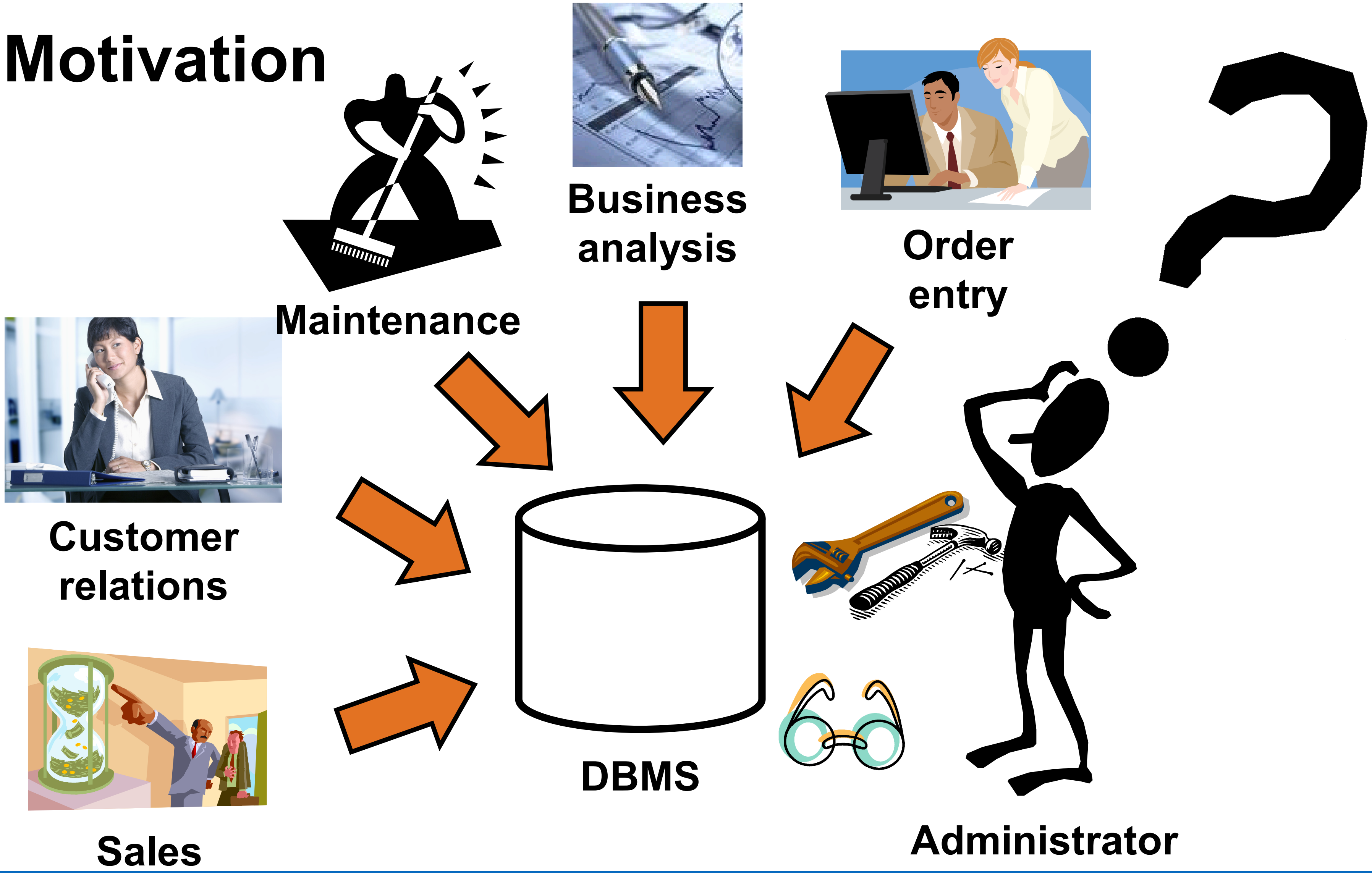
Managing Long-Running Queries

Stefan Krompass[TUM], Harumi Kuno[HPL], Janet Wiener[HPL], Kevin Wilkinson[HPL], Umeshwar Dayal[HPL], and Alfons Kemper[TUM]

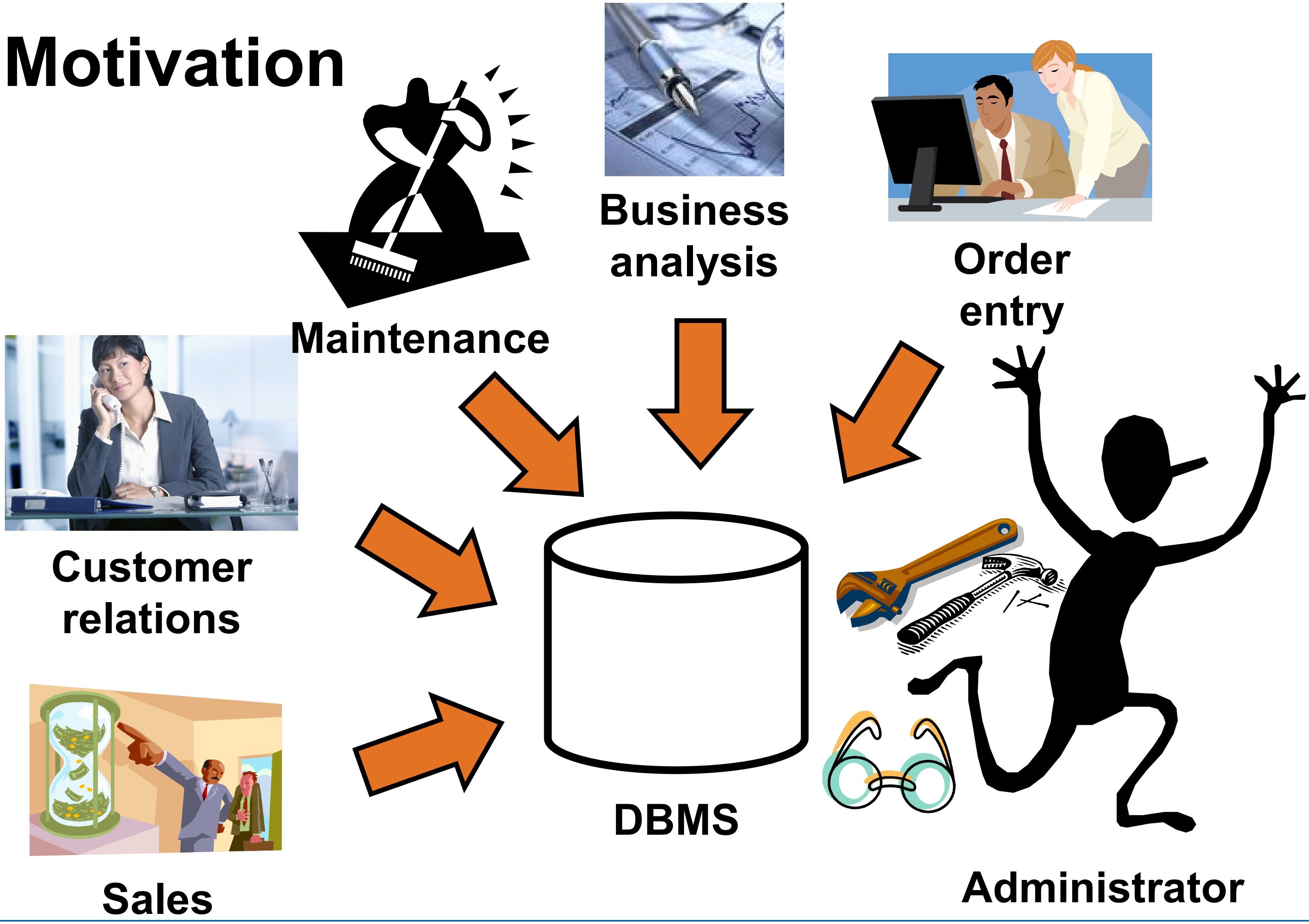
[TUM]Technische Universität München
Munich, Germany

