News on Scheduling Research in Delft (Holland) and Eindhoven (the Netherlands)

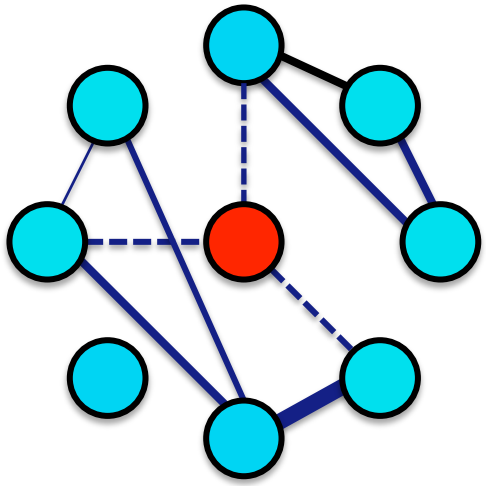
Dick Epema

with Bogdan Ghit, Alex Iosup, and Alexy Ilyushkin (TUD)
and Aleksandra Kuzmanovska (TU/e)

Parallel and Distributed Systems Group
Delft University of Technology
Delft, the Netherlands

and

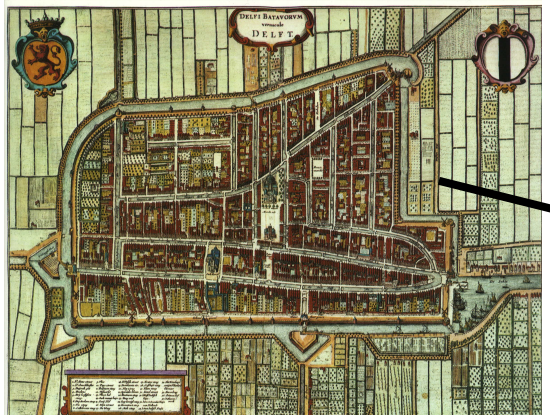
System Architecture and Networking Group
Eindhoven University of Technology
Eindhoven, the Netherlands



12 March 2015

TU Delft
Delft University of Technology

Delft – the Netherlands – Europe

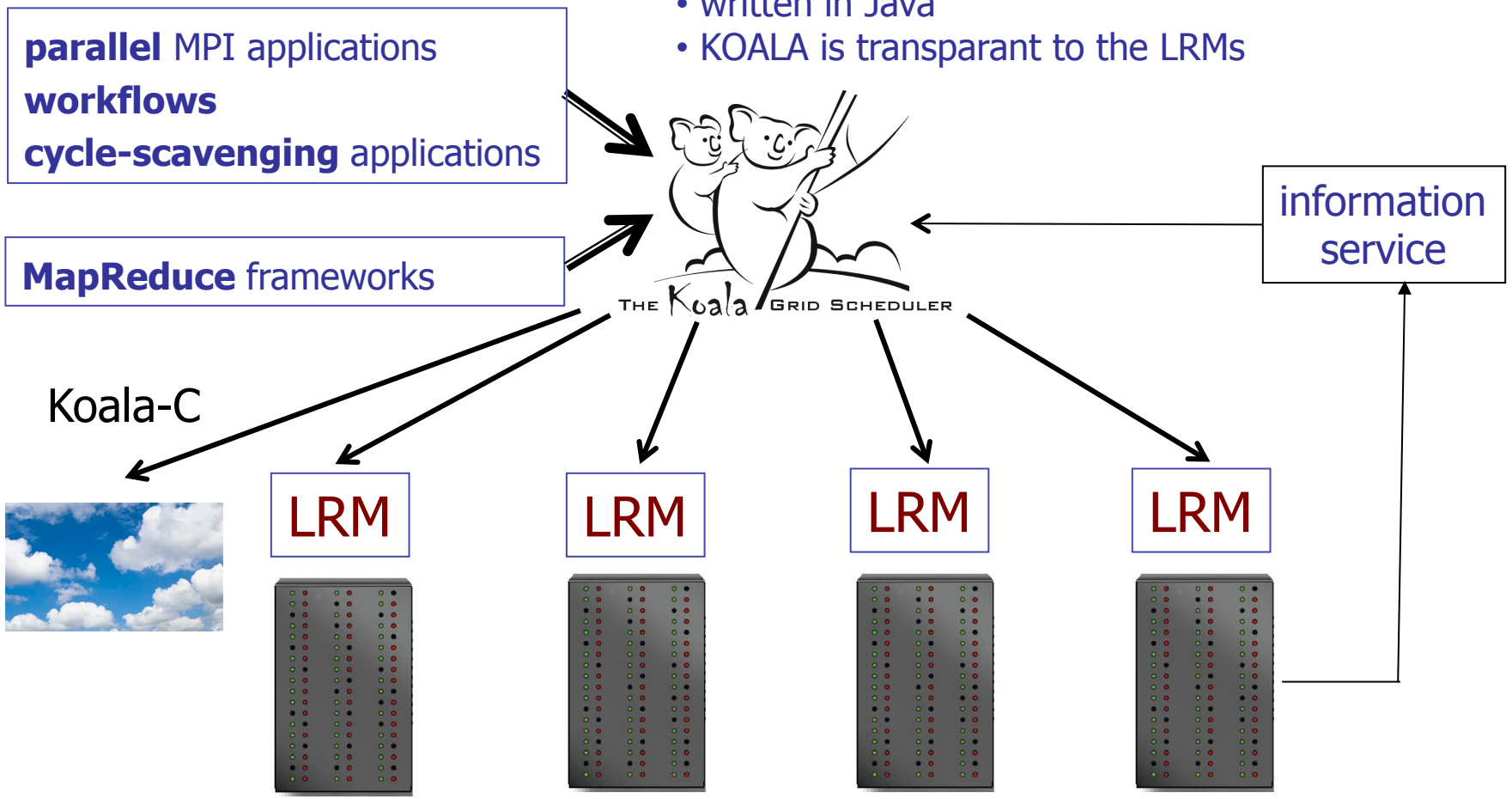


Holland
(2/12 provinces)

