Improving the load of supercomputers based on job migration using container virtualization

Stanislav Polyakov
Skobeltsyn Institute of Nuclear Physics, Moscow State University

Supercomputers and jobs

- Supercomputers can have thousands to tens of thousands of processors.
- Jobs can have dimensions varying from less than a minute to multipple days and from a single core to hundreds or thousands of processors.

  Thus scheduling is essential to keep the load (average number of active computational nodes) high.

Scheduling problems

Varying sizes of jobs and jobs aborting or completing earlier than expected pose a challenge to schedulers. Some schedulers also have optimization objectives other than load, such as average waiting or turnaround time or maintaining locality.

As a result, average load of supercomputers seldom exceeds 90% and might be as low as 70%.

Opportunistic use of supercomputer resources

Supercomputer load can be increased by supplying additional low-priority jobs with small dimensions (a job with a size of minimal scheduler slot can fit into any gap left by the scheduler).

This approach is used, for example, to run ATLAS problems on Titan 2 supercomputer.

Numerous problems do not require parallelization and the respective jobs can run on a single core, however it is less common for them to also require short execution time.

Container virtualization

Operating system like Linux allows running multiple isolated user-space instances using a single kernel. These instances are called containers. Containers have evolved from Linux `chroot` mechanism and gained popularity with the advent of container platform Docker.

Advantages that made containers popular include small overhead and starting time.

Container virtualization features

Typically container virtualization platforms use copy-on-write mechanism that allows to record only the changes made to a container file system, resulting in very small sizes of the images.

Some platforms allow live container migration.

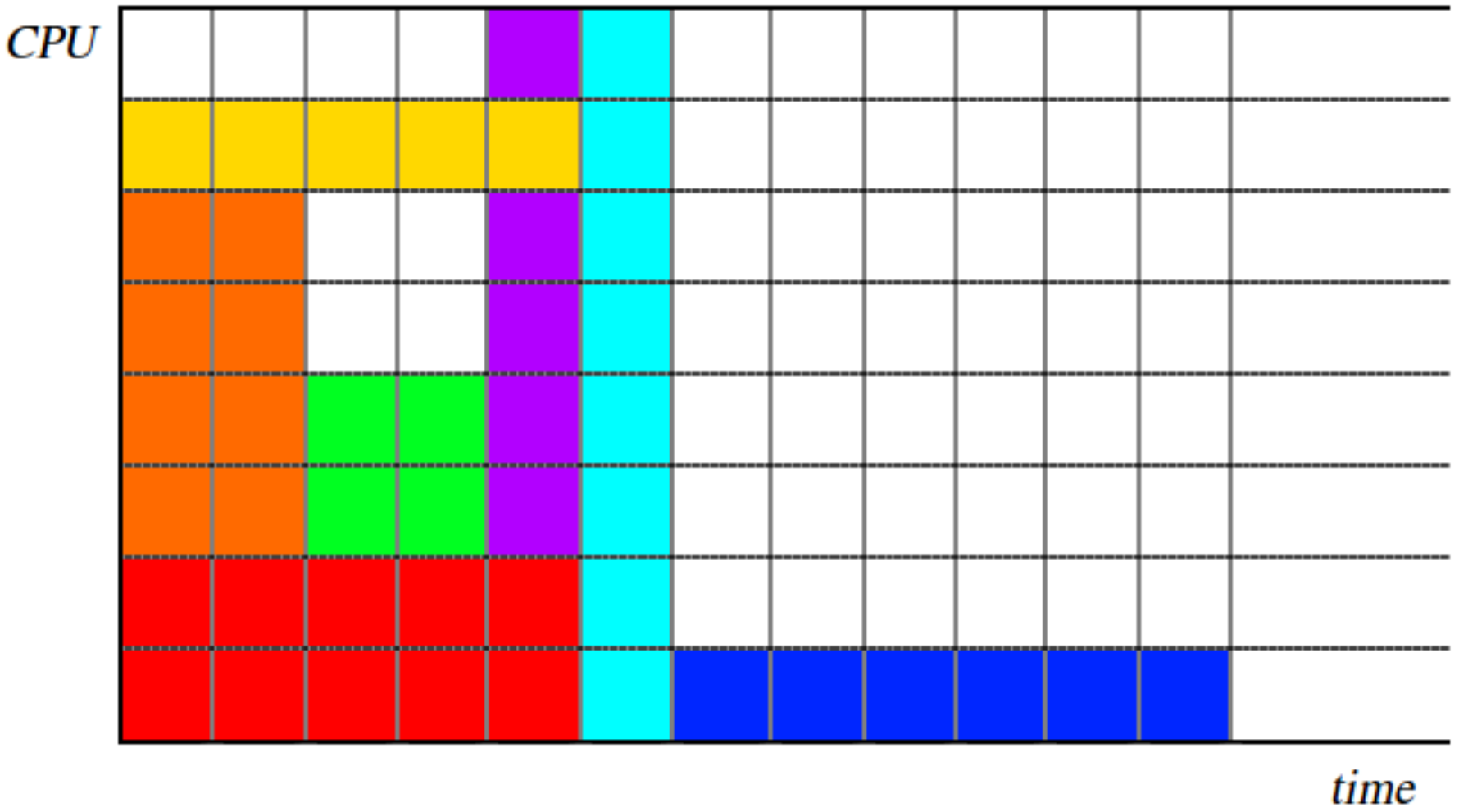
Opportunistic use of idle supercomputer resources with container virtualization

The idea: deploy single-core jobs wrapped in containers on idle supercomputer nodes, then migrate them to other nodes or return back to the queue before the allotted time is over.