[HPL]Hewlett-Packard Laboratories
Palo Alto, CA, USA

Motivation


Maintenance
Business analysis
Order entry
Customer relations
Sales
DBMS
Administrator

Motivation



Maintenance

Business analysis

Order entry

Customer relations

Sales

DBMS

Administrator

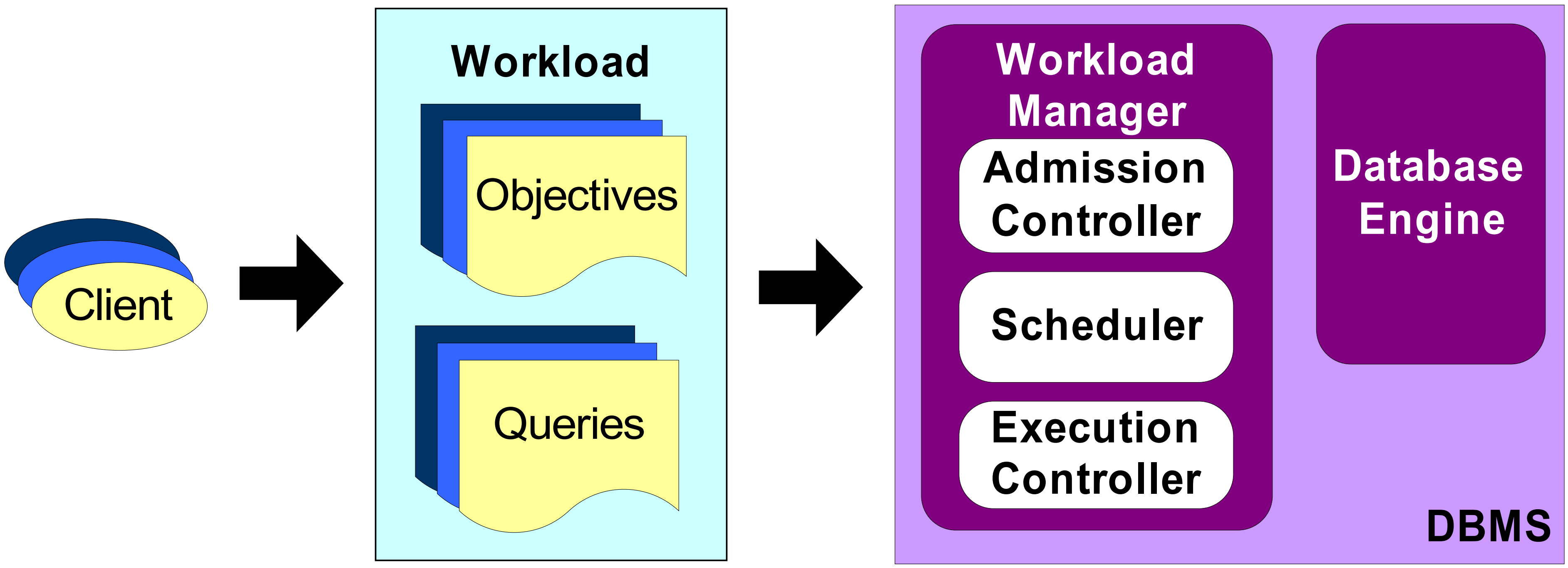
Goals of this work

- Develop technology to study policies for mixed workloads
- Initial study of managing mixed workloads, in particular: impact of long-running queries on a workload
  - Unreliable cost estimates
    under-informed admission control and scheduling decisions
  - Unobserved resource contention
    monitored resource not the source of contention
  - System overload

Outline

- Workload management components & workload management policies
- Experiments
- Conclusions

Workload management overview



Experimental approach

- Create workloads that inject “problem” queries (our workloads are derived from actual mixed workload queries)
- Develop a workload management software that implements admission control, scheduling, and execution control policies
- Workload manager feeds queries into database engine **simulator**
  - Investigate workloads that run for hours
  - Obtain reproducible results
  - Experiment with comprehensive set of workload management policies
  - Inject problem queries

Experimental input: queries

query type	size of query pool	queries per workload	average elapsed time
short	2807	400	30 sec
medium	247	23	10 min
long	48	3	1 hr

