cesnet
metacentrum

How to manage fairness in a distributed computing system

Dalibor Klusáček
CESNET

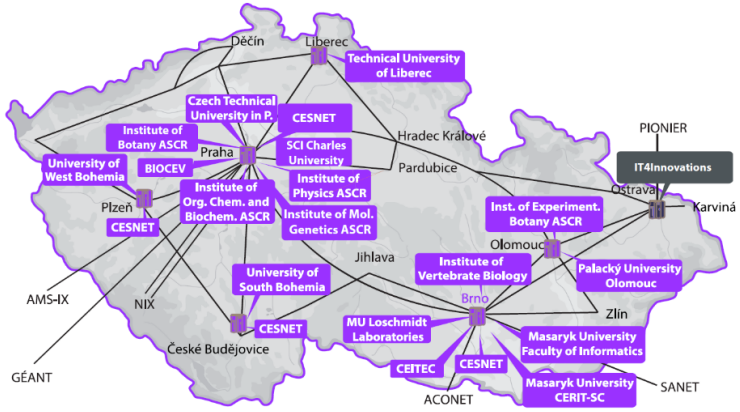
Sitola 2024, October 9, 2024

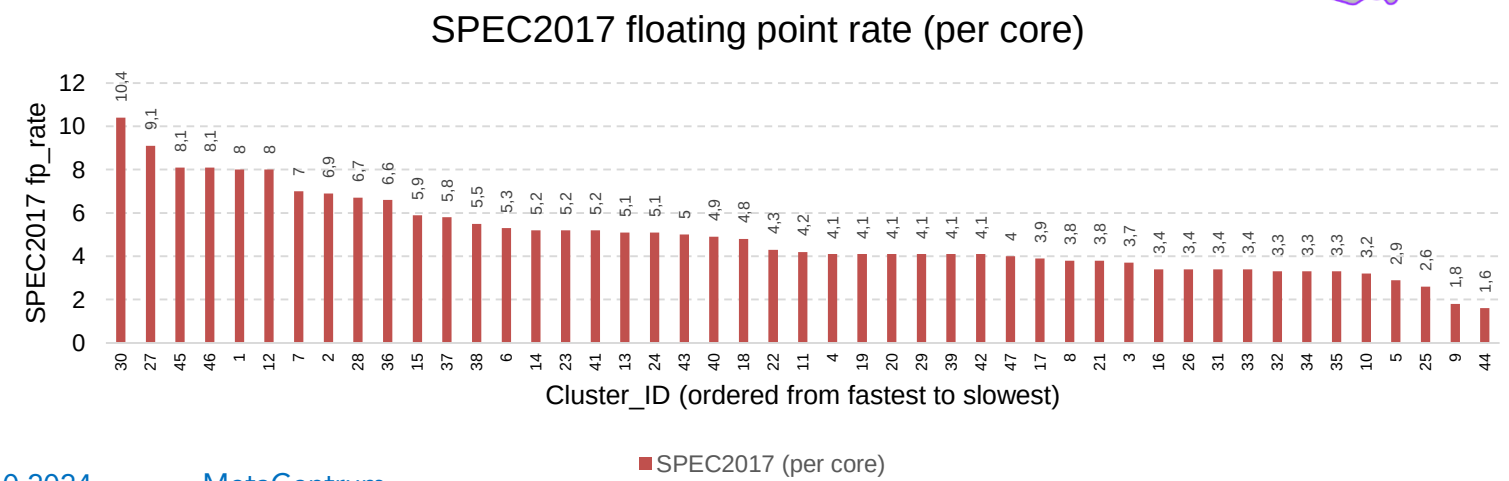
Outline

- **Work based on our experience when managing the scheduler in the Czech National MetaCentrum Infrastructure**
- **“Free of charge” computing environment**
  - Requires the use of policies that assure user-to-user fairness
- **Fair-sharing principles and challenges**
  - How it works in resource managers like Slurm or PBS
  - How it can be “tailored”
- **Fairshare simulator**
  - Used for analysis and tuning of the real system

Distributed & Heterogeneous Clusters

- **We operate clusters across all major cities**
  - Distributed & heterogeneous infrastructure
  - Different speeds/age
  - Some nodes are “popular” (new/fast)