With sufficient amount of such jobs 100% load can be reached, and thanks to the short starting time of the containers even very short schedule gaps can be used efficiently.

Backfill

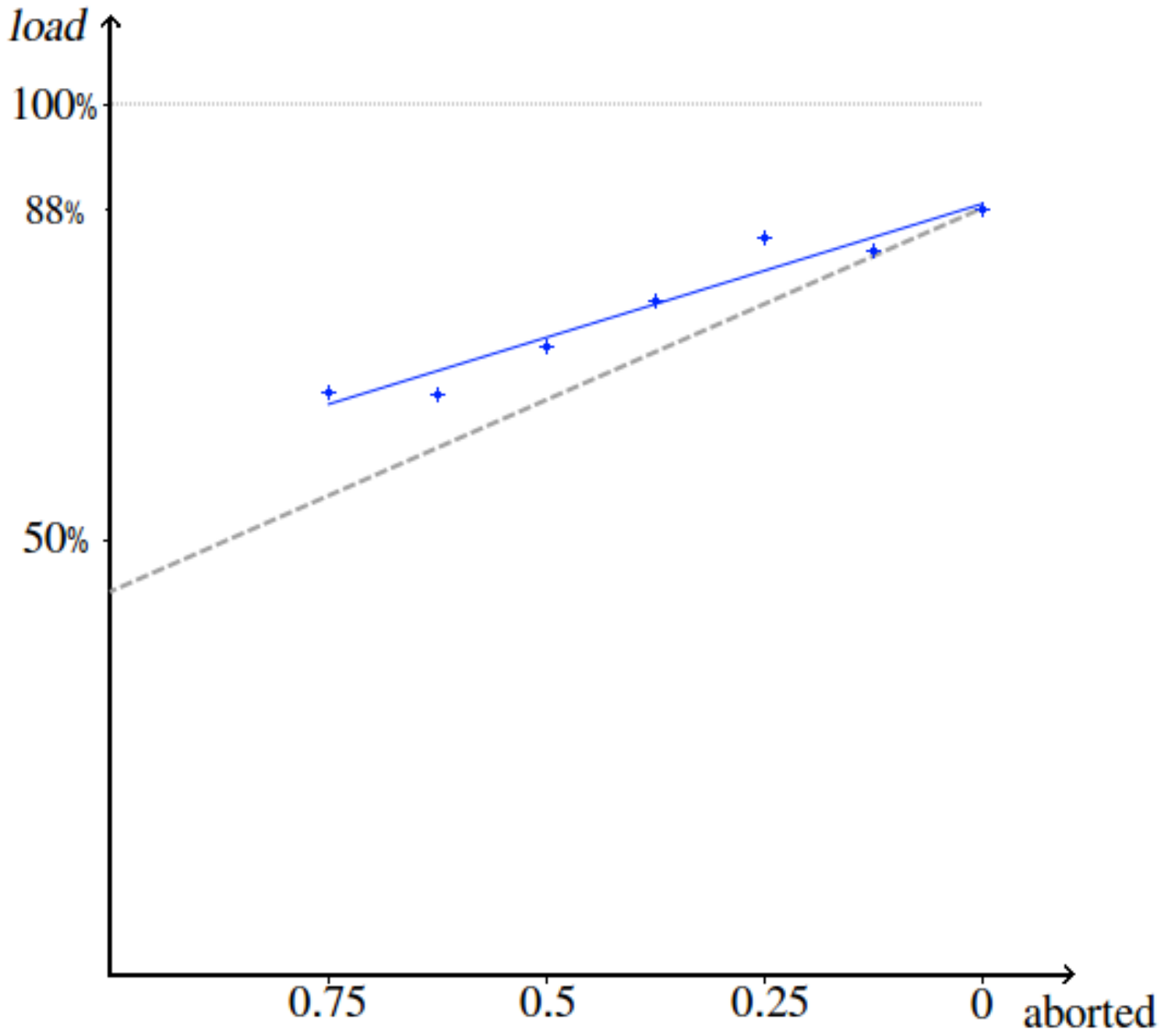
Backfill is a scheduling algorithm that assigns the earliest possible time to the highest priority job and then attempts to fill the resulting schedule gap with the other jobs from the queue. A “greedy” variation of the algorithm tried to add the jobs in order of their priority, and another variation tries to find job(s) with dimensions that best fit the schedule gaps.


CPU
time

The estimate of the load increase

To estimate the potential benefits of the proposed system, the “greedy” version of backfill algorithm was run for the simulated queue of supercomputer jobs (the dimensions were generated with realistic distribution) to create a schedule for 512 processing units and 1440*360 time units.

On each run, a different probability for early job completion/abortion was assumed (same for all jobs longer than minimal schedule slot, with uniform distribution of early completion time).


load
100%
88%
50%
0.75
0.5
0.25
0
aborted