Problem queries

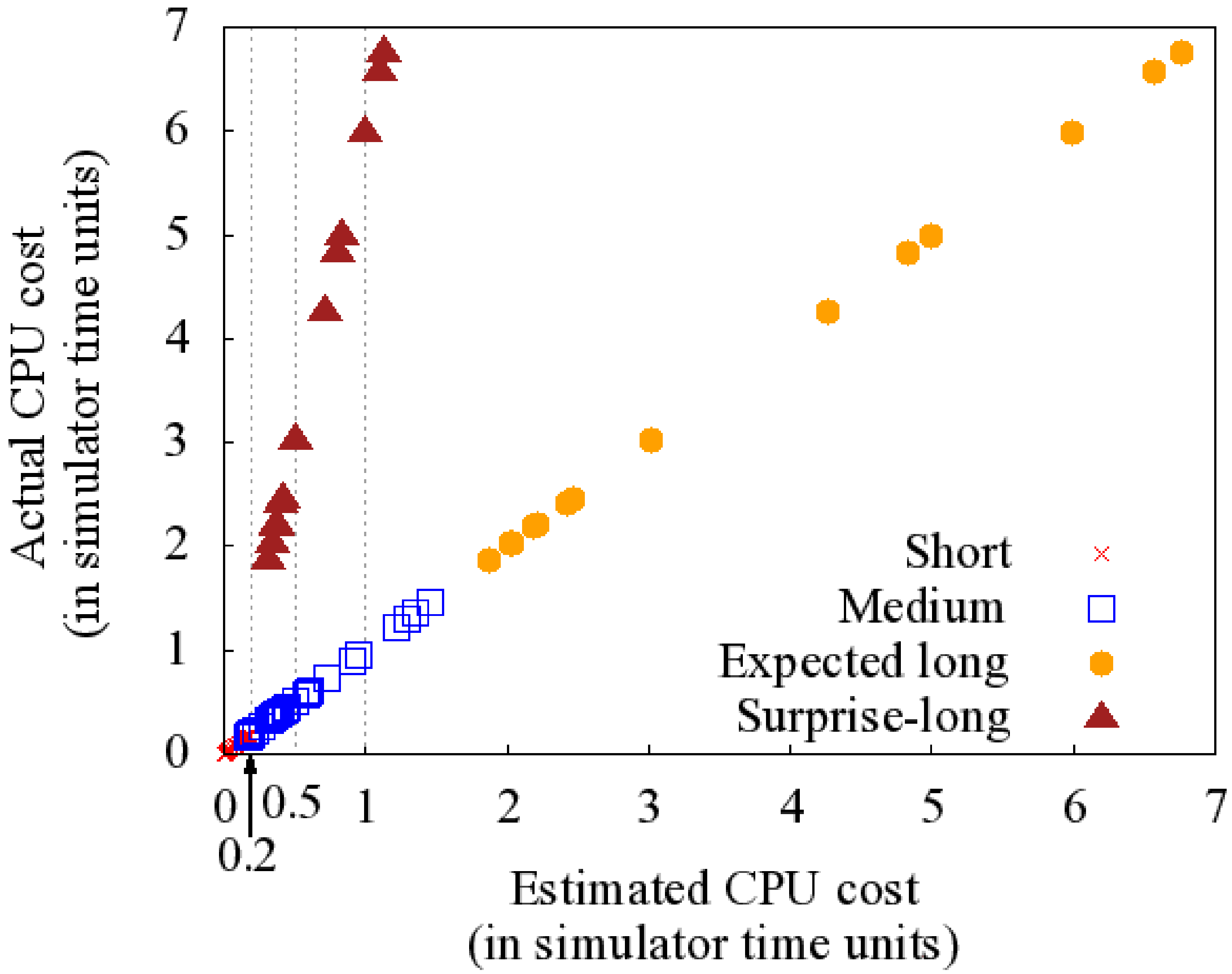
Taxonomy of long-running queries

Query type	Query expected to be long	Query progress reasonable	Uses equal share of resources
expected-long	yes	yes	yes
expected-hog	yes	yes	no (> equal)
surprise-long	no	yes	yes
surprise-hog	no	yes	no (> equal)
overload	no	no	yes
starving	no	no	no (< equal)