Distributed & Heterogeneous Clusters

- **Resources provided for free**
  - There is no immediate motivation for the users to use the system efficiently
- **How to guarantee fair access for our users?**
  - Do nothing vs. have an actual POLICY (i.e., we know what we want to achieve)
- **Policies (few examples)**
  - Assign fixed resources (resource quotas)
  - Assign time slots (calendar)
  - Try something more flexible and more efficient
    - Try to understand if it works as intended
    - Be able to explain it to others

cesnet
metacentrum

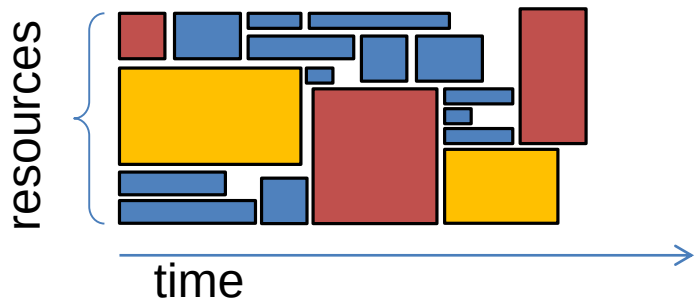
FAIR-SHARING

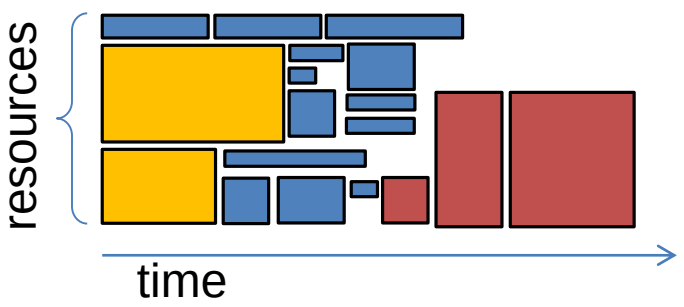
Fair-sharing (1)

Fair-sharing principles

- Prioritize users according to their (recent) resource usage
- Max-Min principle: more usage implies lower priority
  - User is waiting = priority ↑
  - User is computing = priority ↓
- Over long time period user's "share" is balanced with other active users

Example: Fair vs. Unfair scheduling





Fair-sharing (2)

- **Fair-sharing principles**
  - Prioritize users according to their (recent) resource usage
  - Max-Min principle: more usage implies lower priority
- **Implementation in resource managers is complicated**
  - **usage**(job) = **allocated_resources**(job) * **runtime**(job)
  - A lot of freedom how to measure the usage, e.g.:
    - Single vs. multiple resources are tracked (CPU, RAM, GPU, SSD, …)
  - Weights to reflect cost and/or age of hardware
    - Slow machine => longer runtime => higher fairshare penalty ☹
  - Decaying (to reflect the aging => focus on recent usage)
    - Periodically decrease the recorded usage of every user
    - Recent usage is more important than old one

Fair-sharing (3)

- **Fairshare prioritization has many parameters to set up**
  - The resulting formula to compute the user's priority (Fairshare Factor) becomes undecipherable for a common user

$$usage_{effective}(User_i) = \frac{usage(User_i)}{usage(total)}$$

$$usage_{target}(User_i) = \frac{shares(User_i)}{shares(total)}$$

$$FF(User_i) = 2^{-\left(\frac{usage_{effective}(User_i)}{usage_{target}(User_i)}\right)}$$

- **Further challenges**
  - It is hard to judge properly the impact of various parameters
  - There is no tool in PBS/Slurm to visualize the prioritization process
  - Only a basic CLI tool to print **current** priorities

cesnet
metacentrum

FAIR-SHARING SIMULATOR

Fair-sharing Simulator, Metrics

Simulator mimics the functionality of Slurm/PBS

- It uses existing workload log from an HPC system
- Replays the workload using its timestamps
- Thus it can reconstruct how users used the system over the time
- And it calculates and visualizes crucial metrics

Displays useful fair-sharing metrics

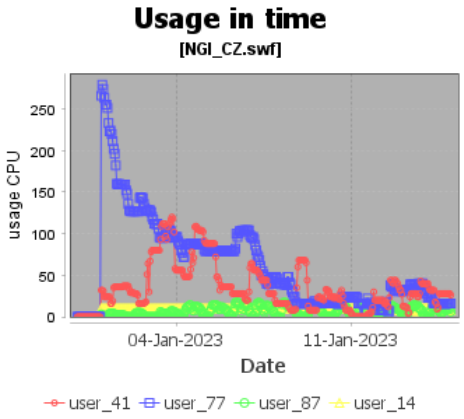
- Actual resource usage
- Total consumed resources (cumulative per user)
- **Fairhare Factor** (the actual relative user priority as if computed by Slurm/PBS)

