Russian Supercomputing Days (September 24-25, 2018, Moscow, Russia)

Using Resources of Supercomputing Centers with Everest Platform

Sergey Smirnov, *Oleg Sukhoroslov*, Vladimir Voloshinov

Institute for Information Transmission Problems
of the Russian Academy of Sciences (Kharkevich Institute)

25.09.2018

Motivation

- Computational methods are widely used for solving complex scientific and engineering problems
  - Require the use of high-performance computing resources
  - On-premises clusters, supercomputing centers, distributed computing infrastructures, clouds
- Supercomputing centers is an important source of HPC resources
  - Significant amount of resources
  - Many users and projects, queues and wait times
  - Support efficient execution of tightly coupled parallel applications
  - Use specialized hardware
  - Generally free for scientific projects

Motivation

- The wide use of HPC resources among scientists is complicated due to a number of problems
- Lack of convenient interfaces for running computations
  - Low-level command line environment and batch system facilities
  - Demotivate researchers with less technical background
- Lack of tools for automation of routine activities
  - Execution of multiple jobs, parameter sweeps, complex workflows
  - Can be useful even for advanced users
- Other problems
  - Reliable execution of long-running computations
  - Massive computations spanning multiple resources
  - Accounting for resource characteristics and local policies

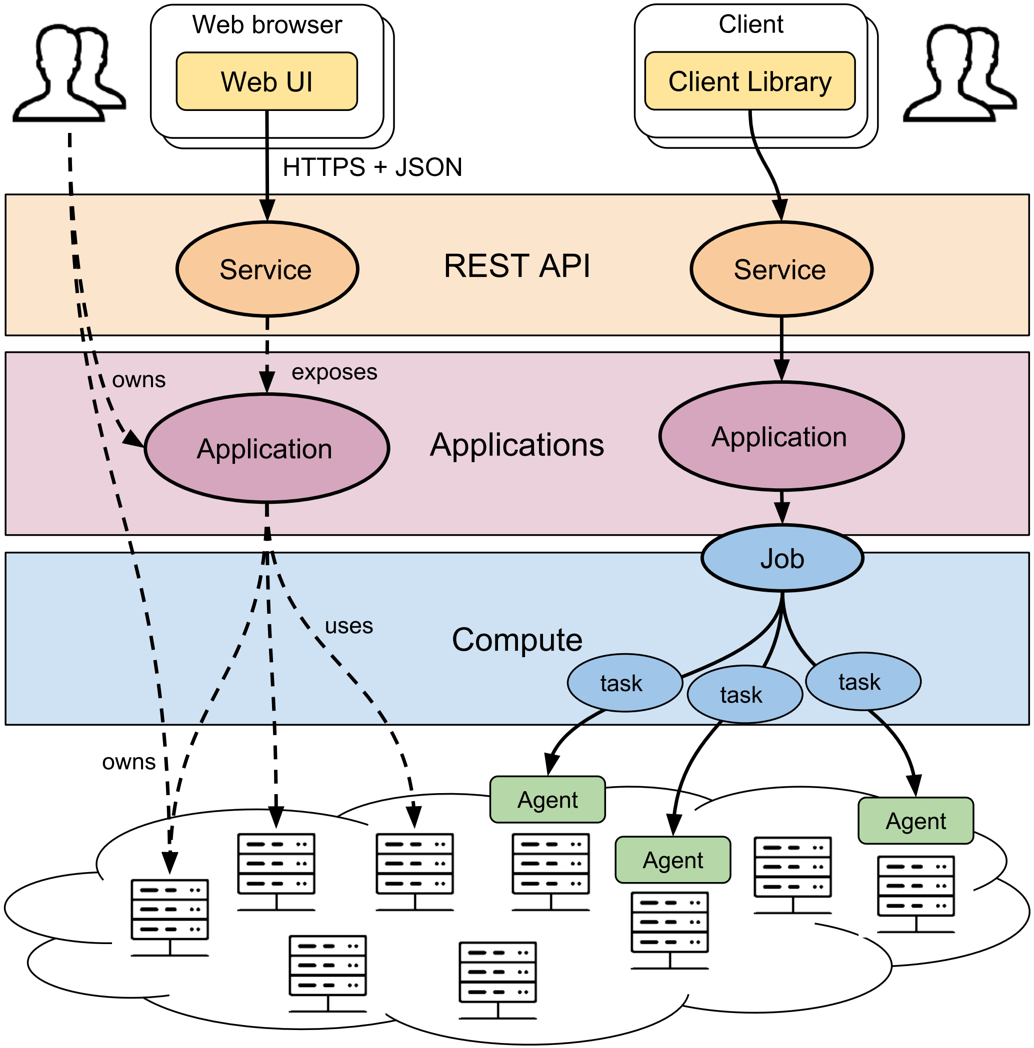
Related Work

- Web-based interfaces to HPC systems
  - Familiar and user friendly
  - Remote submission of parallel applications via web forms
- Grid portals and scientific gateways
  - Facilitating the access to distributed computing resources
  - Additional services, e.g. collaborative capabilities
  - Frameworks for development of computational portals
- Features missing in the first generation systems
  - Combine multiple applications, run multi-step workflows
  - Extension of functionality by users, e.g. publication of new applications
  - Enabling users to attach and access their own resources

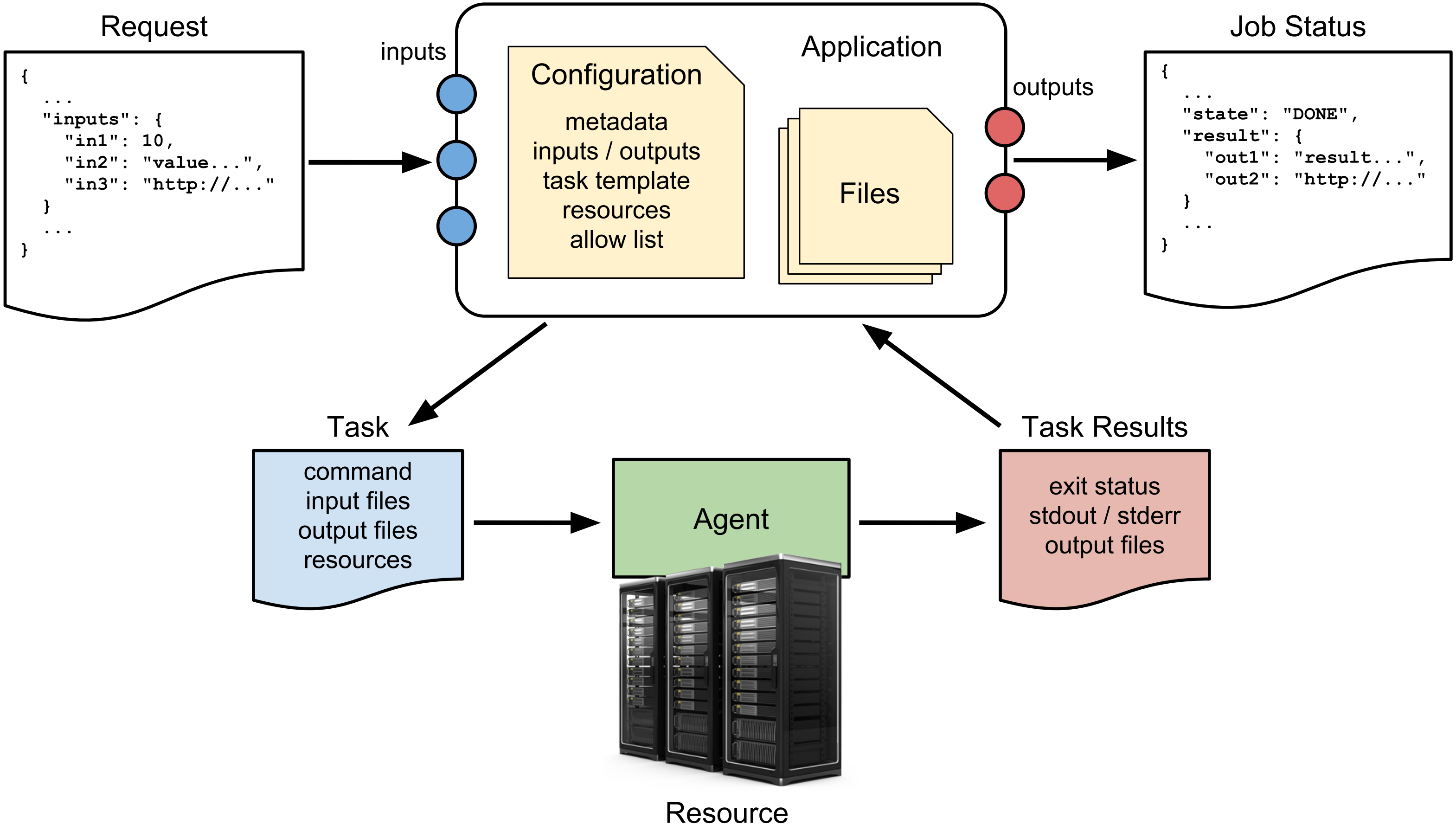
Everest

- Web-based platform supporting
  - Publication of computational applications as services
  - Execution of applications on external computing resources
  - Sharing applications and resources with other users
  - Composition of applications (workflows)
- Platform as a Service
  - Remote access via web browser and REST API
  - Single platform instance can be accessed by many users
  - No installation is required
- Public instance with open registration
  - http://everest.distcomp.org/