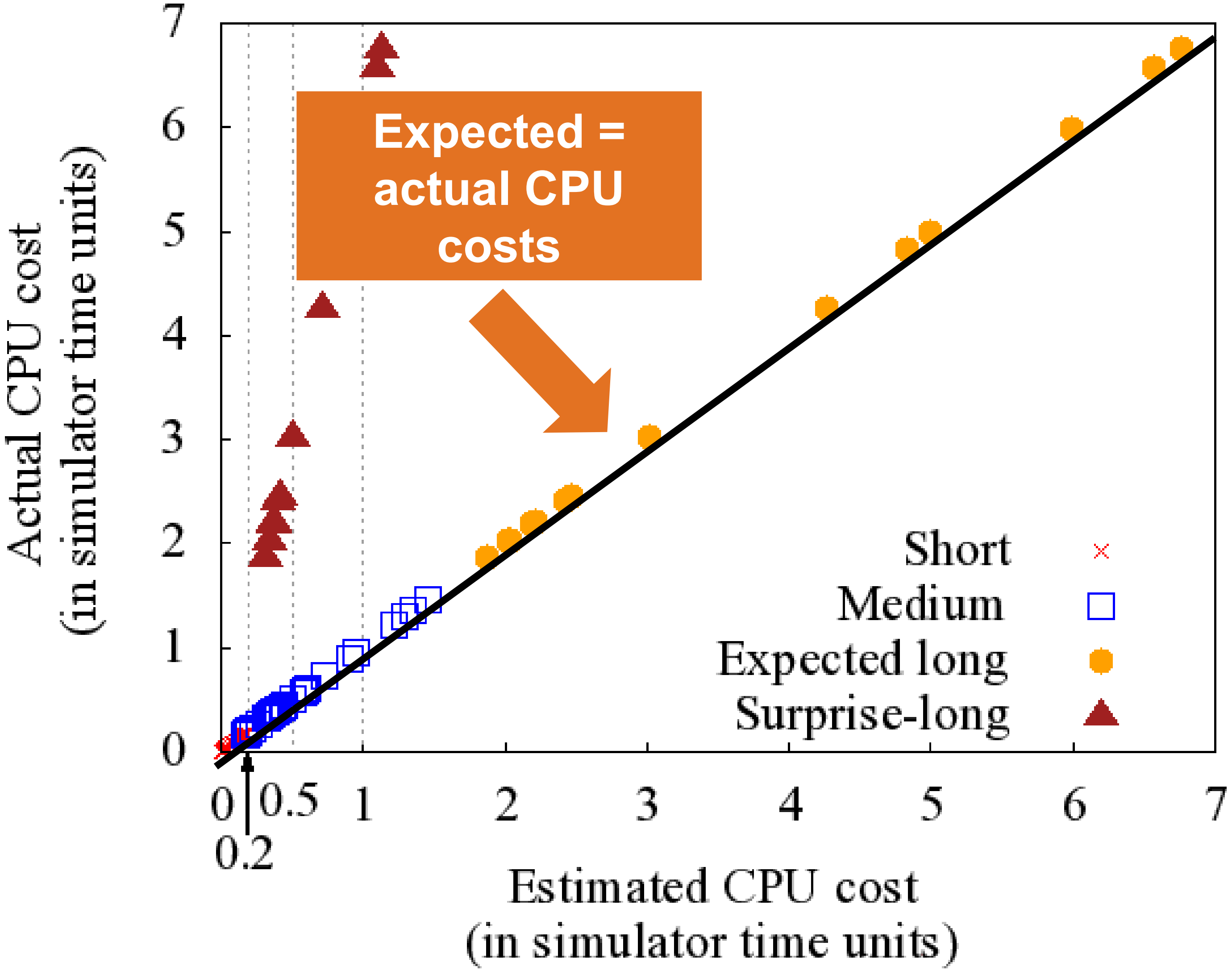
Experimental inputs

- **Workload types:** expected-long, surprise-long, ~~surprise-hog~~
- **Admission control**
  - Policies: none, limit expected costs of a query
  - Thresholds: 0.2m, 0.5m, and 1.0m
- **Scheduling**
  - Queue: FIFO
  - Multiprogramming level (MPL)
- **Execution control**
  - Policies: none, kill, kill&requeue, suspend&resume
  - Thresholds: absolute 5000 (time units), absolute 12000, absolute 5000 & progress < 30%, relative 1.2x

How did we choose the thresholds?

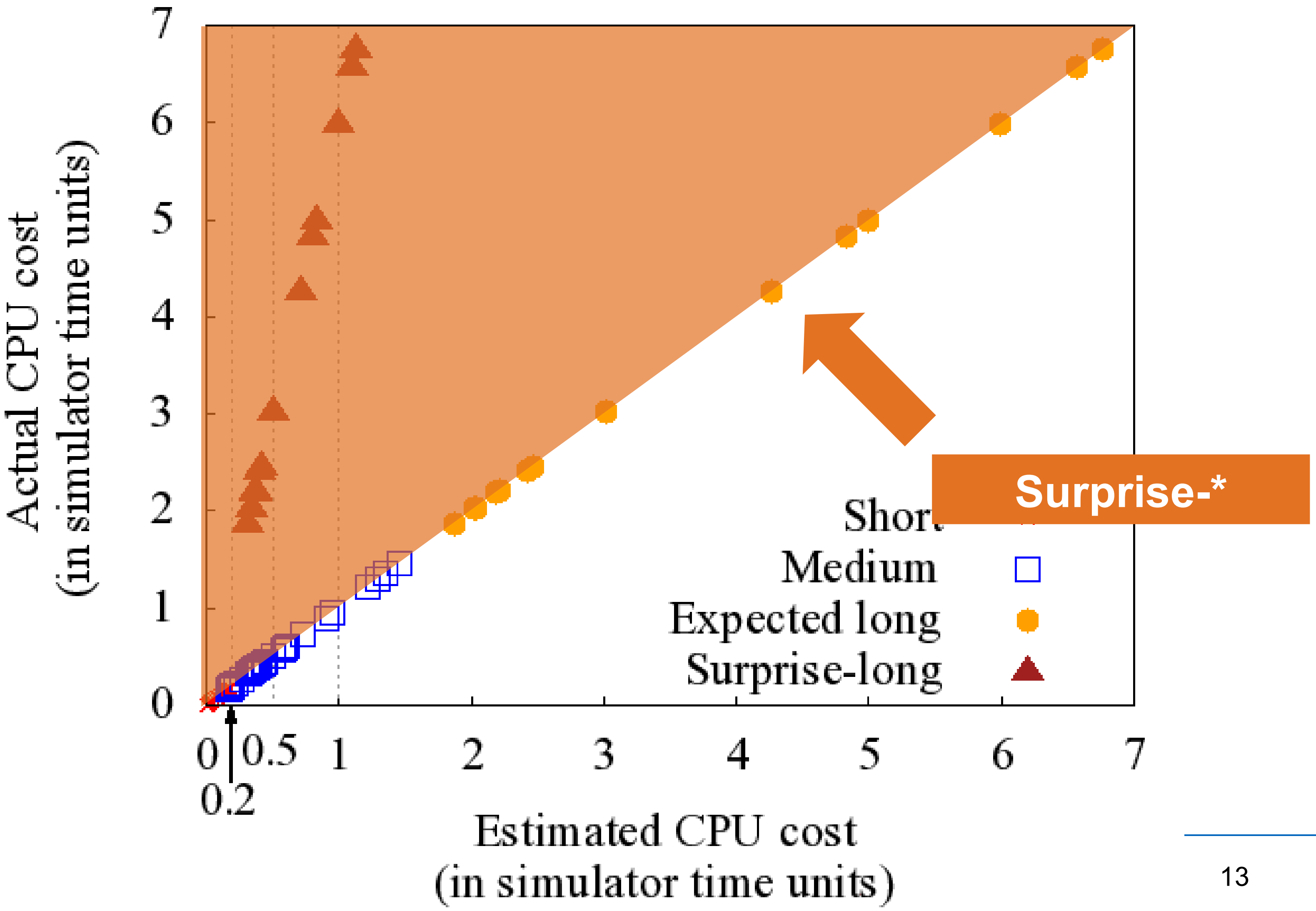


How did we choose the thresholds?