TU Delft

The KOALA multicluster scheduler

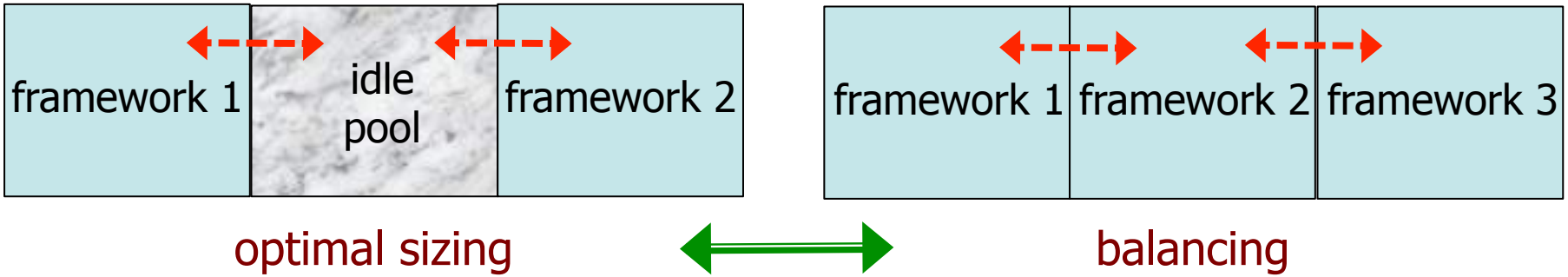
- KOALA is our research vehicle for scheduling research
- deployed since 2005
- written in Java
- KOALA is transparant to the LRMs



TU Delft

Scheduling frameworks

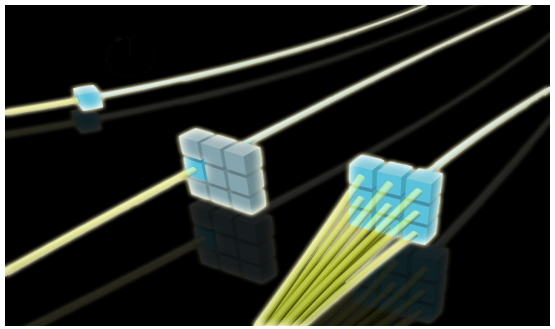
- **Reduce**
  - **scheduling overhead** of centralized scheduler
  - **complexity** of centralized scheduler
- **Provide isolation among frameworks**
- **KOALA**
  - requests large chunk of a cluster and
  - allocates dynamic parts of it to frameworks
- **Two models:**



TU Delft

Types of Isolation

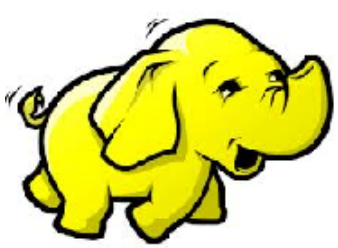
Performance isolation



Failure isolation

Data isolation

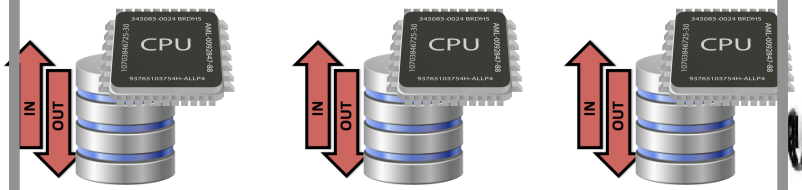
Version isolation



TU Delft

Resizing MapReduce: no data locality

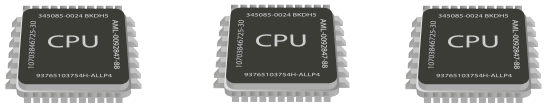
Core nodes



INPUT/OUTPUT DATA

- Classical deployment
- Uniform data distribution
- **No removal**

Transient nodes (TR)



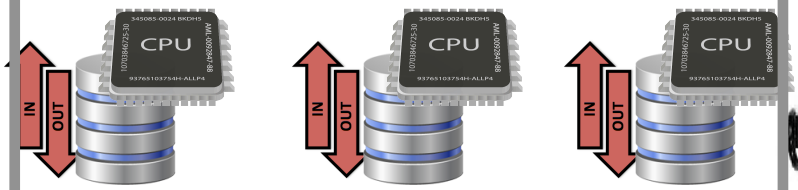
NO DATA

- No local storage
- R/W from/to core nodes
- **Instant removal**

TU Delft

Resizing MapReduce: relaxed data locality

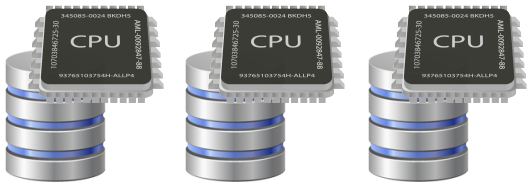
Core nodes



INPUT/OUTPUT DATA

- Classical deployment
- Uniform data distribution
- **No removal**

Trans-core nodes (TC)

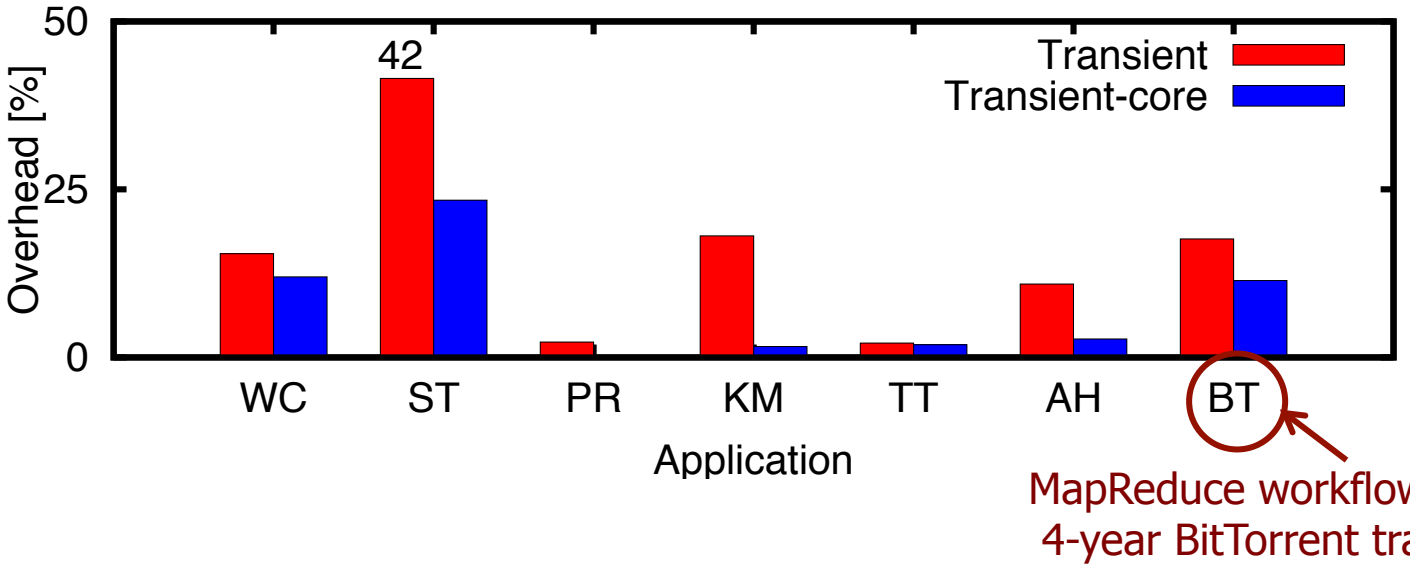


OUTPUT DATA

- Local storage, no input
- Only R from core nodes
- **Delayed removal**

TU Delft

Performance of no versus relaxed data locality



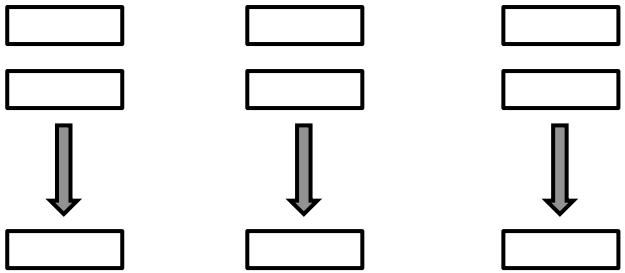
- single-application performance decrease
- base line: 20 nodes with full HDFS deployment
- 10 core nodes + 10 transient/transient-core nodes

