers may have a poor Internet connection which can lead to a very long initialization phase while data is read from external servers.

For testing the checkpoint-restart mechanism, we ran an interactive job inside ATLAS@Home VM, created a tarball with compressed checkpoint image and saved the tarball in the VM image. Preliminary results showed that the performance gains coming from restarting from a checkpoint image on ATLAS@Home strongly depend on external database access speed. In cases of fast database access ATLAS Geant4 simulation job initializes in $\sim$4 minutes, while for nodes with slow database access this number can go up to $\sim$20 minutes. By restarting from a checkpoint image we eliminate all need for database access during initialization and the restart usually takes 15-20 seconds, which is a rather significant improvement even for jobs with a fast database connection.

5.2. Intel KNL supercomputer

Cori is a Cray XC40 supercomputer at NERSC, Berkeley, USA. It is a unique system among supercomputers of its size with two different kinds of nodes: 2.4k Intel Xeon "Haswell" processor nodes (Cori Phase 1) and 9.7k Intel Xeon Phi "Knight's Landing" nodes (Cori Phase 2). ATLAS has been running Geant4 simulation production workflows on Cori KNL nodes since July 2017. In August 2017 the experiment simulated 135 million events on NERSC supercomputers, most of them on Cori Phase 2.

ATLAS simulation production jobs on Cori do not need to access external databases for reading detector geometry and condition data. Instead they use frozen snapshots of these databases stored in SQLite files on the Cori shared file system. Nevertheless, individual CPU cores on Intel KNL processors have considerably less processing power than, for example, CPU cores on Intel Haswell processors (1.4 GHz vs 2.3 GHz) and, as a result, on KNL we observe a significant slowdown of the serial initialization step of AthenaMP jobs.

To study performance gains from the usage of the process checkpoint-restart technique on Intel KNL, we created two types of checkpoint images: compressed and uncompressed. The image archive files were unpacked into the job run directories prior to submitting test jobs to the Cori batch queue, such that the jobs did not have to use additional time on unpacking archives. Using this approach we tested compressed and uncompressed checkpoint images and compared the results to startup times of conventional AthenaMP jobs. For each configuration we ran 300 single-node jobs. The results of our studies are presented in Table 1. The job startup time was measured between launching AthenaMP and the moment it forked worker processes. The results show that even with a compressed checkpoint image we can achieve a 13-fold speedup of job startup time with respect to conventional AthenaMP, while for uncompressed images the performance gains are much larger.

6. Summary and Outlook

We have successfully tested the DMTCP tool for checkpoint-restart of Geant4 simulation jobs in the multi-process framework AthenaMP. First tests with production ATLAS jobs on

Table 1. AthenaMP Simulation Startup Times on Cori KNL Nodes.

	Image size	Startup time (sec)	Startup speedup
`Conventional AthenaMP`	N/A	663.1 ± 22.8	1
`Compressed image`	550MB	50 ± 9.7	13.3x
`Uncompressed image`	1.8GB	20.8 ± 9.1	31.5x

ATLAS@Home and Intel KNL platforms have demonstrated that by restarting from checkpoint images we can considerably speedup job startup times.

There are however several improvements to be implemented before deploying on a larger scale. We plan to run more tests in order to demonstrate that by restarting from a checkpoint image we do not introduce a visible penalty into the event processing throughput. Also the process of creating and distributing checkpoint images must be automated and the results of the jobs restarted from checkpoint images must be validated for physics correctness.

7. Acknowledgments

The authors would like to thank our volunteer testers in ATLAS@Home for running the jobs for us and providing useful feedback.

The performance results on Cori KNL platform presented in this paper have been obtained by using resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] ATLAS Collaboration *JINST* **3** (2008) S08003
[2] Evans L and Bryant P “LHC Machine”, *JINST* **3** (2008) S08001
[3] Calafiura P et al. on behalf of the ATLAS Collaboration “Running ATLAS workloads within massively parallel distributed applications using Athena Multi-Process framework (AthenaMP)”, *J. Phys.: Conf. Ser.* **664** (2015) 072050 doi:10.1088/1742-6596/664/7/072050
[4] Arya K, Cooperman G, Dotti A and Elmer P “Use of checkpoint-restart for complex HEP software on traditional architectures and Intel MIC”, *J. Phys.: Conf. Ser.* **523** (2014) 012015 doi:10.1088/1742-6596/523/1/012015
[5] Agostinelli S et al. GEANT4 Collaboration “Geant4 a simulation toolkit”, *Nucl. Instrum. Meth. A* **506** (2003) 250 doi:10.1016/S0168-9002(03)01368-8
[6] ATLAS Collaboration (2010) “ATLAS Simulation Infrastructure”, *Eur. Phys. J* **C70** (2003) 823
[7] Adam-Bourdarios C et al. on behalf of the ATLAS Collaboration “ATLAS@Home: Harnessing Volunteer Computing for HEP”, *J. Phys.: Conf. Ser.* **664** (2015) 022009 doi:10.1088/1742-6596/664/2/022009
[8] DMTCP project, ”DMTCP” [software], version 2.4.5, 2016. Available from https://github.com/dmtcp/dmtcp/releases/tag/2.4.5 [accessed 2016-07-19]
[9] Maeno T for the ATLAS Collaboration “PanDA: Distributed production and distributed analysis sys tem for ATLAS”, *J. Phys.: Conf. Series* **119** (2008) 062036 doi:10.1088/1742-6596/119/6/062036
[10] Anderson D *Proc. 5th IEEE/ACM Int. Workshop on Grid Computing, GRID 04* (2004) pp 410
[11] VirtualBox project, ”VirtualBox” [software], version 5.1.22, 2017. Available from https://www.virtualbox.org