Assessing Job Wrapping as an Strategy for Workflow Optimization on Shared HPC Platforms

Manuel G. Marciani[1,2](manuel.gimenez@bsc.es), Gladys Utrera[2,1], Miguel Castrillo[1], Mario C. Acosta[1,2], Francisco Doblas-Reyes[1,3]

1: Barcelona Supercomputing Center (BSC) Barcelona, Spain 2: Universitat Politècnica de Catalunya (UPC) Barcelona, Spain 3: Institució Catalana de Recerca i Estudis Avançats (ICREA) Barcelona, Spain

Introduction and Motivation

Recently, the Earth Sciences community has begun to consider the performance of the overall execution of simulations, both in terms of waiting time and restarting failed jobs [1].

To tackle queue time, Autosubmit [4] developers came up with the idea of bundling multiple subsequent and/or independent jobs into a single submission.

In this paper, we test whether wrapping saves queue time under different usage scenarios.

Methods

We use the BSC Slurm simulator [2], which is an open source software for simulating the popular workload manager. To model the environment we use two types of traces: static and dynamic.

We generate static traces following the distribution of jobs from the LUMI supercomputer. We then introduce a single job, track its queue time, and vary its geometry and the fair share of the user.

As for the dynamic workloads, we used a real machine trace [3] in which we include two workflows: a vertical one with 7 tasks and a horizontal, with 15 tasks. We simulate each workflow twice, wrapped and unwrapped, and vary the fair share of the user.

Wrappers

Wrapping, in its two elemental types is depicted below, where we have an outer rectangle indicating the jobs that are submitted in a bundle. Additionally, there is the option of combining vertical and horizontal wrappers.

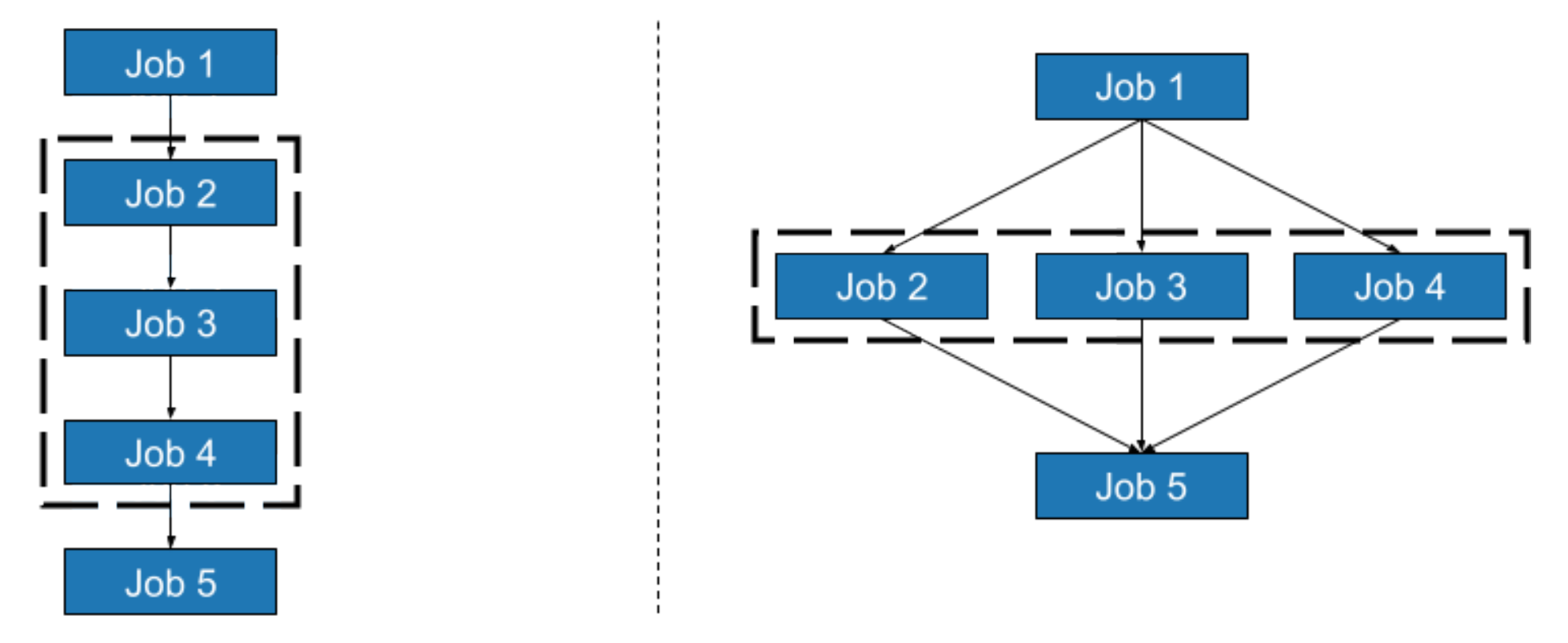


Figure 1: Horizontal and vertical wrappers example.