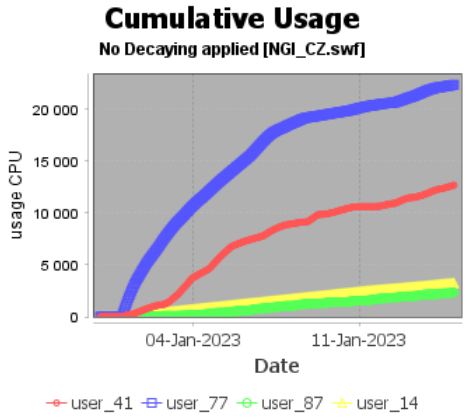
Allows us to test fairshare settings

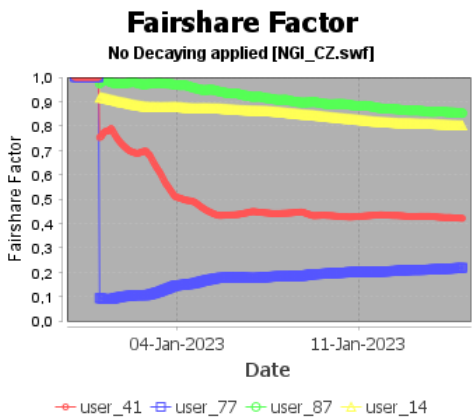
- Impact of measuring multiple resources within fairshare
- Impact of resource weighting (to reflect heterogeneity)
- Impact of decaying (how much and how often the usage history is “deleted”)

What Resources to Track?