Architecture



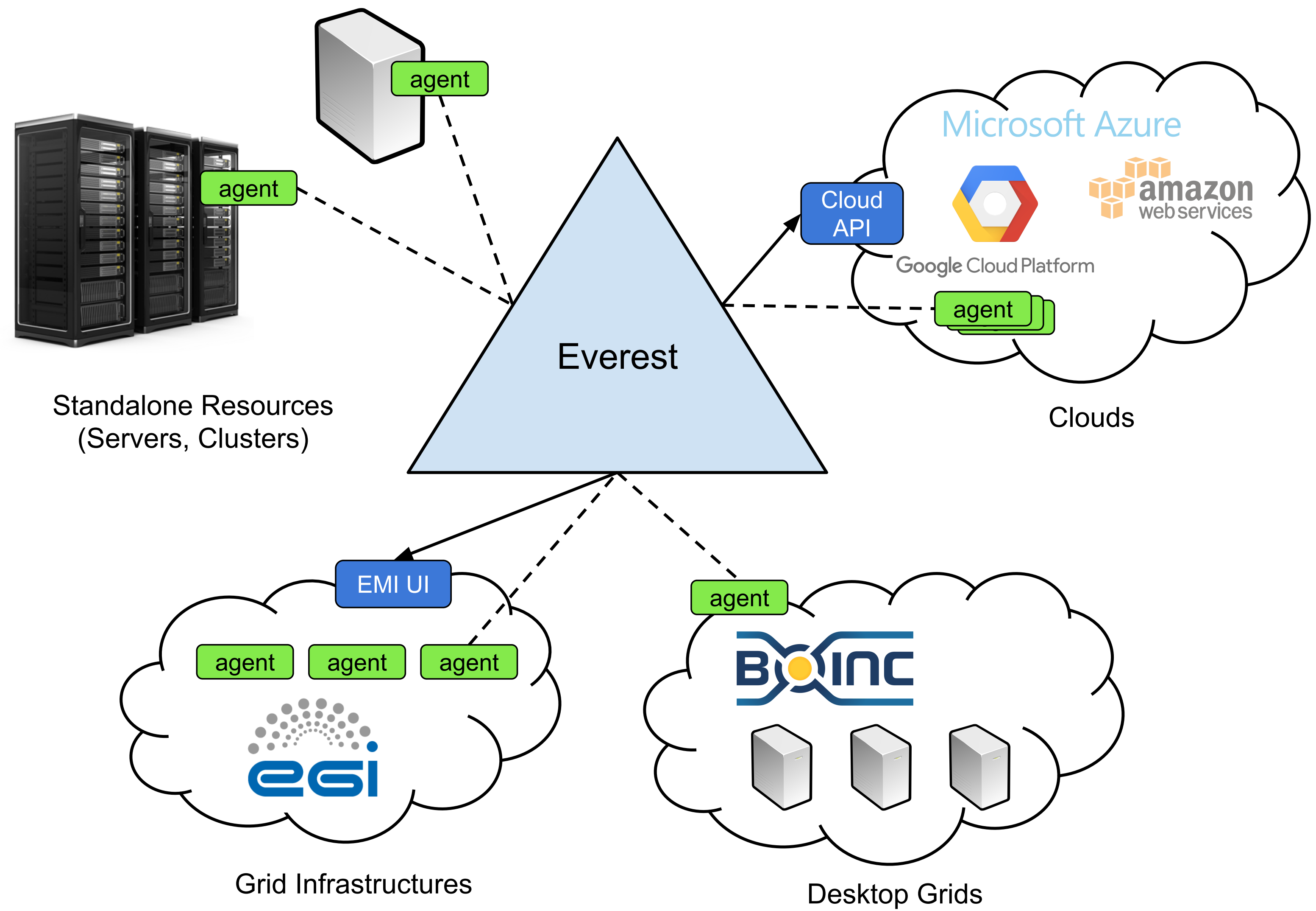
Application



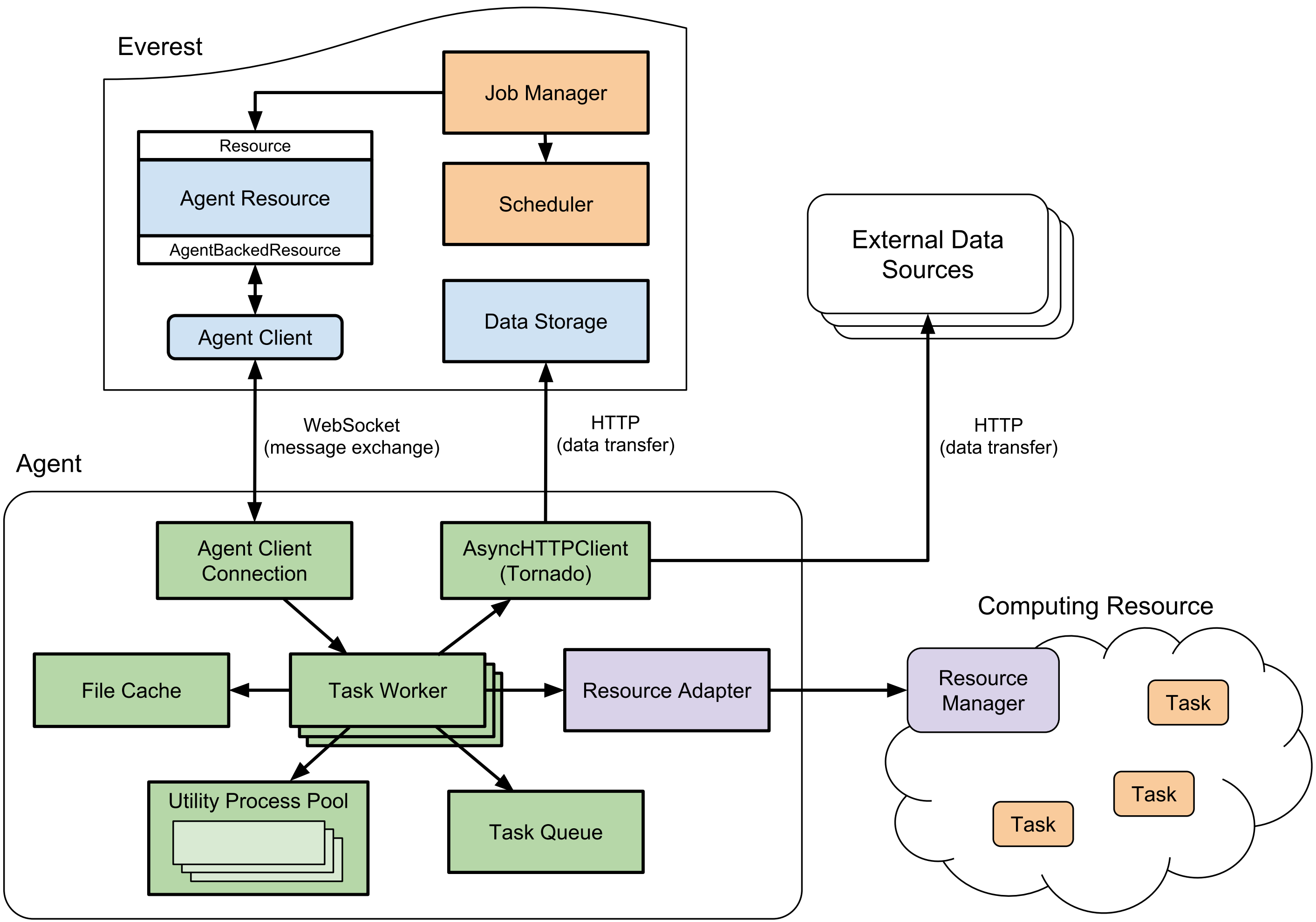
Supported Application Types

- Command
  - Generic template for applications with command-line interface
  - Single compute task
- Parameter Sweep
  - Large number of independent compute tasks with parametrized inputs
  - Generic service for running parameter sweep experiments
- Many-task Application
  - Multiple compute tasks, can be created dynamically
  - Task dependencies are managed by the application
- Workflow
  - Composition of multiple applications
  - Multiple jobs with dependencies