Fair tree

The *Fair tree* is one algorithm available in Slurm to compute fair share and its main feature is the sharing of machine responsiveness among users in the same account.

The algorithm is a *depth first traversal* of the user tree starting from the *root* account, ordering decreasingly its children by their *Level Fair Share*, which is the ratio of the shares given by the system adminstrators and the usage. Then it recurs if it evaluates an account. Otherwise, it assigns a fair share.

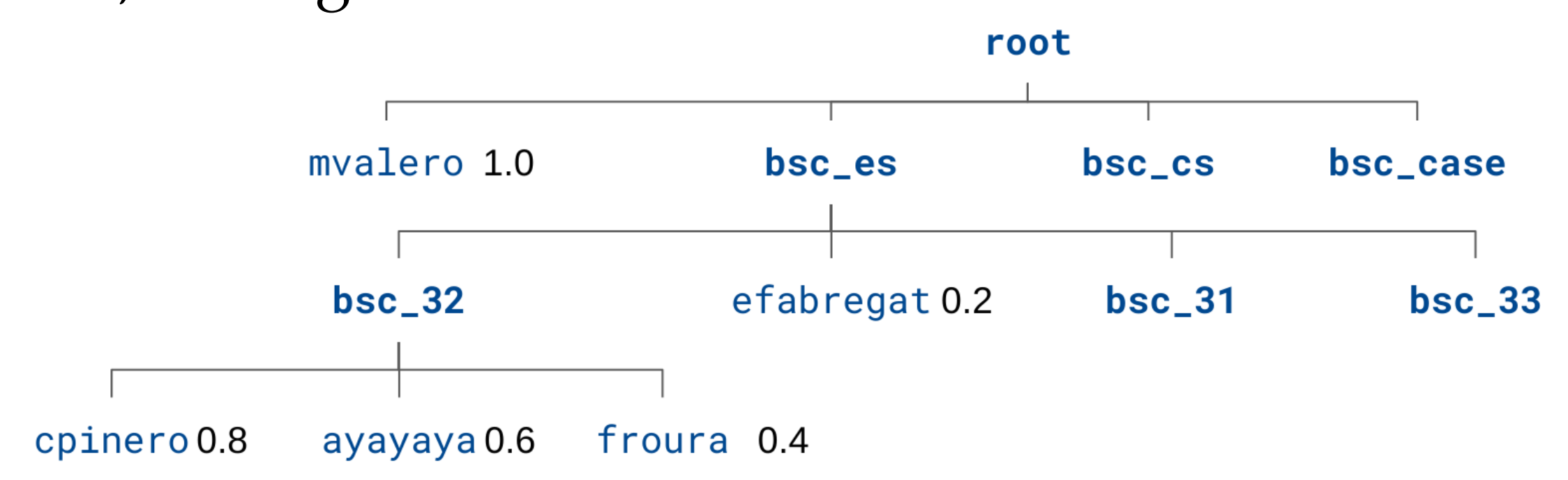


Figure 2: User and Account tree example.

Scheduling in Slurm

Under the *multipriority* configuration, Slurm computes priority from a handful of factors. The main ones are:

- *Age* factor, computed as

$$a_i = \frac{t}{T}. \tag{1}$$

- *Size* factor, computed as

$$s_i = \frac{r_i}{S} \text{ or } s_i = \frac{S - r_i + 1}{S}. \tag{2}$$

- *QoS* factor, which stands for *Quality of Service*, computed as

$$q_i = \frac{q_u}{max\,\{q \mid q \in Q\}}. \tag{3}$$

- *Fair share* factor, explained above, which is the *quantification of a user's right to the machine.*