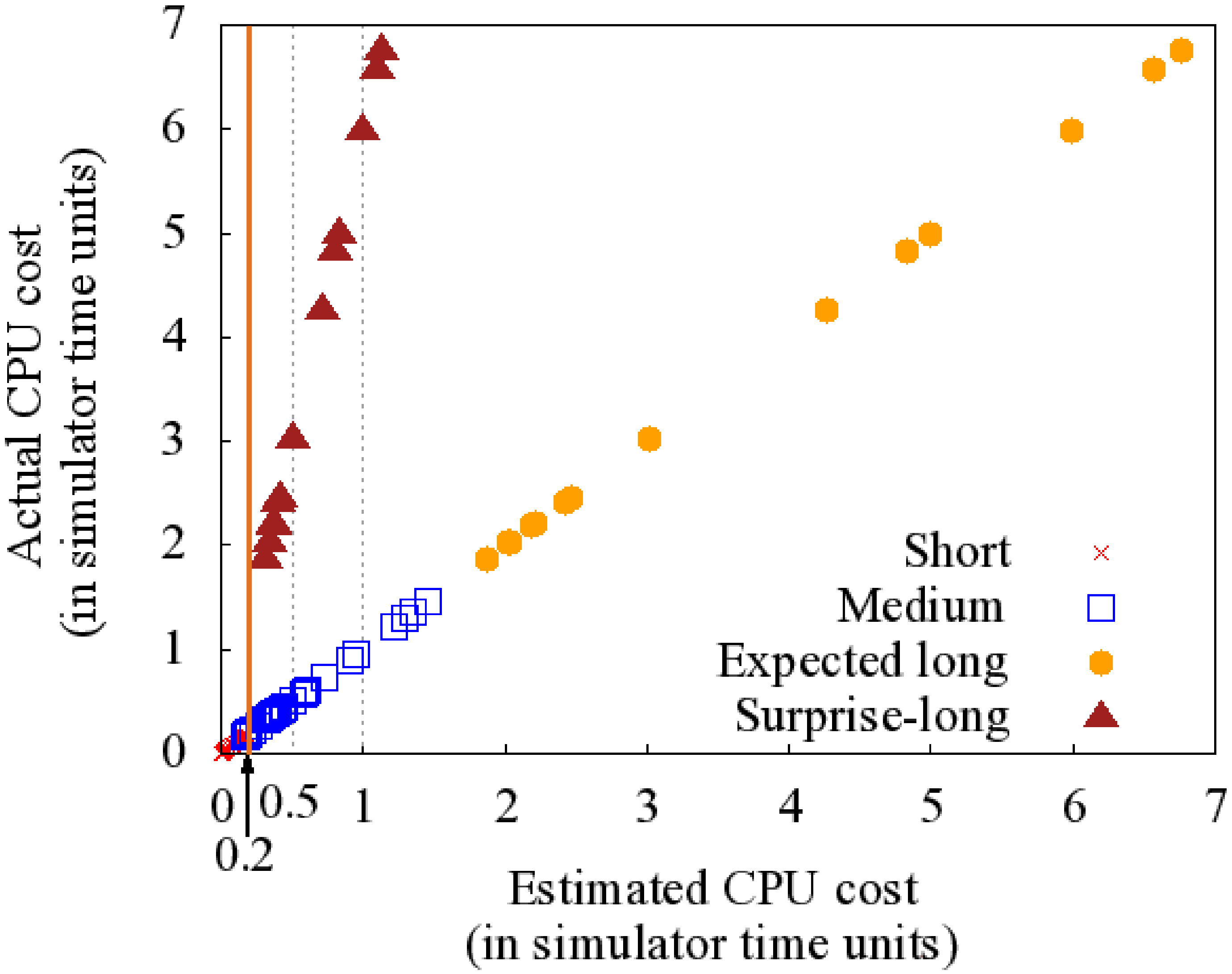


Expected = actual CPU costs

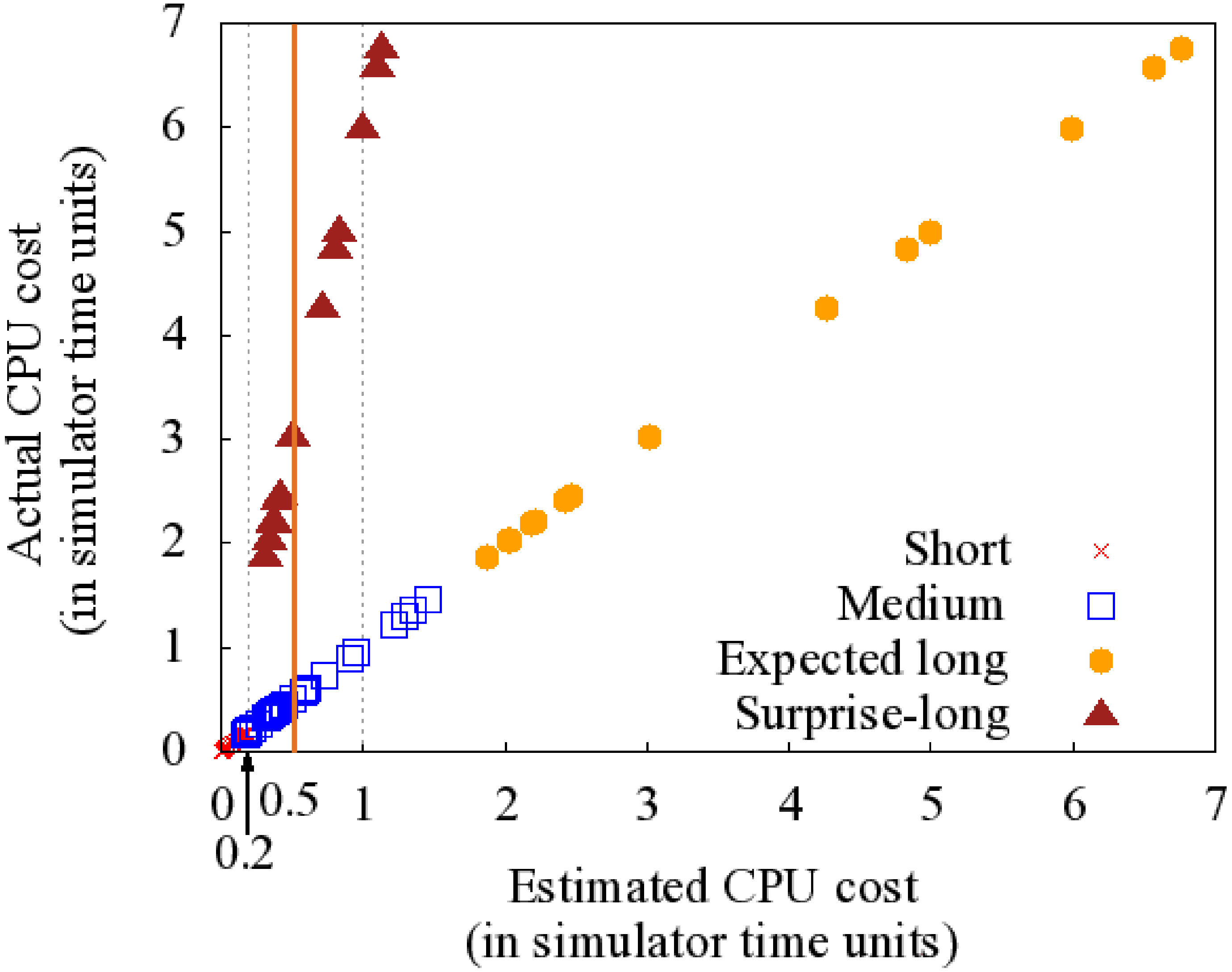
How did we choose the thresholds?