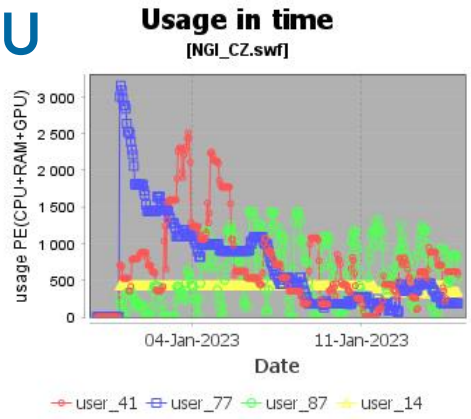
- CPU only

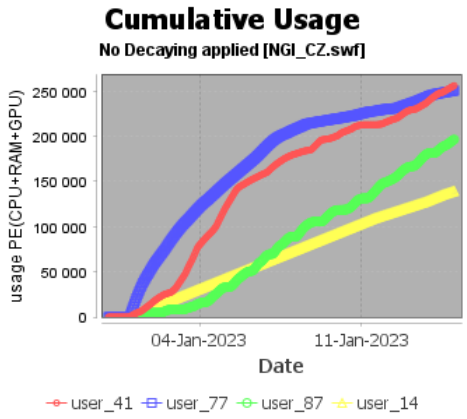


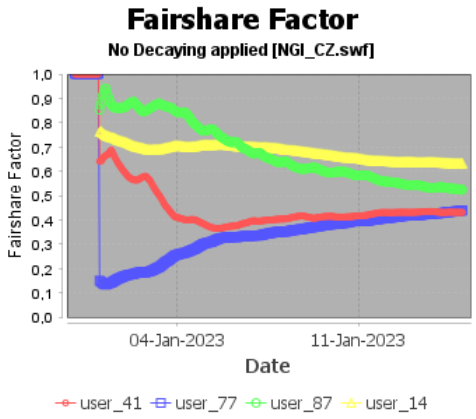




- CPU+RAM+GPU

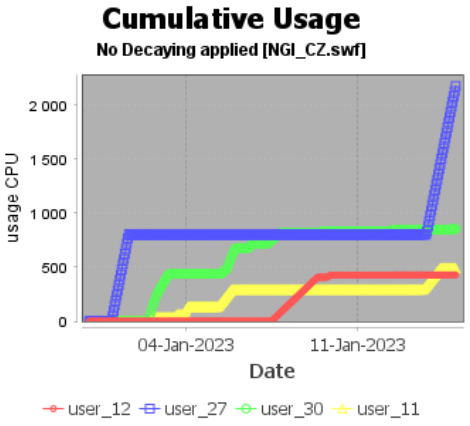


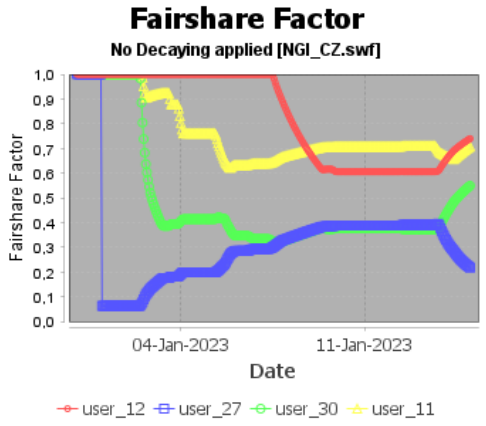




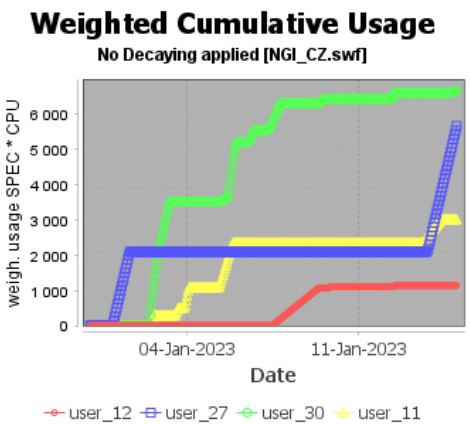
Weighting Resources (via their speed/cost/…)

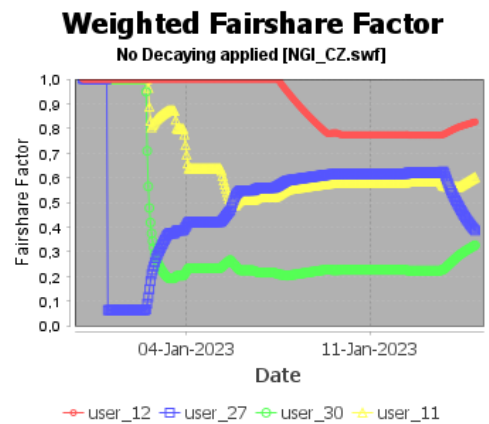
- **No weights ->**
  - All nodes/CPUs considered as equally "expensive"





- **Weights ->**
  - Based on SPEC 2017 speed benchmark

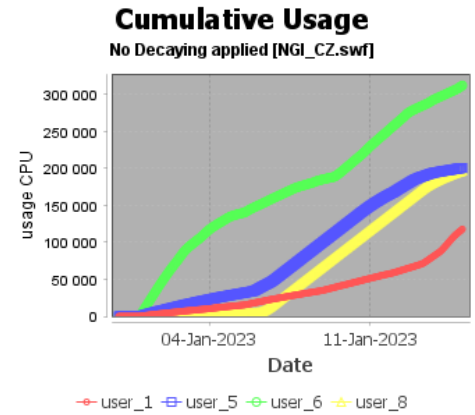


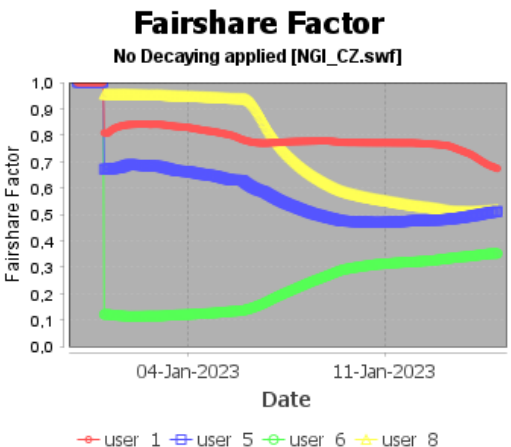


cesnet
metacentrum

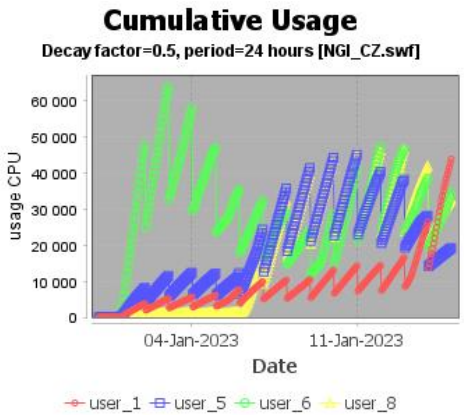
Decaying
(Putting More Weight on Recent Usage)