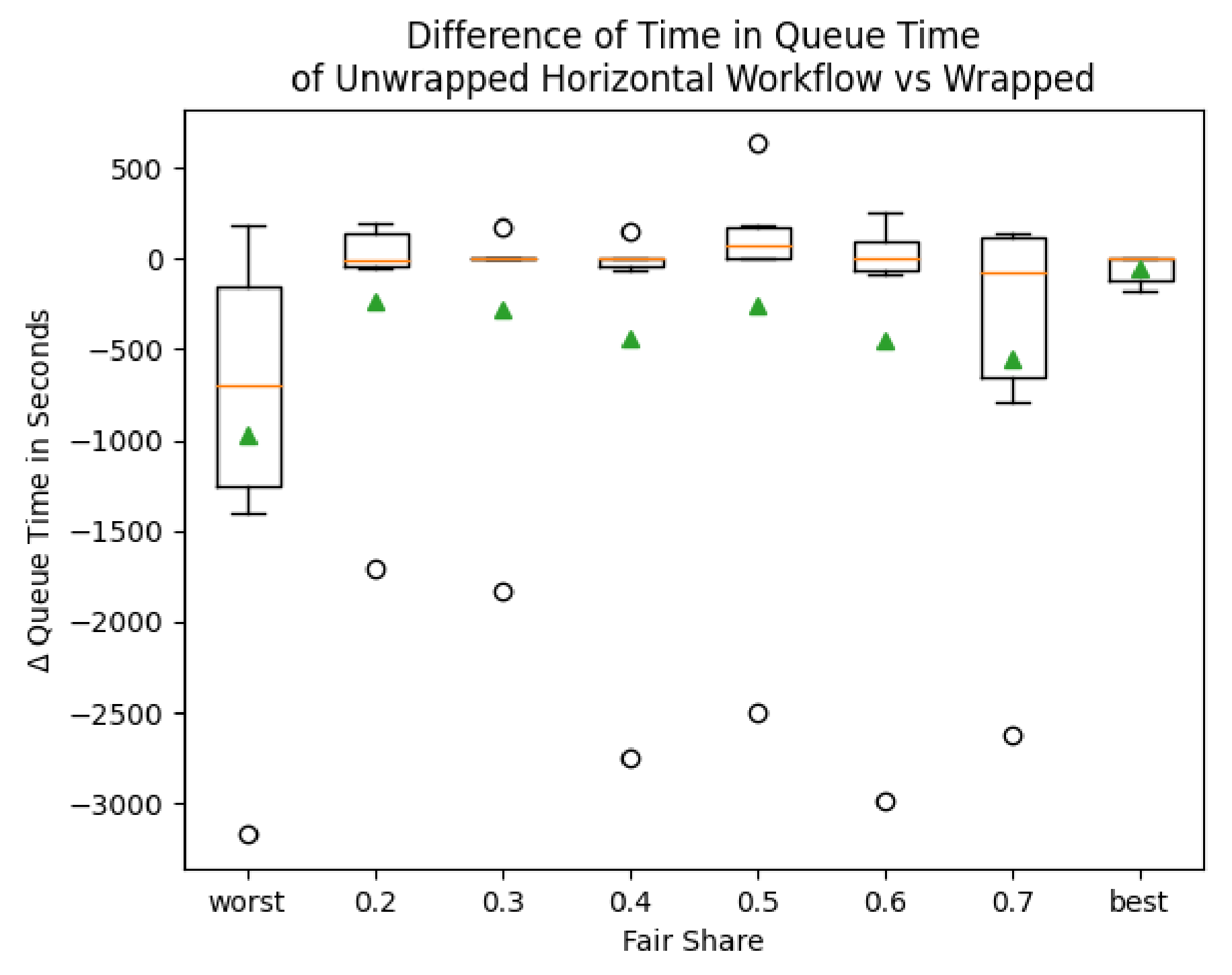
With all of these, the priority is the weighted sum of the factors,

