The Extension of Torque Scheduler Allowing the Use of Planning and Optimization Algorithms in Grids

Václav Chlumský, Dalibor Klusáček, Miroslav Ruda

vchlumsky@cesnet.cz

CESNET z.s.p.o., Czech Republic

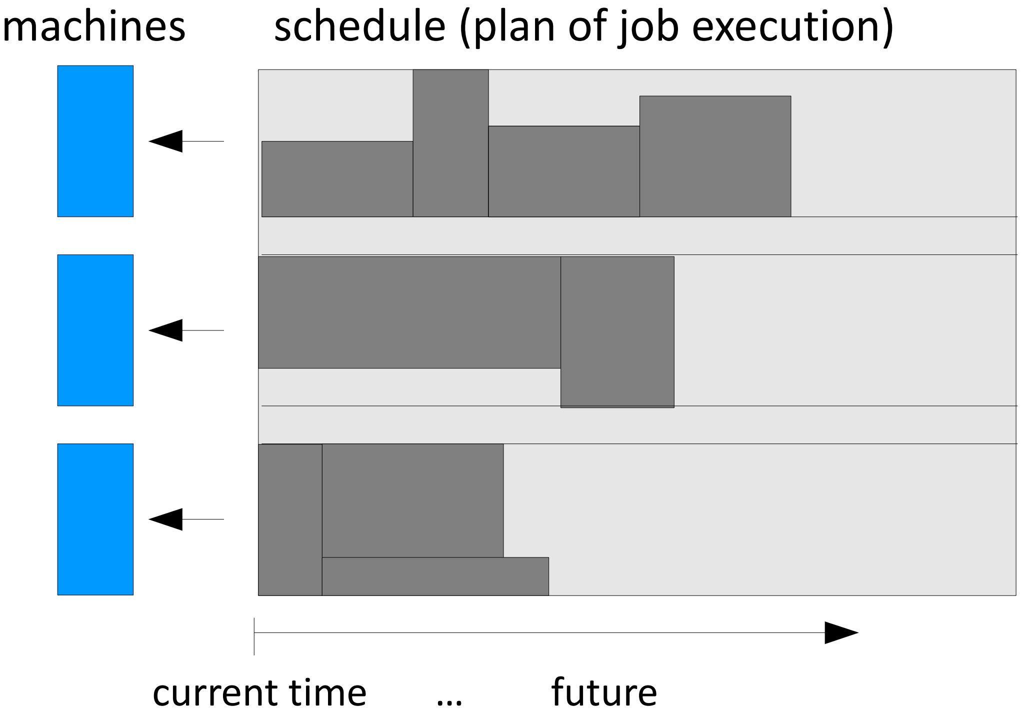
11th Cracow Grid Workshop | November 7 – 9, 2011

Motivation

- **Efficient job scheduling in Grids is difficult task**
  - large, heterogeneous and dynamic system
- **State of the art scheduling approaches**
  - queue-based
  - robust and tolerable to dynamic environment
  - ad hoc decisions (at the very last moment)
  - limited predictability, planning and "self evaluation"
  - hard to use them when several objectives (goals) are to be followed simultaneously

Schedule-based Solutions

- **Schedule represents plan of job execution**
  - straightforward planning
  - allows to predict behavior
  - easy **evaluation** wrt. selected optimization criteria
    - evaluation detects "problems"
  - **optimization of schedule**
    - to fix the problem
    - to improve the quality
    - using advanced scheduling techniques such as (meta)heuristics