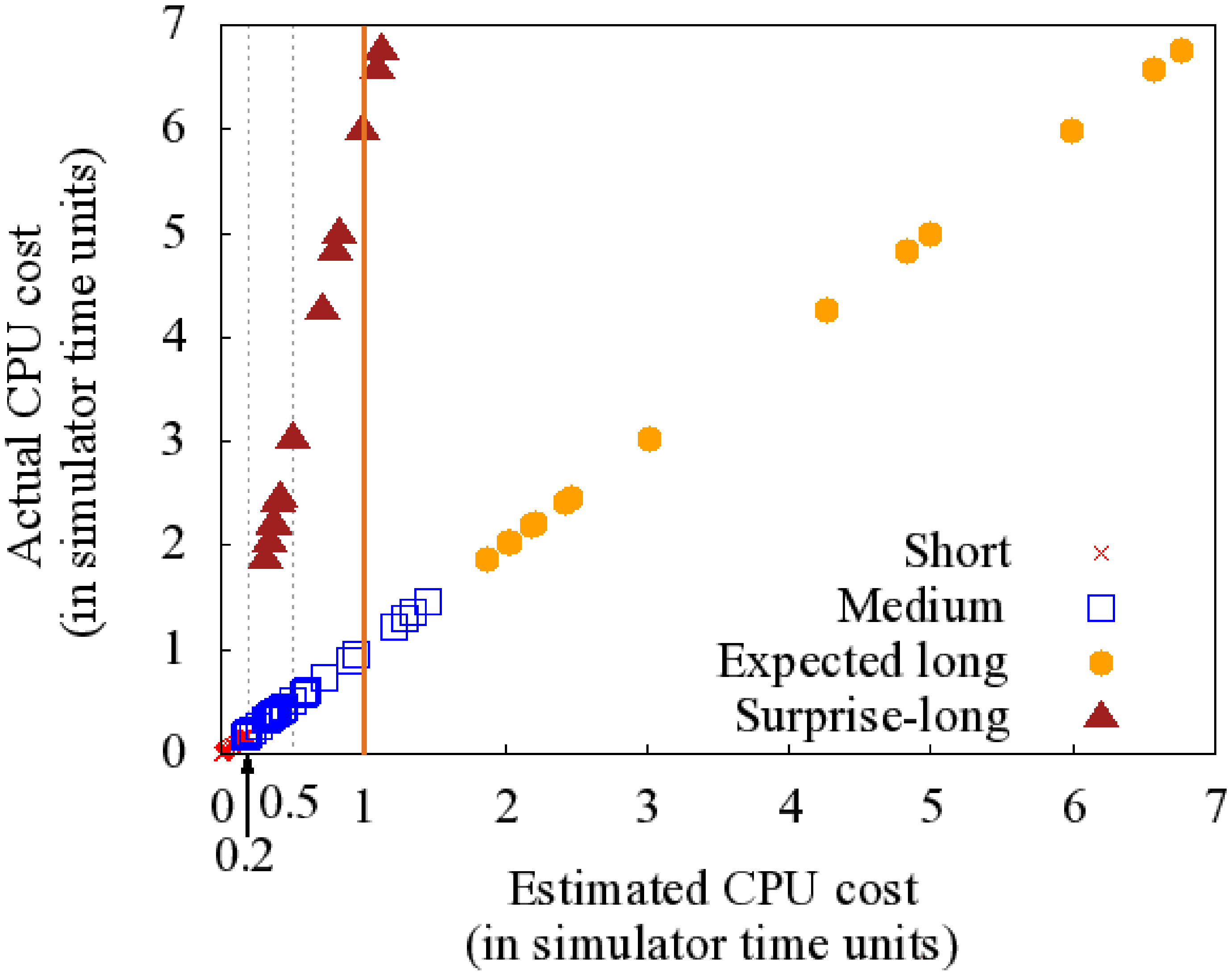
How did we choose the thresholds?



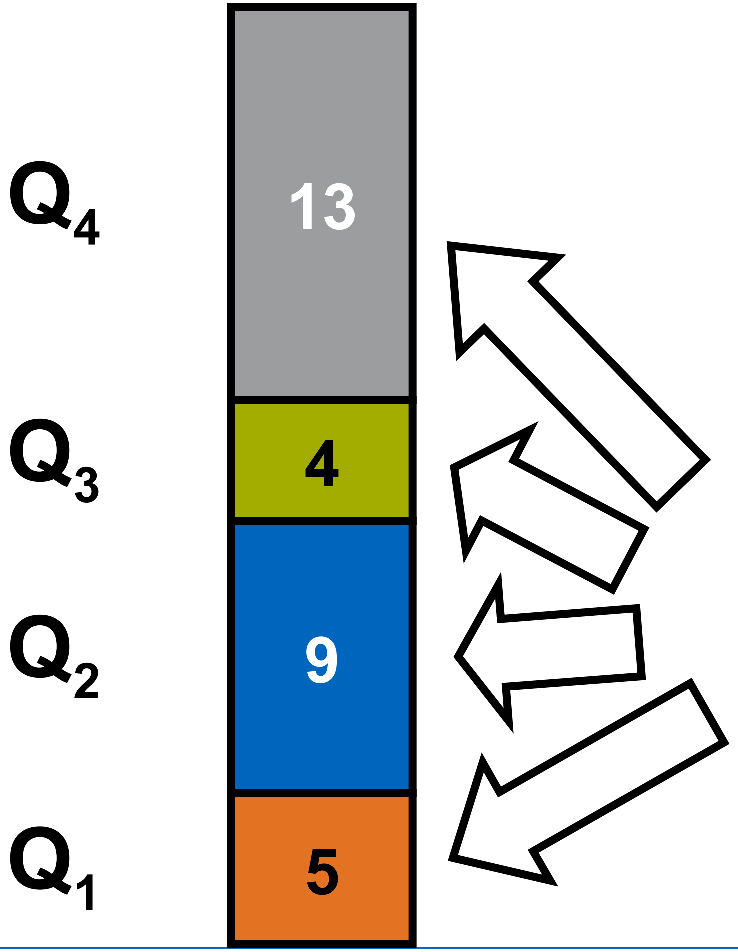
How did we choose the thresholds?



How did we choose the thresholds?