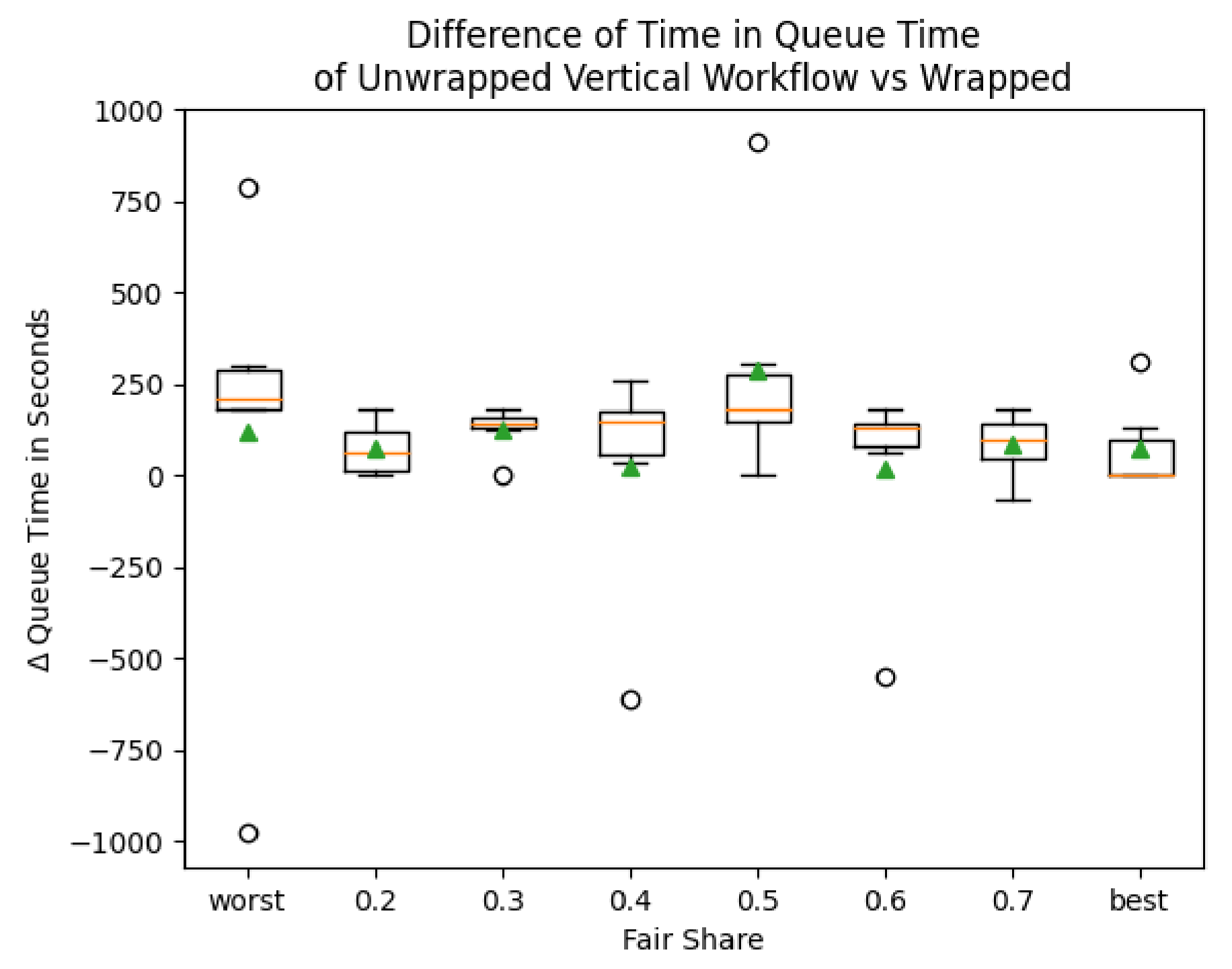
$$P_i(t) = a_i \cdot w_a + s_i \cdot w_s + f_i(t) \cdot w_f + q_i w_q, \tag{4}$$

where weights are set by system adminstrators.

