

Skygrid

Alexander Baranov¹, Konstantin Nikitin¹, Andrey Ustyuzhanin¹
¹ Yandex School of Data Analysis

Abstract. The Skygrid system was developed in order to provide distributed computing resources for SHiP experiment at CERN. Skygrid is designed to perform computations in grid manner with central scheduling service - Metascheduler, and multiple computing sites. Skygrid employs two new computational paradigms: running computations in containers and applying web browsers for volunteer computing. Each of these paradigms allows running computations on broader heterogeneous resources and making these computations reproducible in the same time.

1. Introduction

Skygrid is a system for distributed computing which was developed as a system for distributed computations for SHiP experiment. Main tasks of these system is Monte-Carlo simulation and event reprocessing. Skygrid performs computations in GRID manner: users submit their jobs to Metascheduler - central queue service, which then schedules jobs to executors. This scheme is depicted on Figure 1. Users, Metascheduler and executors talk via HTTP so no special software is needed on each party.

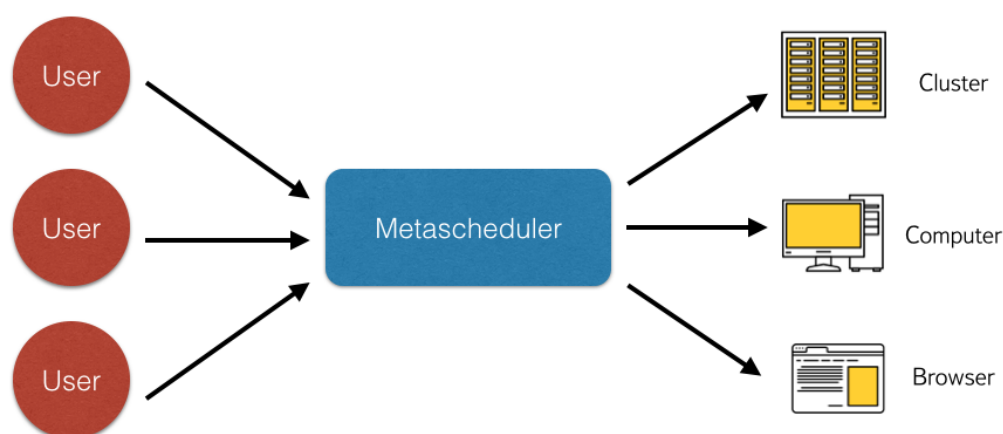


Figure 1. Skygrid architecture

Users describe their jobs as JSON documents called job descriptors. Job descriptor typically contain some parameters for job execution, URIs of input files and destination where to upload



output files. Job descriptors are submitted to one of the queues in Metascheduler.

Metascheduler contains multiple queues which are associated with job types. Jobs(their descriptors) are pushed to the queues by users and then pulled by executors. Internally queue could have basic FIFO structure or it could perform some advanced scheduling which account computation cost and data availability. Once submitted each job is associated with unique job id. Metascheduler provides REST API to check/update job status.

Executors in skygrid could be basically anything which knows how to execute the job of given type. Examples of executors include

- Clusters with internal scheduling
- Single computer node
- Application (e.g. web-browser)

Two job executors were built on top of this architecture: Docker container executor and distributed in-browser computing executor.

2. Docker executor

Docker is an open-source project that automates the deployment of applications inside software containers, by providing an additional layer of abstraction and automation of operating-system-level virtualization on Linux.

With Docker one can pack an application and all its dependencies(OS, binaries, libraries, data) into software container and then execute it on host with different set of dependencies inside the container.

Good HEP-related example is Monte-Carlo generation: one can build a container with all software involved in Monte-Carlo generation such as Pythia and Geant with suitable Linux distribution such as Scientific Linux and then execute in on any convenient host with, for example, Ubuntu Linux.

There are multiple advantages in using Docker for computations sandboxing. First of all, one can strictly fix computation environment to make guarantees on computations reproducibility. Secondly, computations could be performed on platform which is supported on any particular grid site. Finally Docker applications run as userspace processes so there is no penalty on such virtualization.

Skygrid provides capabilities to execute Docker containers in distributed manner. The scheme is the next:

- (i) In job descriptor user specifies container to run, command line inside it, input files location and destination of output files.
- (ii) User submits job descriptor to Metascheduler.
- (iii) Executor pulls job descriptor from Metascheduler.
- (iv) Executor downloads container, mounts input files inside it and starts the container.
- (v) Executor finishes container execution, uploads output data and marks job in Metascheduler as completed.
- (vi) User obtain output data from specified location.

3. Distributed in-browser computing

Another thing we are approaching with Skygrid is volunteer computing. With @HOME projects people can install BOINC client on their home computers and contribute to scientific computations. The advantage of this approach is that any software can be deployed on volunteers computers inside virtual machines. The disadvantage here is that volunteers should install software (Virtualbox and BOINC client) on their machines.

With Skygrid we are taking another approach to this problem. We call it Distributed in-browser computing or DiBroCop. The only requirement for volunteer machine in DiBroCop is presence of web-browser which is common now. Computing in DiBroCop can be decomposed to next steps:

- (i) Software which would run in volunteers browsers is compiled into JavaScript.
- (ii) Volunteer opens web page, which pulls a DiBroCop job from Skygrid.
- (iii) Job is executed in browser and results are reported back by AJAX request.

With DiBroCop's approach it is really easy to run computations for users but there is difficulty with cross-compiling software to JavaScript. This is visualized on Figure 2.

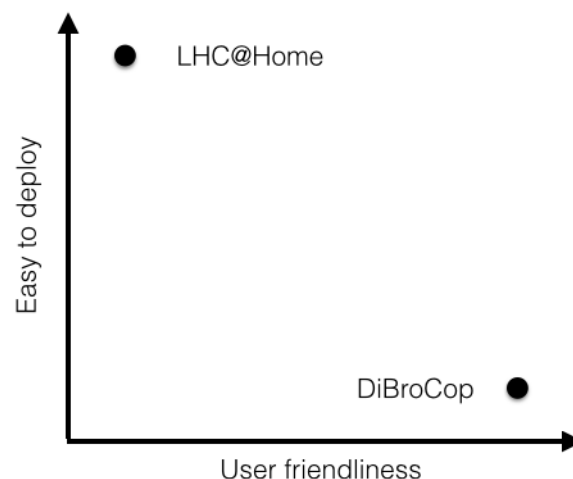


Figure 2. @HOME projects and DiBroCop could be used to achieve different goals.

4. Conclusion

Skygrid is currently deployed with one 1700-CPU(60 machines) computing site, which is used to generate and reprocess Monte-Carlo for SHiP experiment. All computations are being run in docker containers. With DiBroCop we successfully compiled Pythia to JavaScript through LLVM. One can test MC generation in browser on <http://xni.github.io/pythia/main01.html>

Skygrid is opensource and being developed on Github: <https://github.com/anaderi/skygrid>