Computing Resurces



Agent


Everest
Job Manager
Resource
Agent Resource
AgentBackedResource
Scheduler
Agent Client
Data Storage
External Data Sources
WebSocket
(message exchange)
HTTP
(data transfer)
HTTP
(data transfer)
Agent
Agent Client Connection
AsyncHTTPClient (Tornado)
Computing Resource
File Cache
Task Worker
Resource Adapter
Resource Manager
Task
Task
Task
Utility Process Pool
Task Queue

Everest Improvements

- Supporting the efficient use of HPC resources via the platform
- Based on the experience of integration with several supercomputing centers in Russia
  - Data Processing Center of NRC Kurchatov Institute (NRC KI)
  - Supercomputer Simulation Laboratory of South Ural State University (SSL SUSU)
- Agent improvements
  - Accounting for the maximum number of jobs per user limit
  - Automatic tasks directory cleanup
  - Handling of job submission failures and timeouts
  - Propagation of environment variables
- Supporting advanced resource requirements

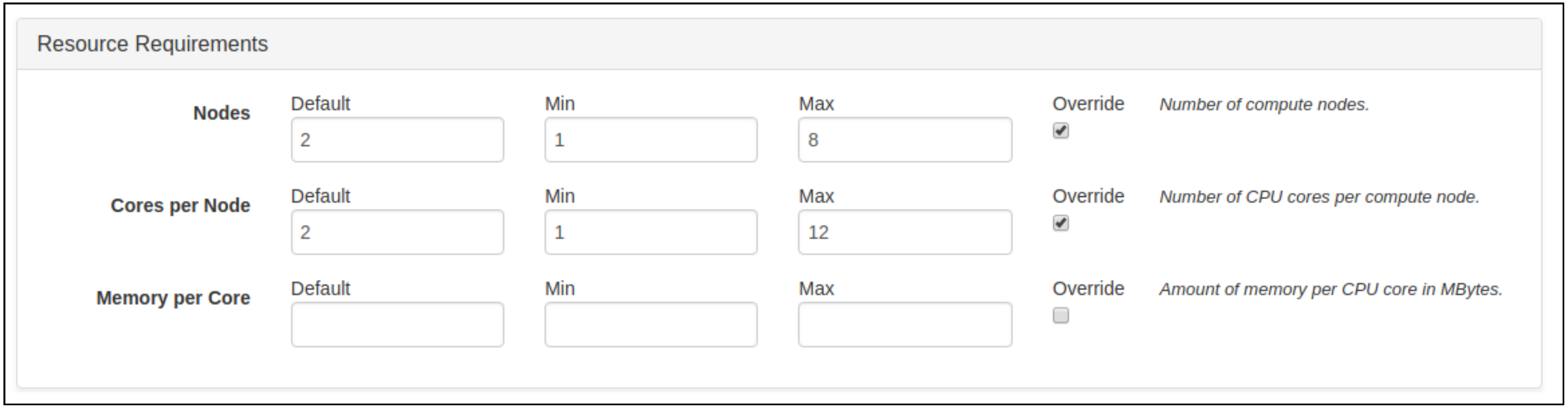
Accounting for Local Policies

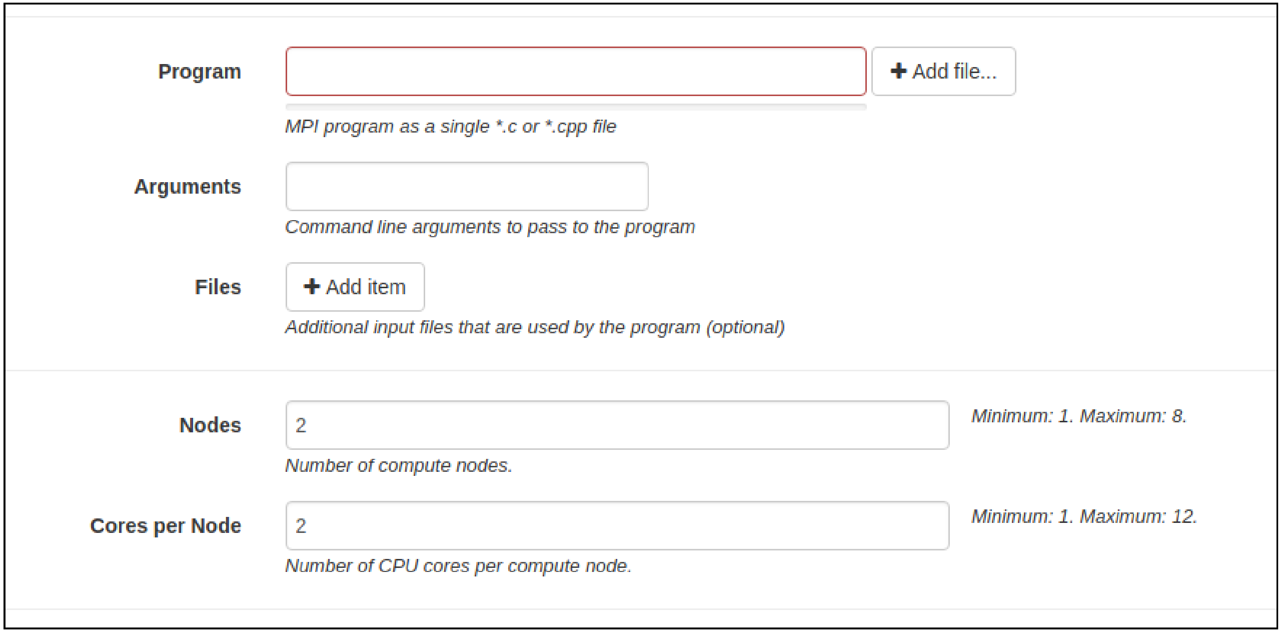
- NRC KI: the number of concurrent jobs per user is limited to 64
- The default Slurm adapter runs a single job per Everest task
  - For single-core tasks it is possible to utilize only 64 cores at once
- The new advanced Slurm adapter
  - Use complex Slurm jobs consisting of multiple tasks (job steps) started with `srun` command inside a job script
  - The tasks are grouped by their Everest job ID and are accumulated in the adapter
  - A complex job is submitted when a specified number of tasks has accumulated, or no new tasks have arrived within the specified time
  - The state of individual tasks is checked by reading a file, where the job script prints the return codes of completed tasks
  - The completed tasks can be processed before the whole job has completed
  - The cancellation is supported only for the entire batch of tasks

Advanced Resource Requirements

- Lack of explicit support for resource requirements in Everest
  - By default tasks were run as single-core jobs
  - The number of CPU cores per task can be specified only on the agent level
  - The execution of multi-node parallel applications (MPI) required the use of auxiliary scripts
- The limitation was removed by enabling the Everest users to specify the application resource requirements
  - Number of nodes, cores per node, memory per core