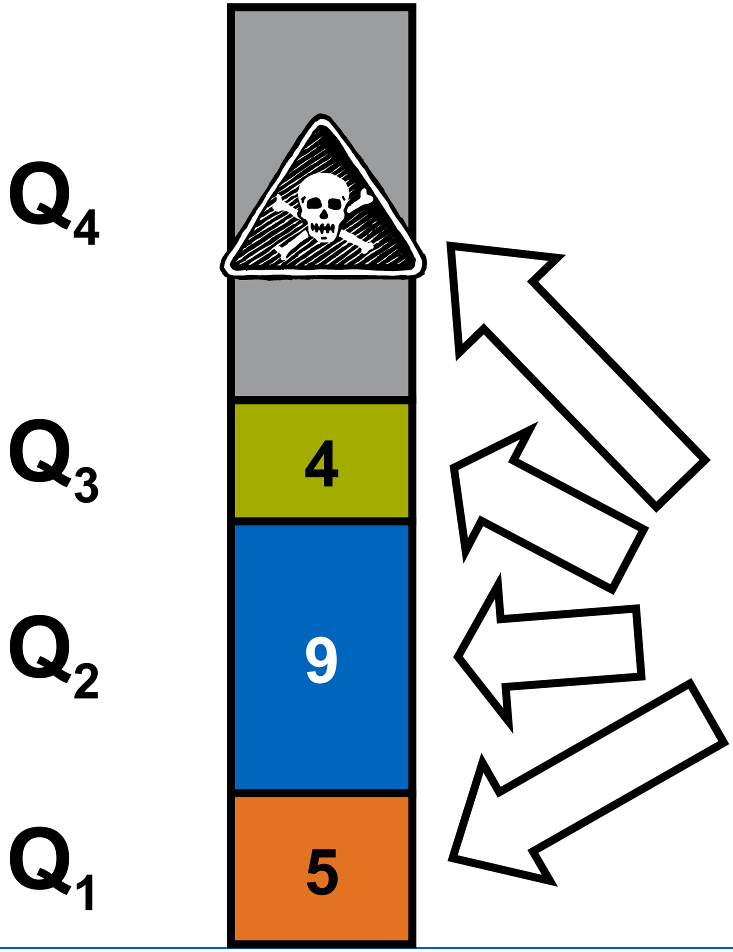


Experimental measure - weighted makespan



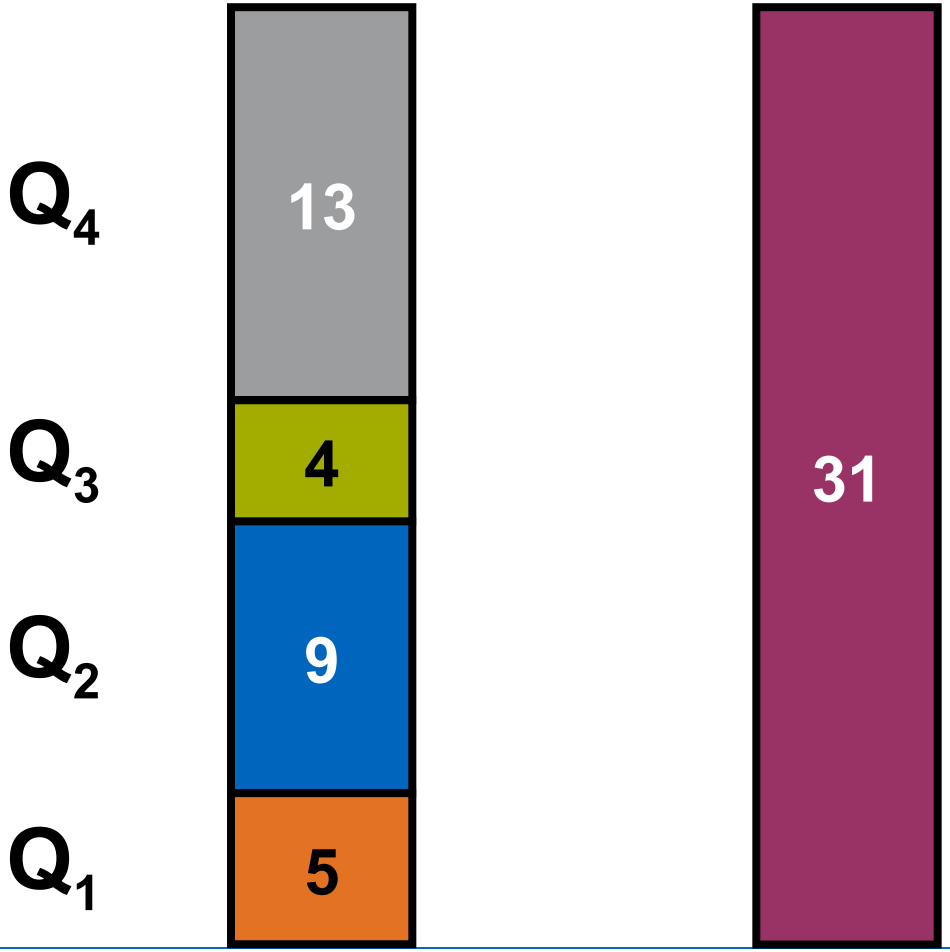
Elapsed time of queries

Experimental measure - weighted makespan



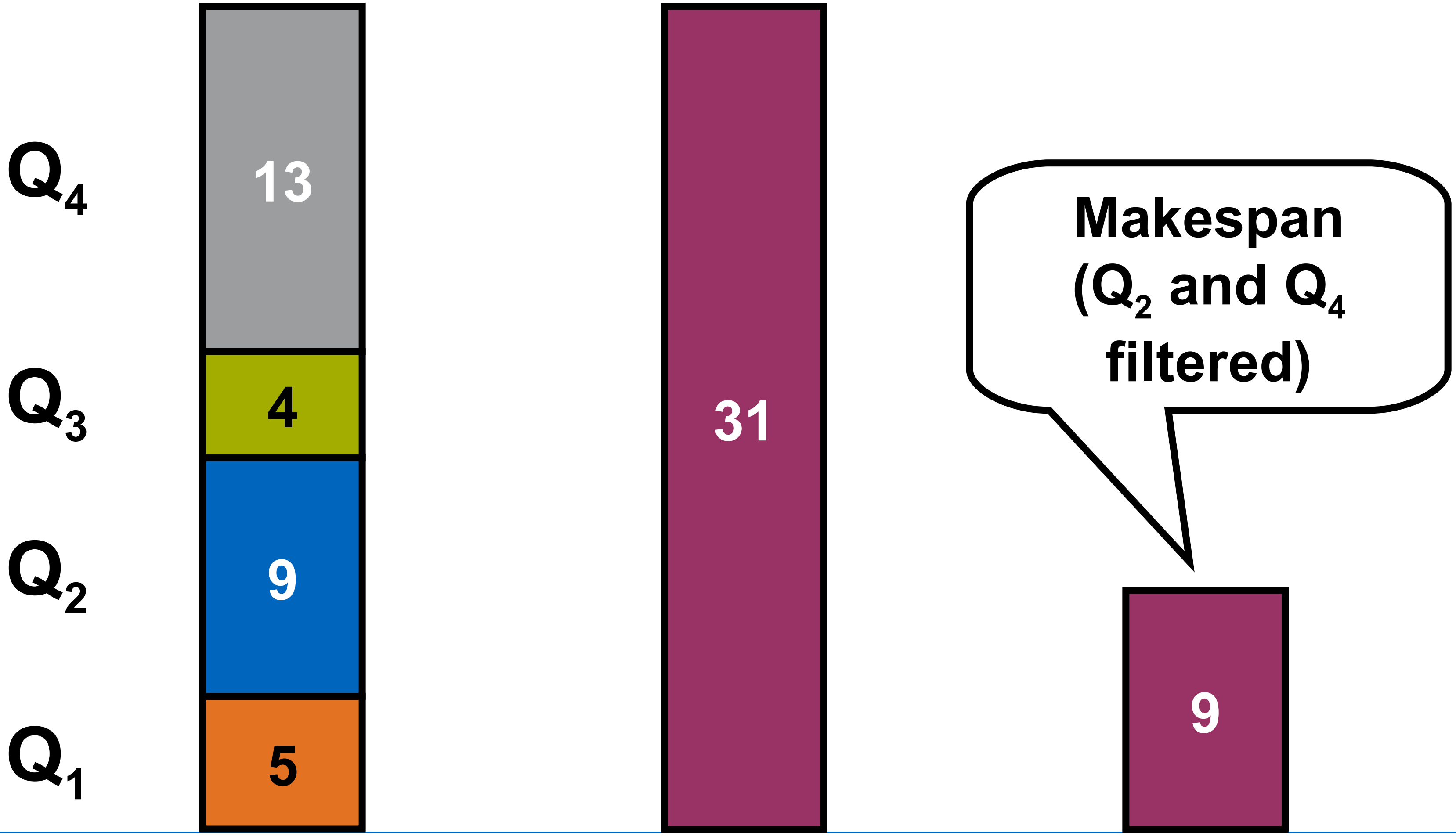