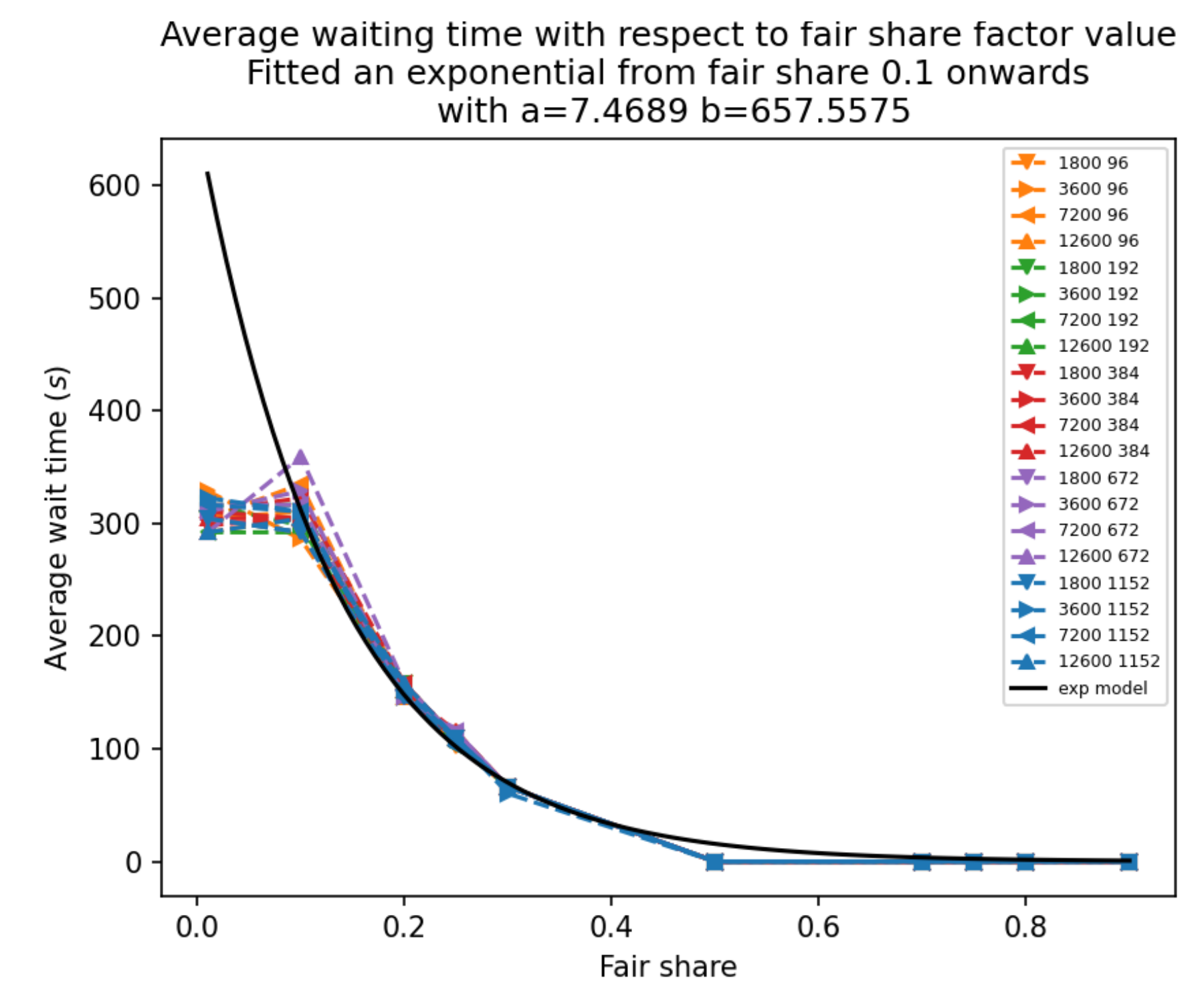
Results

For the dynamic workload we have:





For the static workloads we have, where each line represents a different job geometry.



Conclusions

If the user has a high fair share, *it should submit tasks wrapped* and, if the user has a vertical workflow, it should *aggregate them onto a single job* – regardless of the fair share – with queue time savings, under our scenarios, with up to 6% saving in the total runtime of the tasks.

As for the horizontal case we see in the plots that it undermines, on average, with extreme cases up to 1.5x the runtime of the workflow more of queue time. We expected wrapping to excel under such conditions however we have that backfill is the one responsible by scheduling. Hence, increasing the task size undermines the odds of finding a suitable space for the job.

References

[1] Mario Acosta. Is-enes3 d4.3 - cpmip performance metrics and community advice, 12 2021.

[2] Ana Jokanovic. Evaluating slurm simulator with real-machine slurm and vice versa. *Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS18)*, 2018.

[3] D. G. Feitelson and D. Tsafrir. Logs of real parallel workloads from production systems, 9 2019.

[4] Domingo Manubens-Gil. Seamless management of ensemble climate prediction experiments on hpc platforms. In *2016 International Conference on High Performance Computing and Simulation (HPCS)*, 2016.

Acknowledgements

This project has received funding from the **European High-Performance Computing Joint Undertaking (JU)** under grant agreement **Nº 955558**, from the **European Union's Horizon Europe research and innovation programme** under the grant **Nº 101093293**, and from the grant **CEX2021-001148-S-20-5** funded by

MINISTERIO DE CIENCIA, INNOVACIÓN Y UNIVERSIDADES
UNIÓN EUROPEA Fondo Social Europeo El FSE invierte en tu futuro
AGENCIA ESTATAL DE INVESTIGACIÓN

Funded by the European Union