New Multi-resource Fairshare Prioritization Mechanisms for Heterogeneous Computing Platforms

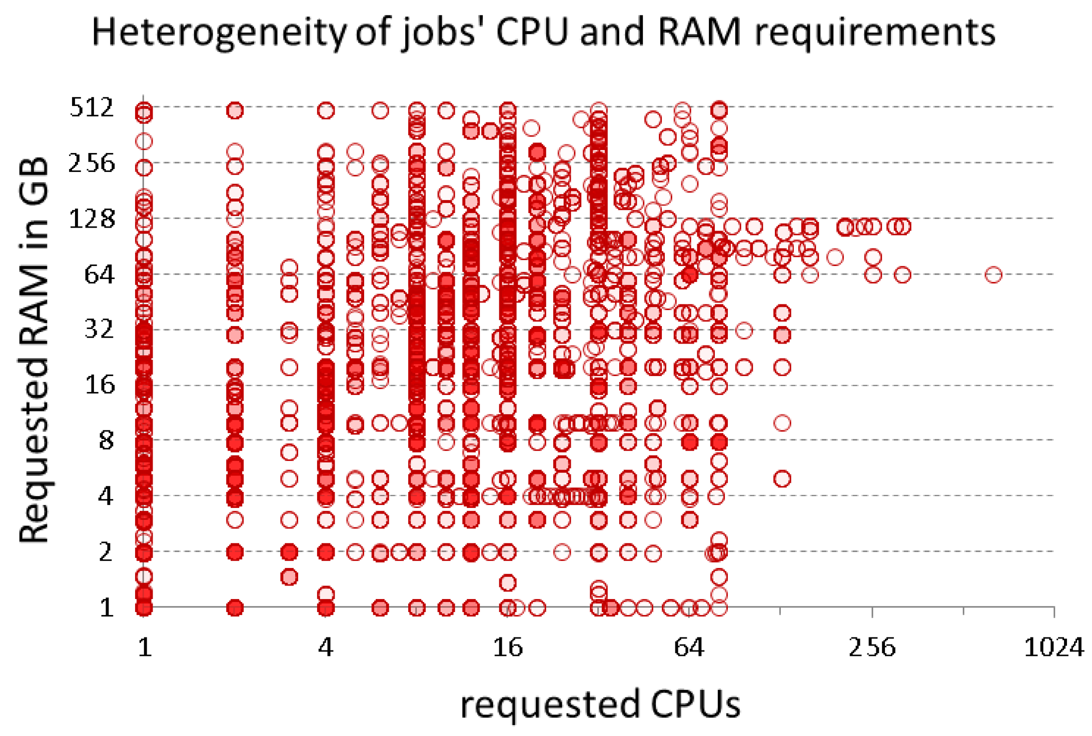
Dalibor Klusáček[1,2] and Hana Rudová[2]
[1]CESNET, Czech Republic — [2]Faculty of Informatics, Masaryk University, Czech Republic
klusacek@cesnet.cz, hanka@fi.muni.cz

1. Introduction

Existing Grid resource managers usually support only **CPU-aware fairshare** mechanisms to guarantee fairness, disregarding other consumed resources.

We presents newly developed **multi resource-aware prioritization mechanism for guaranteeing fairness in heterogeneous Grid-like systems**. This solution is currently being implemented in the TORQUE resource manager used within the Czech national Grid infrastructure *MetaCentrum* [1].

2. Multi Resource-aware Fairshare



CPU-based fairshare prioritization is not suitable for highly heterogeneous jobs and resources [2].