B.I. Ghit, M. Capota, T. Hegeman, J. Hidders, D.H.J. Epema and I. Iosup, "V for Vicissitude: The Challenge of Scaling Complex Big-Data Workflows," **winner SCALE Challenge** at *CCGrid 2014*

TU Delft

Balancing Allocations with FAWKES

Two-level scheduling architecture

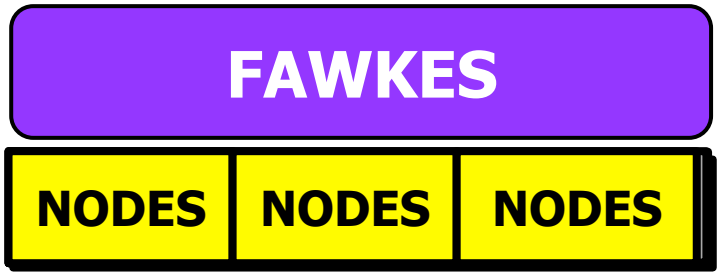


Job submissions

Frameworks

Resource manager

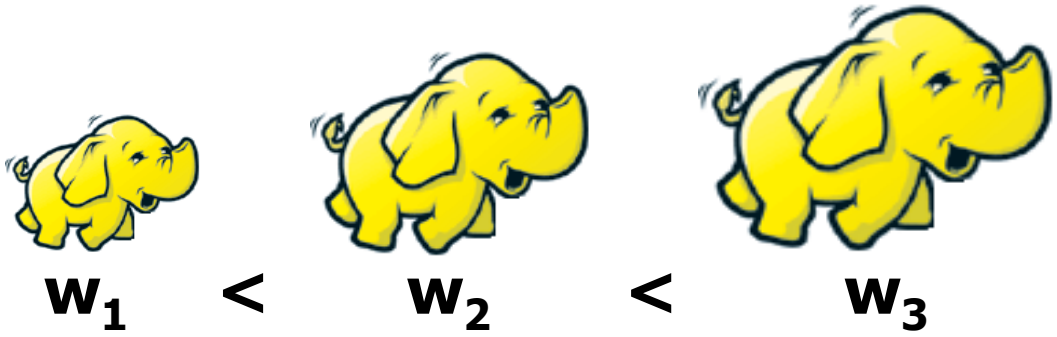
Infrastructure



B.I. Ghit, A. Iosup, and D.H.J. Epema, "Balanced Resource Allocations across Multiple Dynamic MapReduce Clusters," *ACM Sigmetrics 2014.*

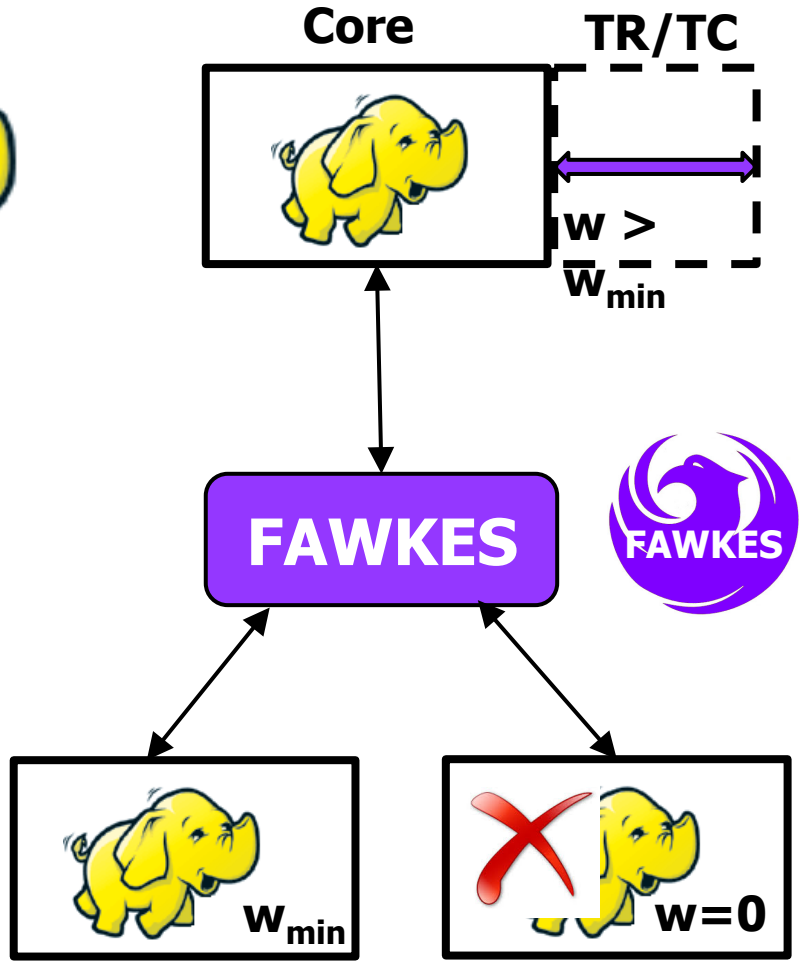
TU Delft

FAWKES in a nutshell



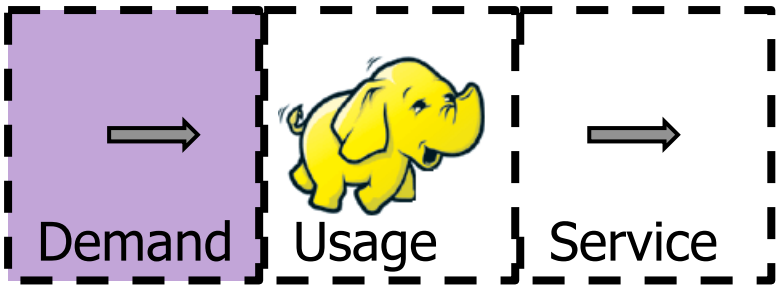
1. Updates dynamic weights when:
   - new frameworks arrive
   - framework states change