  - The application developer can specify default values, allow the users to override some values when running the application, restrict mininum and maximum values
- The platform and the agent were modified to take into account the resource requirements during the task scheduling and execution

Advanced Resource Requirements





Experimental Evaluation

- Loosely coupled many-task applications for solving global optimization problems
- DDBNB (Domain Decomposition Branch-and-Bound)
  - Everest application implementing a coarse-grained parallel version of the branch-and-bound algorithm
  - Based on preliminary decomposition of a feasible domain of the problem by some heuristic rules
  - Subproblems obtained via decomposition are solved by a pool of standalone BNB solvers (SCIP, CBC)
  - The incumbent values found by each solver are intercepted and delivered to other solvers

Voloshinov V., Smirnov S., Sukhoroslov O. Implementation and Use of Coarse-grained Parallel Branch-and-bound in Everest Distributed Environment // Procedia Computer Science. Volume 108, 2017, pp. 1532-1541.

Experimental Evaluation

- Initial experiments: solving the Travelling Salesman Problem in a heterogeneous environment (standalone servers, virtual machines)
- The integration with HPC resources allowed to try DDBNB for solving hard global optimization problems
- Tammes and Thomson problems
  - Well known problems in combinatorial geometry
  - Concern the arrangement of N points on a unit sphere
  - Tammes: maximize the minimal distance between any pair of points
  - Thomson: minimize the electrostatic Coulomb energy
- Experiments on HPC4 cluster at NRC KI
  - Tammes N=8: 128 tasks, 100 minutes
  - Tammes N=9: 256 tasks, 3 days

Conclusion

- The integration of Everest platform with HPC resources of supercomputing centers has been presented
  - A number of improvements has been made in order to support the efficient use of such resources and to solve the common problems
  - The platform enables convenient access to HPC resources, execution of many-task applications, and pooling of multiple resources
- The described implementation has been tested by solving hard global optimization problems with the DDBNB Everest application
  - The use of supercomputing centers has provided the significant increment in computing power available for the experiments
- Future work: integration with other supercomputing centers, improving the described implementation, large-scale experiments involving resources of multiple centers