- $penalty_j = reqCPUs_j \cdot walltime_j$
- users are prioritized wrt. their previously consumed CPU time
- **high RAM, HDD, GPU, etc., requirements are not adequately penalized**

3. Drawbacks of Existing Solutions

Dominant Resource Factor (DRF) and **Bottleneck-based Fairness (BBF)**:

- schedule jobs according to the maximum/bottleneck resource requests
- **unrealistic assumptions that all jobs and resources are infinitely divisible**

Processor Equivalent (PE):

- $PE_j = \max(\frac{reqCPUs_j}{availCPUs}, \frac{reqRAM_j}{availRAM}) \cdot availCPUs$
- **not suitable (unfair) for heterogeneous systems** since PE_j may vary per machine
- user's priority (based on PE_j) may highly depend on scheduler's decisions

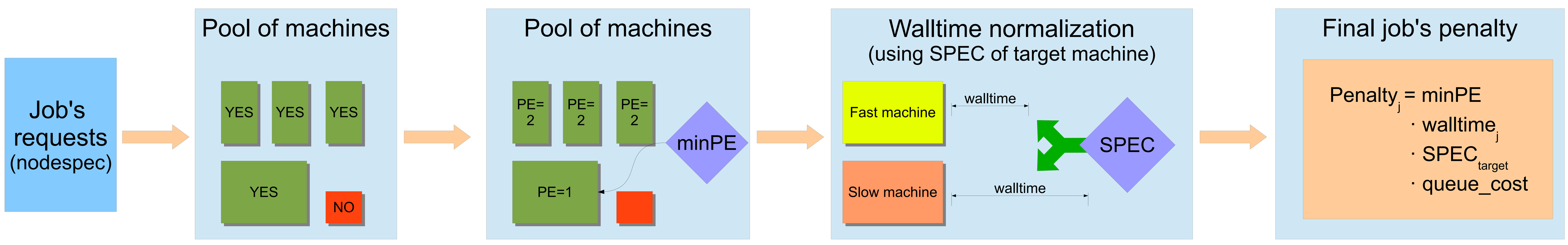
4. Proposed Solution

Main features of the proposed solution:

- **based on an extended PE metric**
- **a multi resource-aware mechanism**
- **fair regardless jobs' and machines' heterogeneity**
- **reflects various machines' speeds**
- **a job's penalty (i.e., user's priority) is not scheduler-dependent**

5. Computation of Fair Job's Penalty over Heterogeneous Resources

The process of penalty calculation



1. Find all suitable machines for a job **2.** Find minimal PE on such machines **3.** Normalize job's walltime **4.** Apply queue cost **5.** Penalty is a product of values

Penalty computation ($penalty_j$) for a given job j and m suitable machines:

- $PE_{j,i} = \max(\frac{reqCPUs_j}{availCPUs_i}, \frac{reqRAM_j}{availRAM_i}) \cdot availCPUs_i \quad (\forall i \text{ such that } 1 \leq i \leq m)$
- $penalty_j = \min(PE_{j,1}, .., PE_{j,m}) \cdot walltime_j \cdot SPEC_{target} \cdot queue\_cost$

Important notes:

- the actual *target* machine(s) where a job is executed may be different from the "cheapest" one
- **a job's "cost" only depends on user's requests and is independent from scheduler's decisions**
- **a job's walltime is normalized** according to the speed of the target machine, eliminating speedup effects of faster machines

6. Conclusion and Future Work

The proposed solution is currently being implemented within *MetaCentrum*'s TORQUE scheduler.

- **it replaces current standard CPU-based fairshare mechanism**
- parameter setup is currently analyzed (SPEC benchmarks for walltime normalization, queue costs, multi-node penalty, etc.)

In the future, **we plan to extend penalty metric to consider other important (but often less restricting) resources like GPUs and HDD.**

- will require modification of $PE_{j,i}$ calculation as, e.g., GPUs are not required by all jobs in the system
- a machine with fully used GPUs may still execute other jobs

Acknowledgments

We acknowledge the gracious support provided by the "Projects of Large Infrastructure for Research, Development, and Innovations" LM2010005 funded by the Ministry of Education, Youth, and Sports of the Czech Republic and the gracious support of the Grant Agency of the Czech Republic provided under the grant No. P202/12/0306.

References

[1] MetaCentrum, November 2013. http://www.metacentrum.cz/.
[2] D. Klusáček, H. Rudová and M. Jaroš. Multi Resource Fairness: Problems and Challenges. In *Job Scheduling Strategies for Parallel Processing*, 2013.