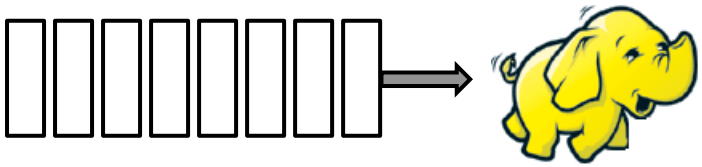
2. Shrinks and grows frameworks to:
   - allocate new frameworks (min. shares)
   - give fair shares to existing ones



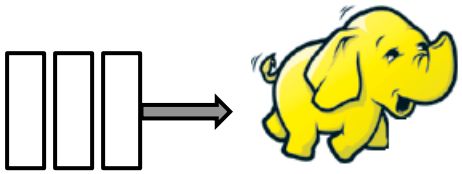
TU Delft

How to differentiate frameworks? (1/3)





versus

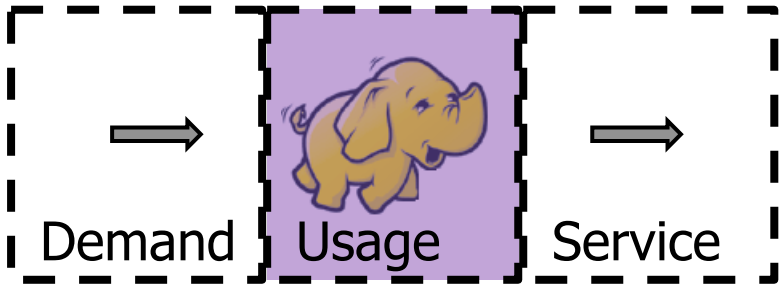


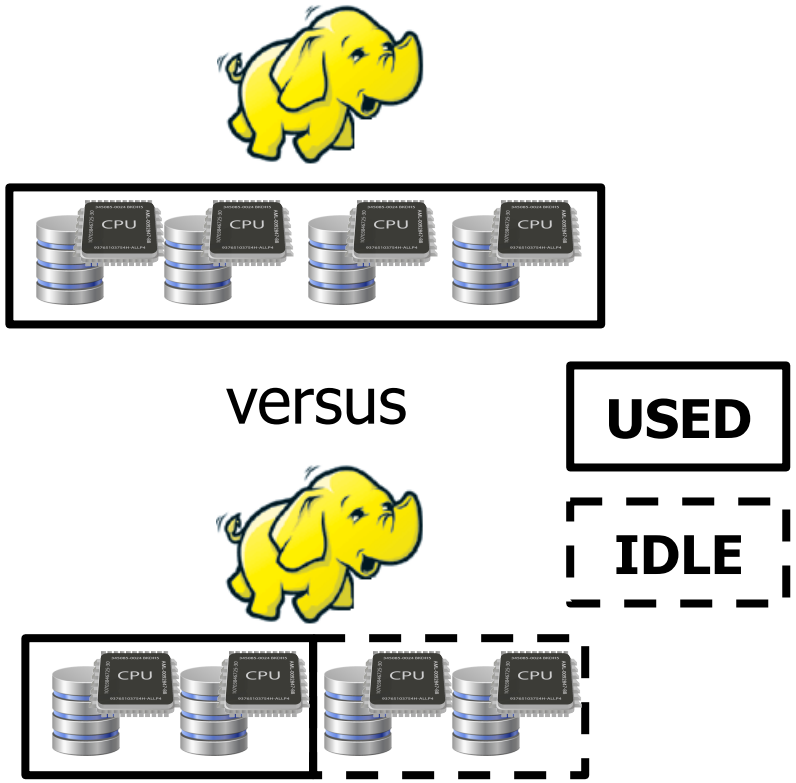
By **demand** – 3 policies:

- Job Demand (JD)
- Data Demand (DD)
- Task Demand (TD)

TU Delft

How to differentiate frameworks? (2/3)



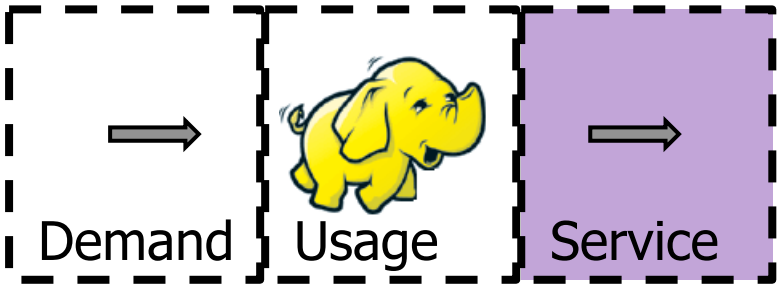


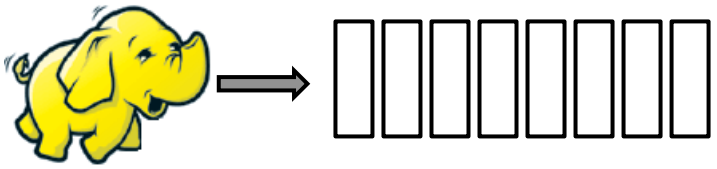
By **usage** – 3 policies:

- Processor Usage (PU)
- Disk Usage (DU)
- Resource Usage (RU)

TU Delft

How to differentiate frameworks? (3/3)

