Contribution ID: **309** Type: **Sectional reports**

Supporting Efficient Execution of Many-Task Applications with Everest

Monday 10 September 2018 14:15 (15 minutes)

Distributed computing systems are widely used for execution of loosely coupled many-task applications. There are two important classes of such applications. Bag-of-tasks applications, e.g., parameter sweeps or Monte Carlo simulations, represent a set of independent tasks. Workflows, which are used for automation of complex computational and data processing pipelines, consist of multiple tasks with control or data dependencies. The report discusses the common problems related to the efficient execution of such applications on distributed computing resources and the relevant solutions implemented within the Everest platform.

Everest [1-3] is a web-based distributed computing platform which provides users with tools to publish and share computing applications as web services. The platform also manages the execution of applications on remote computing resources. Everest implements the PaaS model by providing its functionality via remote web and REST interfaces. A single instance of the platform can be accessed by many users in order to create, run and share applications with each other. Instead of using a dedicated computing infrastructure, Everest performs the execution of applications on external resources attached by users. The platform supports integration with standalone servers, clusters, grid infrastructures, desktop grids and clouds. A user can specify multiple resources, possibly of different type, for running an application.

Everest provides multiple tools for execution of many-task applications. First, it includes a general-purpose service for execution of bag-of-tasks applications such as parameter sweeps. The application tasks are described using a simple declarative notation. Second, it is possible to dynamically add new tasks or invoke other applications from a running application via the REST API. This allows users to run complex many-task applications such as workflows. In this case, the dependencies between tasks are managed internally by a user application. While this approach provides maximum flexibility, it does not allow passing the complete task graph to the platform to enable scheduling optimizations. To overcome this limitation, an new interface for submitting workflows has been added recently.

The application tasks are executed by Everest on computing resources specified by a user. The efficiency of application execution, i.e. the execution time, critically depends on the methods used for task scheduling [4]. Everest implements a two-level scheduling mechanism that allows to plug-in different scheduling algorithms. First, the available resources are fairly distributed among the running applications. Then, the application-level scheduler selects tasks for running on provided resources. The separate schedulers are implemented for bags-of-tasks and workflows, which are based on MaxMin and DLS algorithms respectively. The used algorithms require the estimates of task execution and data transfer times. Currently, these estimates are computed based on the statistics from previous task and application executions.

The other features that are essential for efficient execution of many-task applications include accounting for local resource policies and automatic recovery of failed tasks. For example, the limit on the maximum number of jobs per user imposed by an HPC cluster administrators may not allow to fully utilize the resource when running a single job per Everest task. An advanced adapter for Slurm manager has been developed which allows to solve this problem by submitting complex jobs consisting of multiple tasks. When dealing with failed tasks, Everest distinguishes between critical and recoverable faults. In the latter case, the task is retried multiple times, and the resources with many failures are blacklisted. To account for temporary network failures between Everest and resources, the tasks running on the disconnected resource are not rescheduled immediately to avoid wasting compute time.

1. Everest. http://everest.distcomp.org/

2. Sukhoroslov O., Volkov S., Afanasiev A. A Web-Based Platform for Publication and Distributed Execution of Computing Applications // 14th International Symposium on Parallel and Distributed Computing (ISPDC). IEEE, 2015, pp. 175-184.
3. Sergey Smirnov, Oleg Sukhoroslov, and Sergey Volkov. Integration and Combined Use of Distributed Computing Resources with Everest // Procedia Computer Science, Volume 101, 2016, pp. 359-368.
4. Nazarenko A., Sukhoroslov O. An Experimental Study of Workflow Scheduling Algorithms for Heterogeneous Systems. In: Malyshkin V. (eds) Parallel Computing Technologies. PaCT 2017. Lecture Notes in Computer Science, vol 10421. Springer, Cham, 2017, pp. 327-341.

Primary author: Dr SUKHOROSLOV, Oleg (IITP RAS)

Co-authors: Mr SMIRNOV, Sergey (Institute for Information Transmission Problems of the Russian Academy of Sciences); Mr VOLKOV, Sergey (IITP RAS)

Presenter: Dr SUKHOROSLOV, Oleg (IITP RAS)

Session Classification: 7. Desktop grid technologies and volunteer computing

Track Classification: 1. Technologies, architectures, models of distributed computing systems