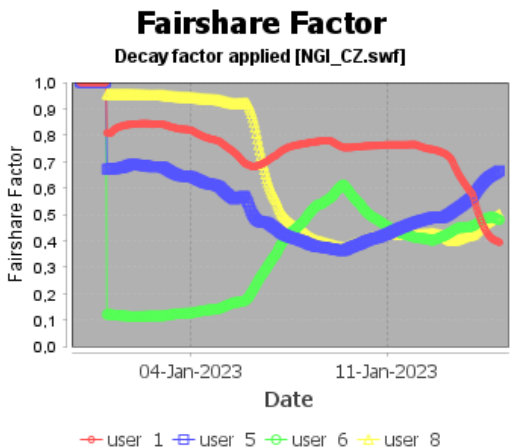
No decay



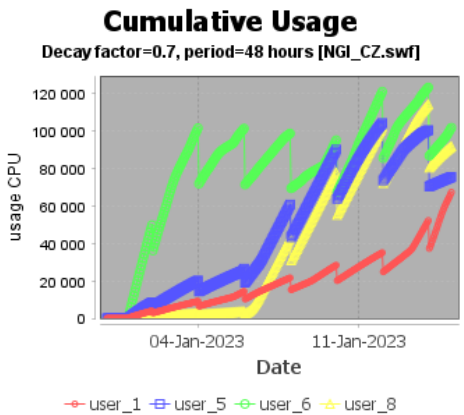


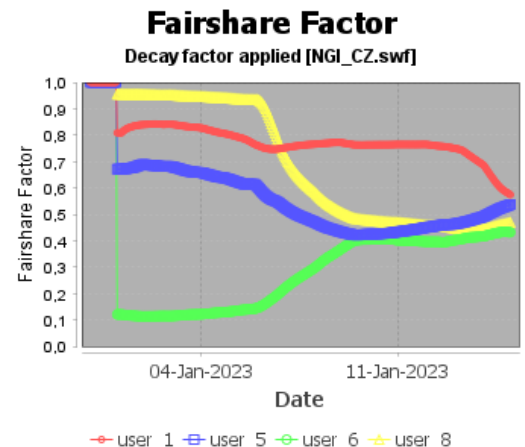
Aggressive Decay





Compromise



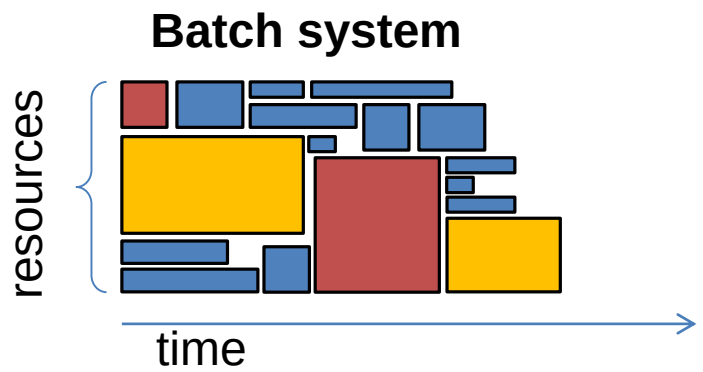


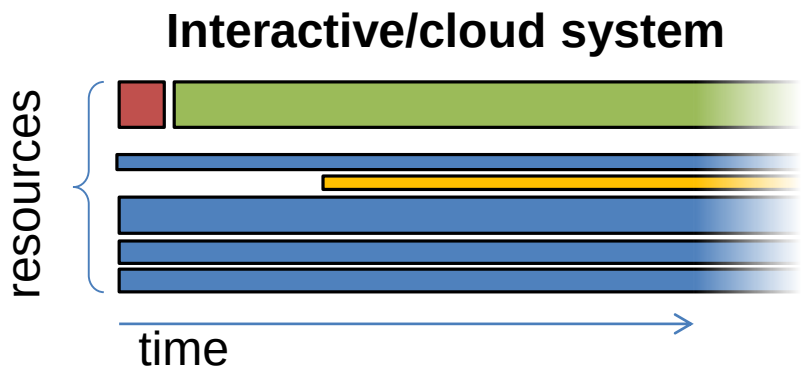
cesnet
metacentrum

TALK SUMMARY

Summary

- **The importance of fair-sharing in "free-of-charge" systems**
  - Works well for batch-oriented computations
  - Not so much for interactive/cloud apps
- **Fair-sharing is not suitable for long-running services**
  - They run "forever", especially in free-of-charge systems





Summary of Contributions

- **Fairshare simulator**
  - Very useful when tuning & explaining fairshare to users
  - Fast testing of various fairshare parameters
    - Decaying
    - Resource weights
    - Multi-resource usage accounting
- **Future work**
  - Addition of jobs scheduling capability
  - Allowing to simulate the impact on the performance
  - Building upon our long experience with job scheduling simulators
    - See our previous papers from PPAM 2007 and PPAM 2019


cesnet
metacentrum
THANK YOU!