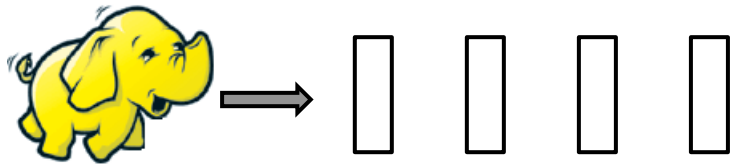




versus

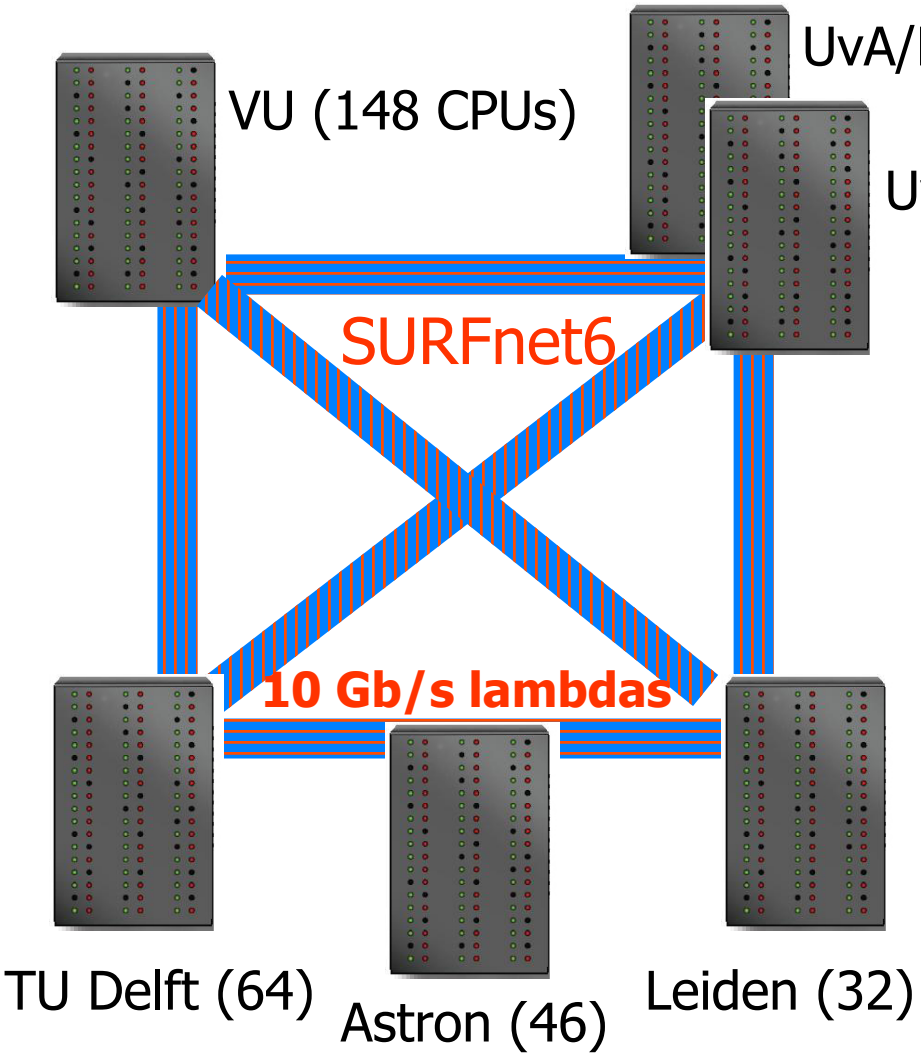
By **service** – 3 policies:

- Job Slowdown (JS)
- Job Throughput (JT)
- Task Throughput (TT)



TU Delft

Our experimental testbed: DAS-4



DAS5 on the way

- Q2 2015
- 400 8-core CPUs
- FDR Infiniband

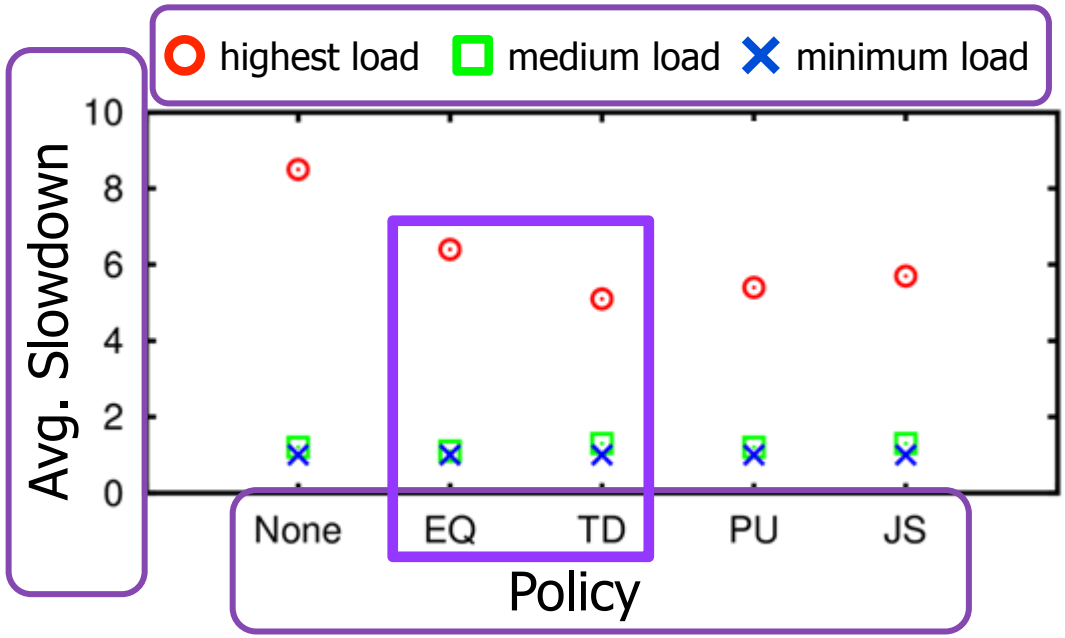
Advanced School for Computing and Imaging

Netherlands Organisation for Scientific Research

TU Delft

Performance of FAWKES

Nodes	45
Frameworks	3
Minimum shares	10
Datasets	300 GB
Jobs submitted	900



Up to 20% lower slowdown

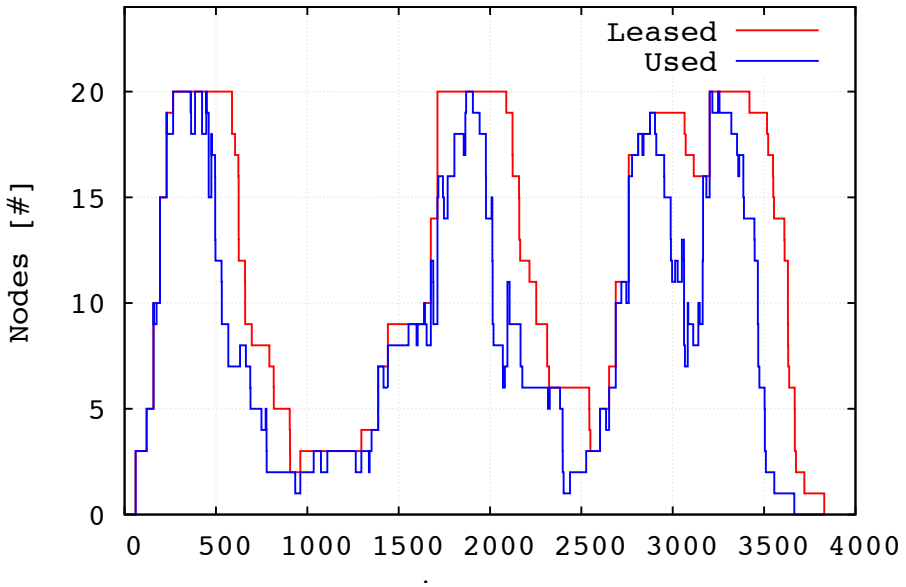
None – Minimum shares
EQ – Equal shares
TD – Task Demand
PU – Processor Usage
JS – Job Slowdown