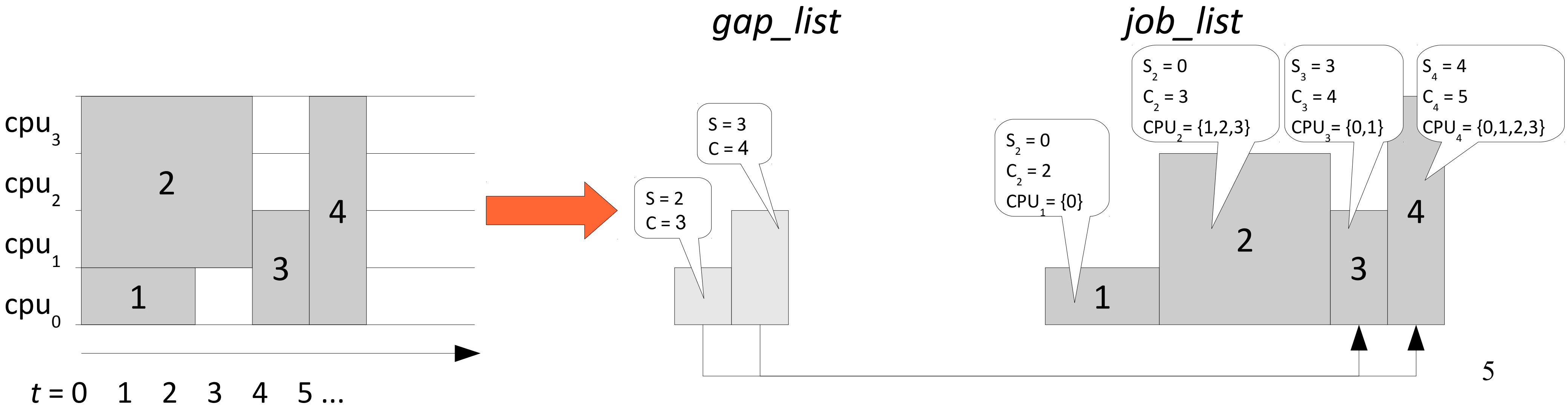


Main Contribution of Our Work

- **Implementation of schedule-based solution in Torque Resource Manager**
- Main features
  - schedule data structure in the `pbs_sched` module
    - planning and prediction
    - when and where jobs will be executed
  - the use of **optimization algorithms**
    - subject to selected optimization criteria
    - **local search-based methods**
    - periodically improving the quality of schedule