TU Delft

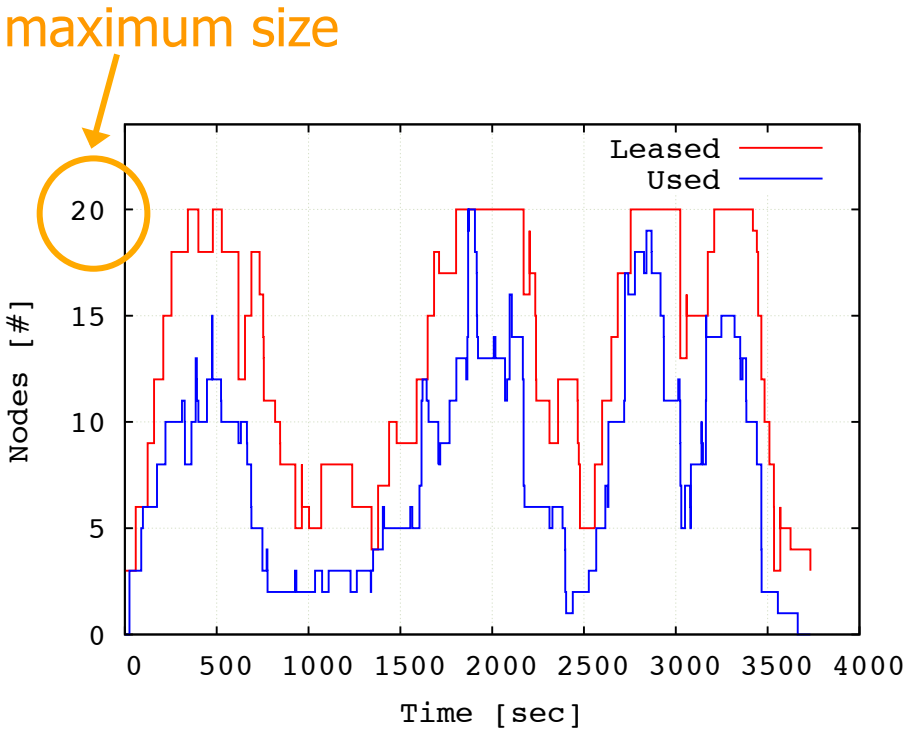
Optimal sizing (1)

- **Fluent** is a component-based framework
  - jobs consist of **batches of identical video applications** with identical runtimes
  - **admission control**: jobs require immediate/fast start
  - metric: **reject rate** (of all applications across all jobs)
- **OnDemand** policy:
  - framework **initiative**
  - explicit grow and shrink requests to KOALA
  - **grow** because of new job that doesn't fit
  - **shrink** after some idle time of resources
- **Proactive** policy:
  - KOALA **initiative**
  - **maintain utilization** (used/allocated) between lower and upper bound (periodic check)

Optimal sizing (2)



OnDemand

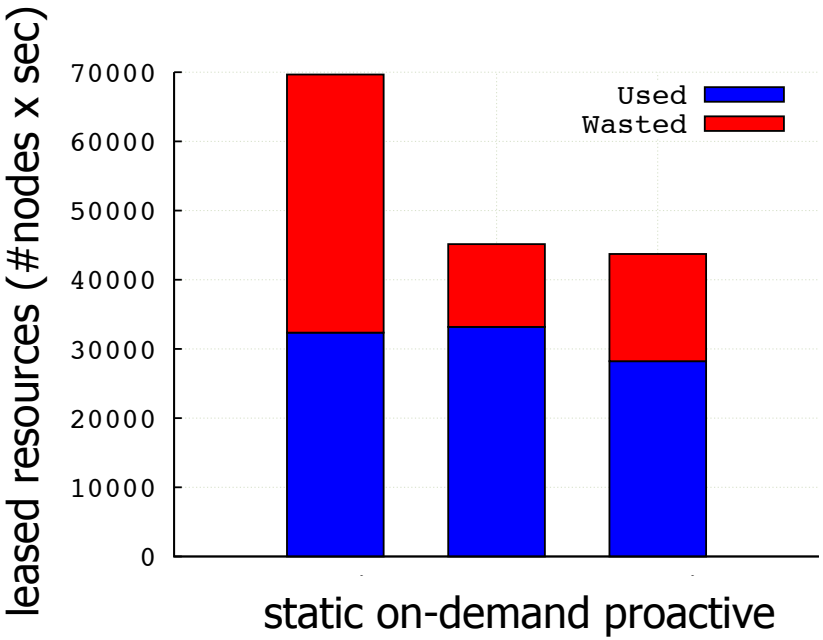


Proactive (util. 40-50%)

A. Kuzmanovska, R.H. Mak, and D.H.J. Epema, "Scheduling Workloads of Workflows with Unknown Task Runtimes," *Workshop Job Scheduling Strategies for Parallel Processing*, May 2014

TU Delft

Optimal sizing (3)