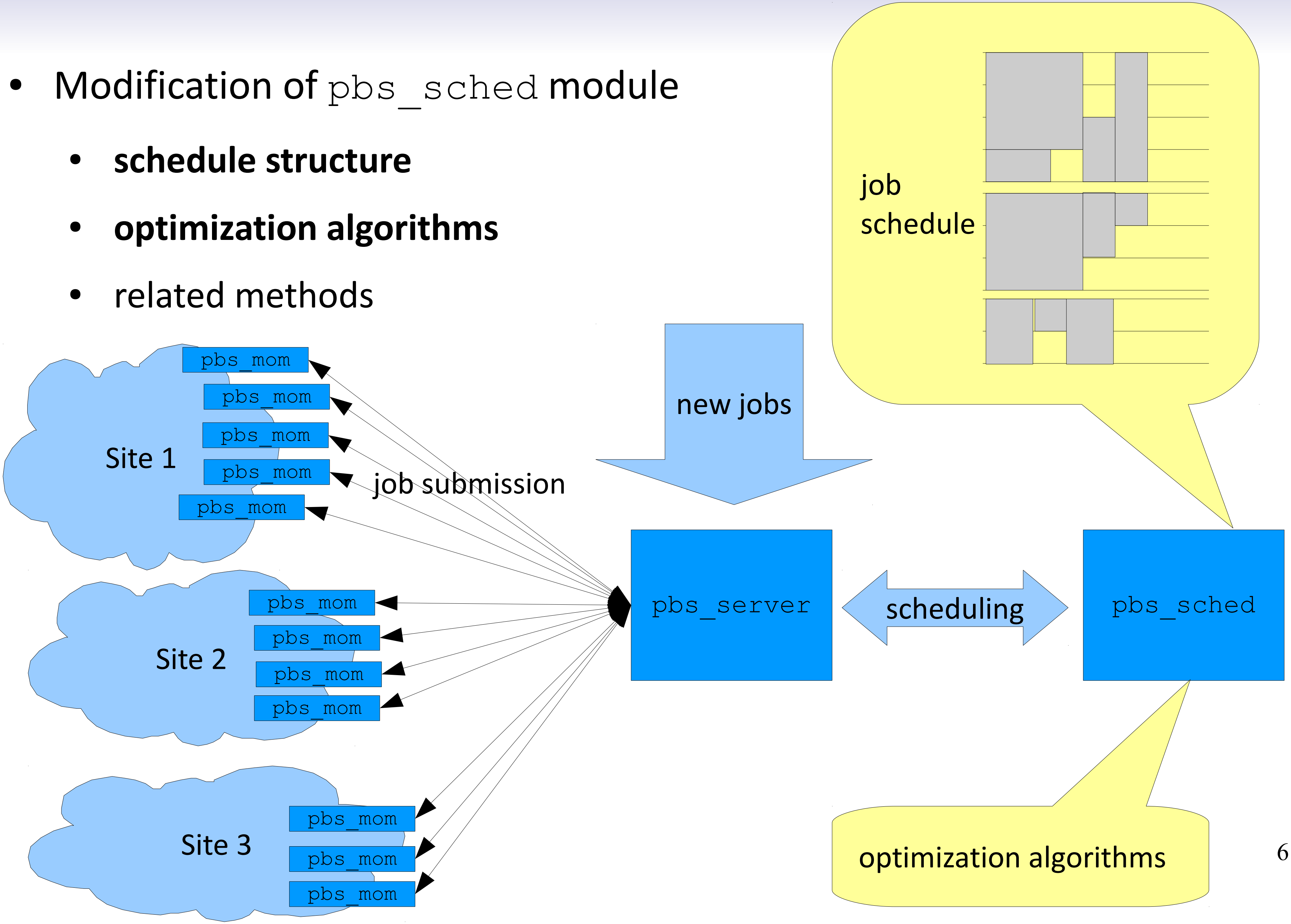
Schedule Representation

- *job_list*: a list of objects that represent job-to-machine mapping in time
- *gap_list*: a list of gaps that represent unused CPU time slots
    - speeds up common operations
        - – e.g., when trying to backfill some job
        - – only the gap list is traversed
        - – start times of previous jobs remain guaranteed



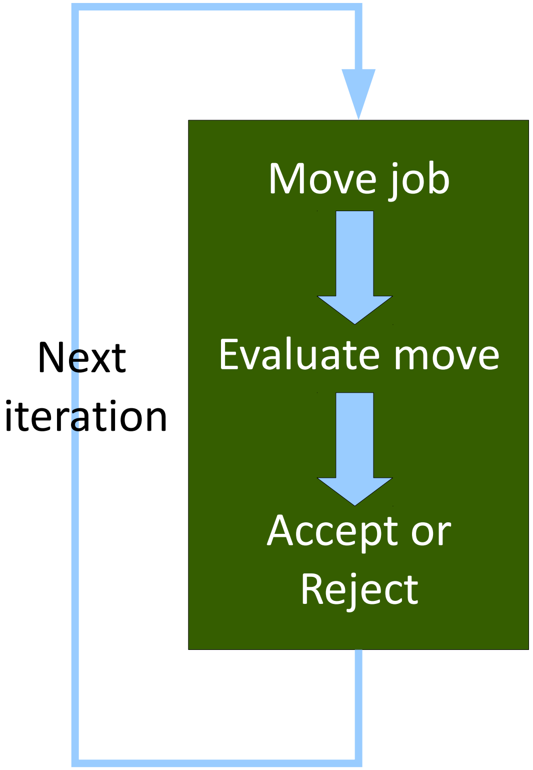
Torque Resource Manager

- Modification of `pbs_sched` module
  - **schedule structure**
  - **optimization algorithms**
  - related methods