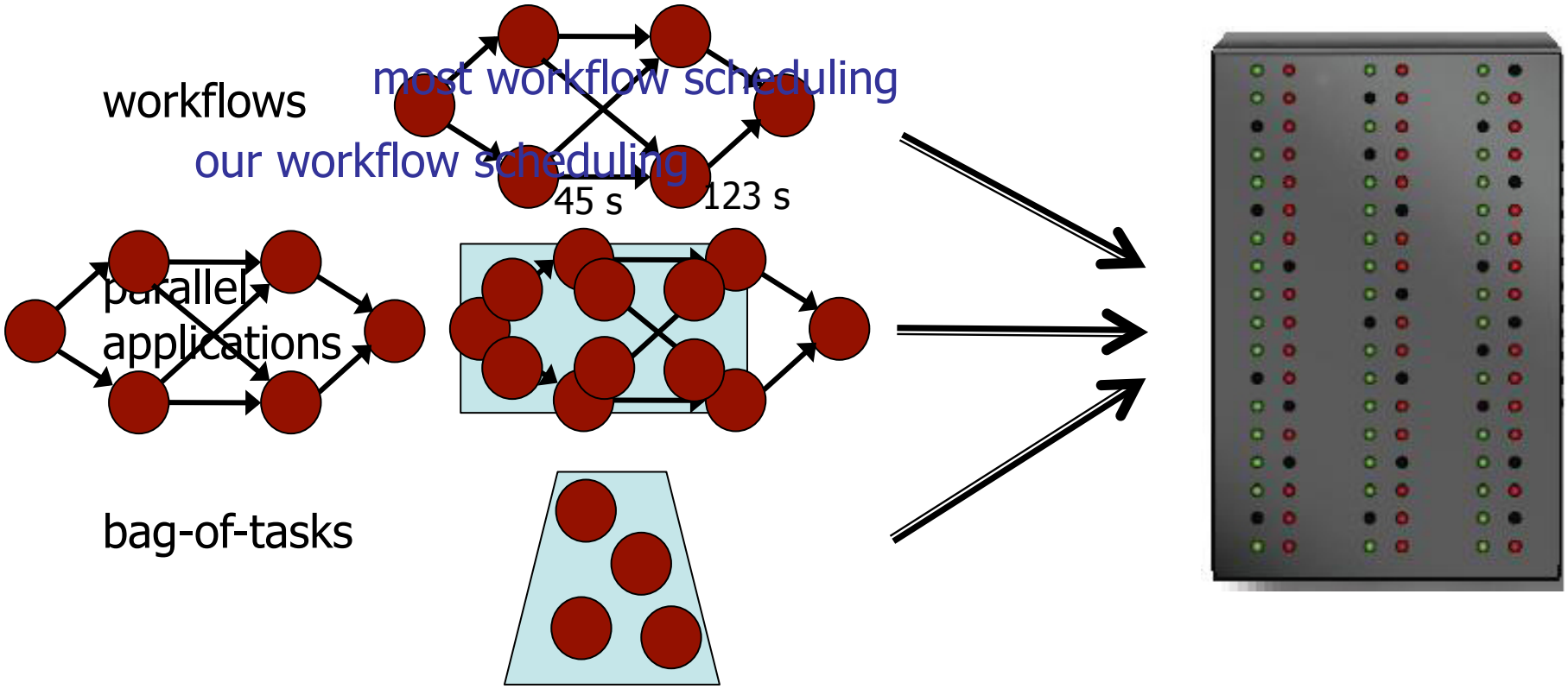


policy	reject rate (%)	utilization (%)
static	13	46
on-demand	13	73
pro-active	21	65

TU Delft

Workflow scheduling (1)

real workloads

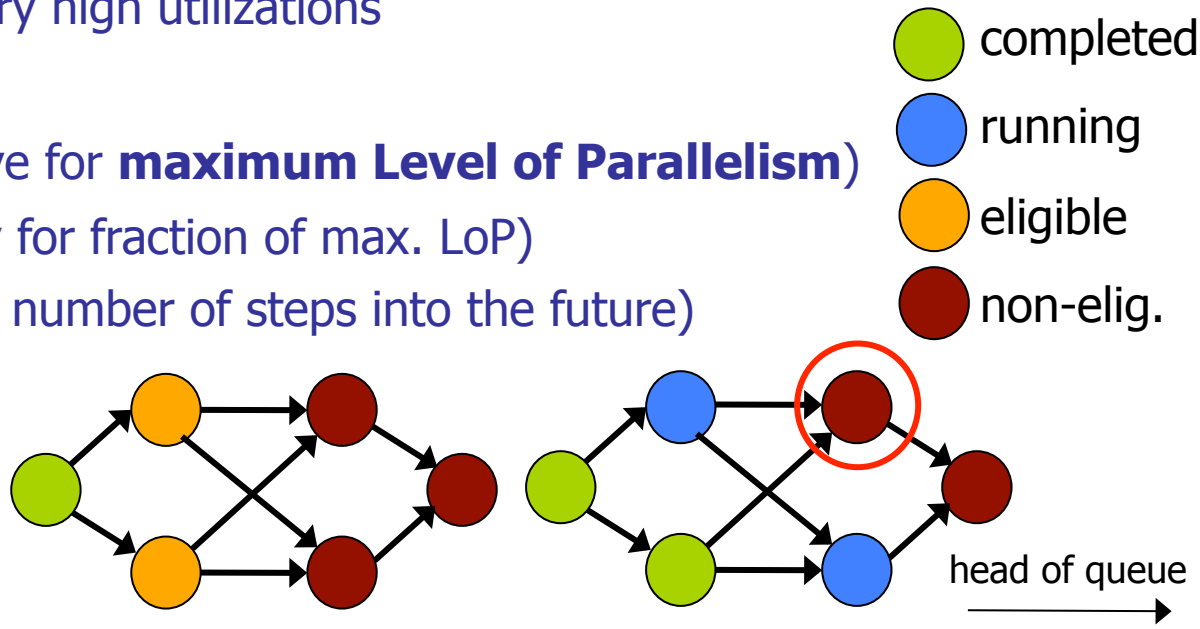


A. Ilyushkin, B.I. Ghit, and D.H.J. Epema, "Scheduling Workloads of Workflows with Unknown Task Runtimes," *15th IEEE/ACM Int'l Symposium on Cluster Computing and the Grid (CCGRID15)*, May 2015

TU Delft

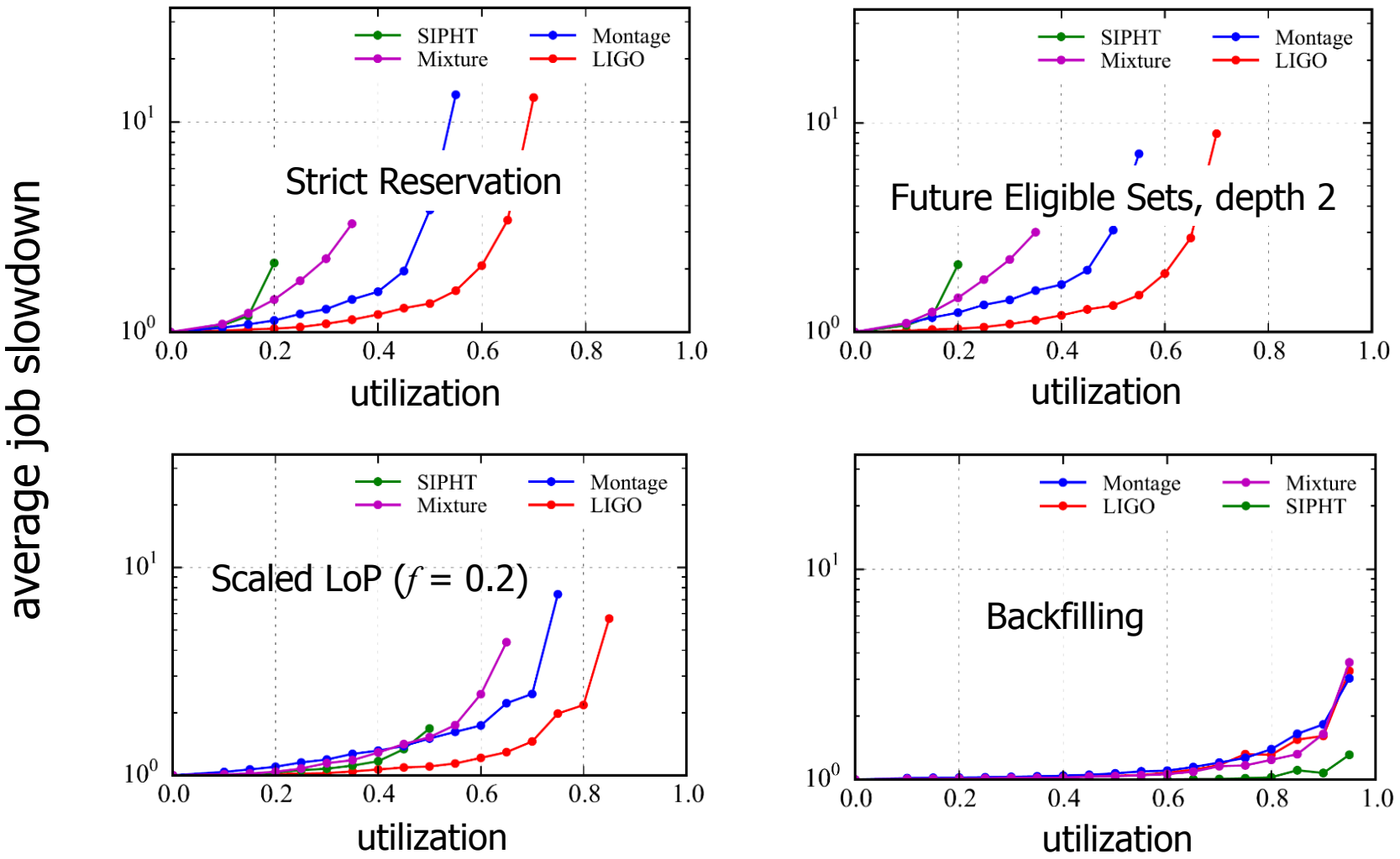
Workload scheduling (2)

- **Research question**
  - how to schedule **workloads of workflows** with **unknown** task runtimes?
- **Reserving some processors for job(s) at the head of the queue**
  - **reduces** time in service
  - but **increases** wait time
  - is clearly not good at very high utilizations
- **Policies**
  - strict reservation (reserve for **maximum Level of Parallelism**)
  - scaled LoP (reserve only for fraction of max. LoP)
  - future eligible sets (look number of steps into the future)
  - (unrestricted) backfilling
- **Metric**
  - job slowdown



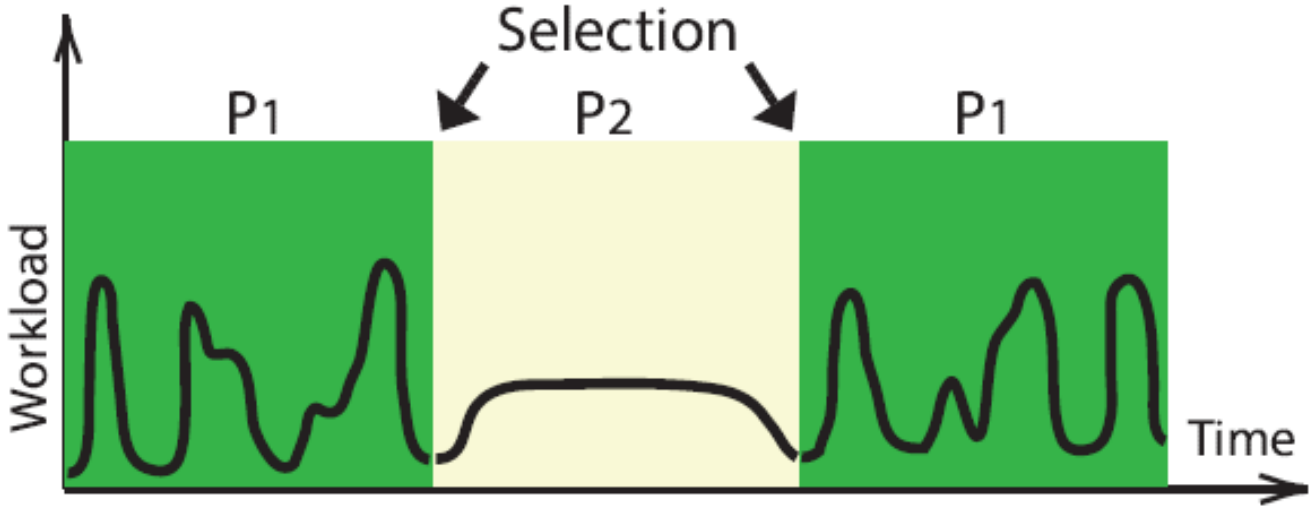
TU Delft

Workload scheduling (3)



TU Delft

Portfolio scheduling



- Create a set of scheduling policies
  - resource provisioning and allocation policies
- Online selection of the active policy, at important moments
  - periodic selection
  - change in pricing model
  - change in datacenter architecture

K. Deng, J. Song, K. Ren, and A. Iosup, "Exploring Portfolio Scheduling for Long-term Execution of Scientific Workloads in IaaS Clouds," *SuperComputing* 2013

TU Delft

Next March in Delft

- Home
- Important Dates
- Call For Contributions
  - TXT version
  - PDF version

Welcome to the 7th ACM/SPEC International Conference on Performance Engineering

The International Conference on Performance Engineering (ICPE) provides a forum for the integration of theory and practice in the field of performance engineering. ICPE is an annual joint meeting that has grown out of the ACM Workshop on Software Performance (WOSP) and the SPEC International Performance Engineering Workshop (SIPEW). It brings together researchers and industry practitioners to share ideas, discuss challenges, and present results of both work-in-progress and state-of-the-art research on performance engineering of software and systems.

Sponsors

TU Delft

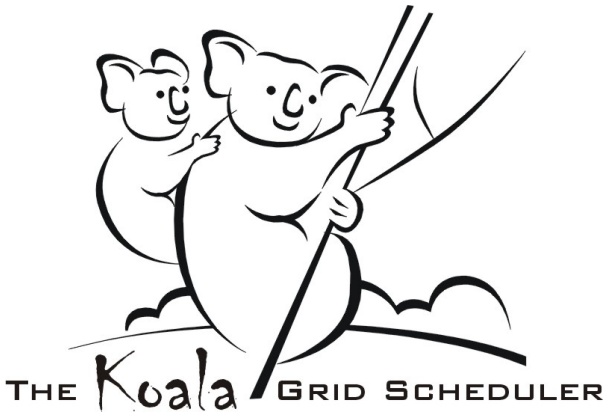
Delft University of Technology

General Chair: Alex Iosup

TU Delft

More information

- **Publications**
  - see PDS publication database at publications.st.ewi.tudelft.nl
- **Home pages**:
  - www.pds.ewi.tudelft.nl/epema
  - www.pds.ewi.tudelft.nl/~iosup
- **Web sites:**
  - KOALA: www.st.ewi.tudelft.nl/koala
  - DAS4: www.cs.vu.nl/das4



TU Delft

Our research tag cloud

TU Delft