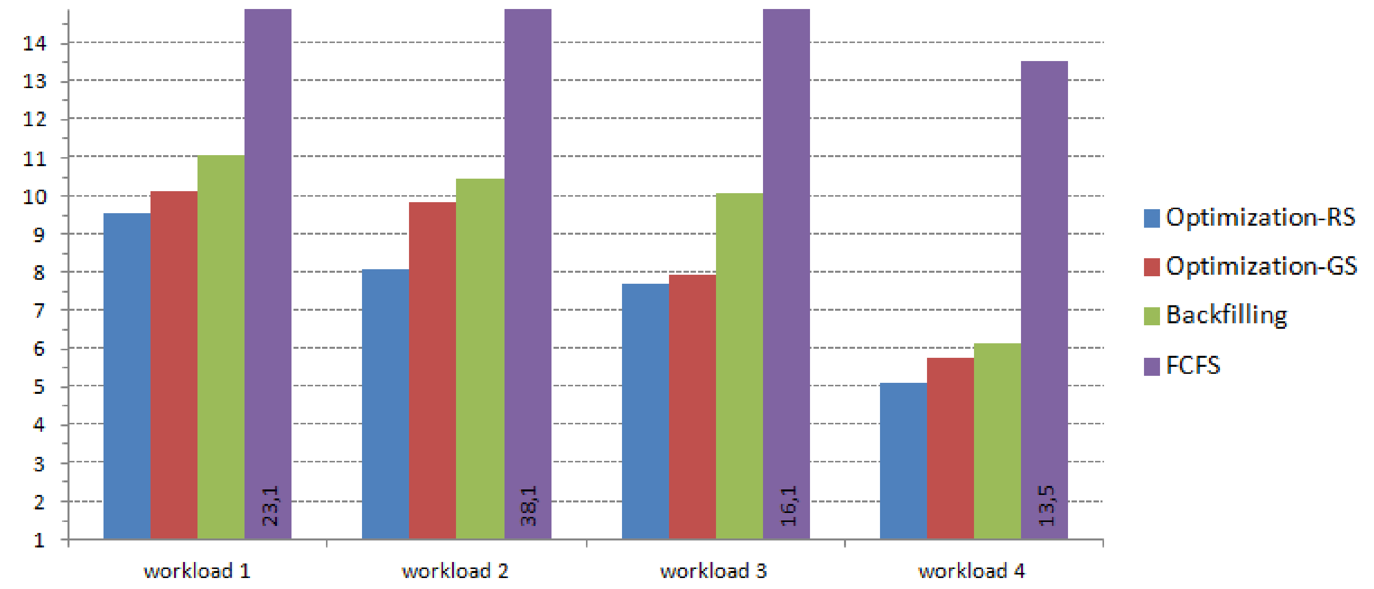
Preliminary Results

- 4 problem instances using real workloads from Czech NGI MetaCentrum
- **2 iterative Local Search-based optimization algorithms**
  - "conservative" gap-search algorithm
    - move random jobs into existing gaps
  - random-search algorithm
    - move random jobs into random positions
  - applied objective – **minimize avg. slowdown**
  - measured criteria
    - avg. slowdown
    - avg. response time
    - avg. wait time
- **Compared wrt. FCFS and Backfilling**

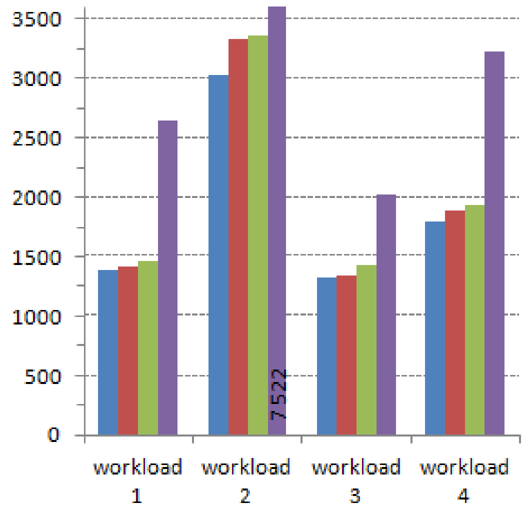


Preliminary Results

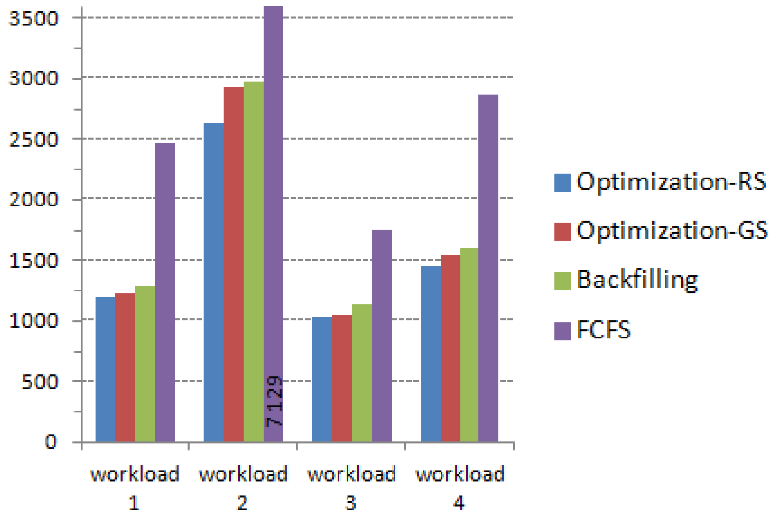
slowdown



response time [s]



wait time [s]



Current and Future Work

- Further testing
  - implementation of multi-criteria objective function
- Fairness related problems
  - *job-to-job* and *user-to-user* fairness
  - proposal and integration of proper objective functions
- Deployment of such a solution in a production environment
- The use of runtime-prediction techniques
  - when estimates are very inaccurate/missing
- Preparation of a publicly available software package
  - Torque with the schedule-based extension of `pbs_sched` module