Elapsed time of queries

Experimental measure - weighted makespan



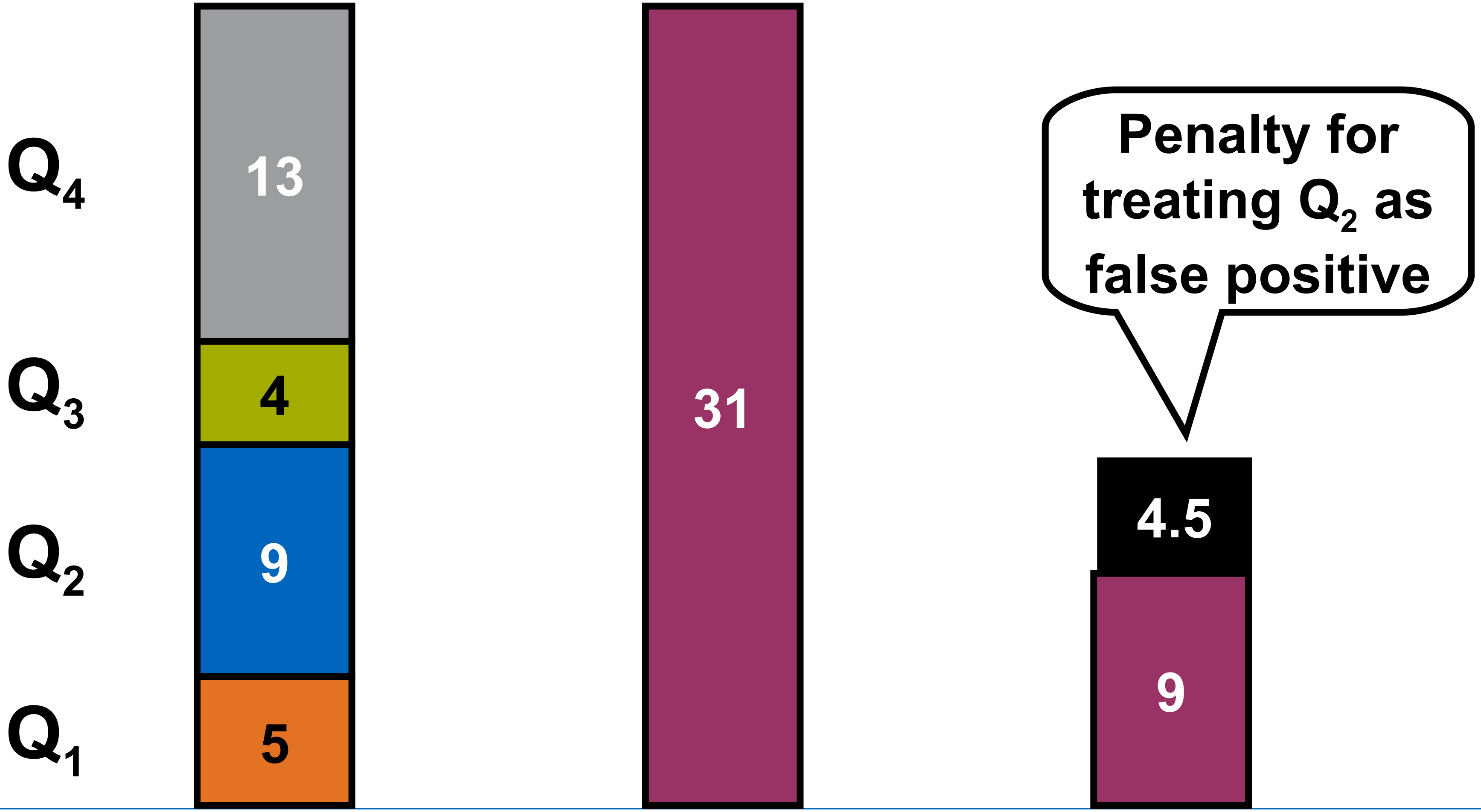
Makespan (Q_4 filtered, as expected ➔ no penalty)

Experimental measure - weighted makespan



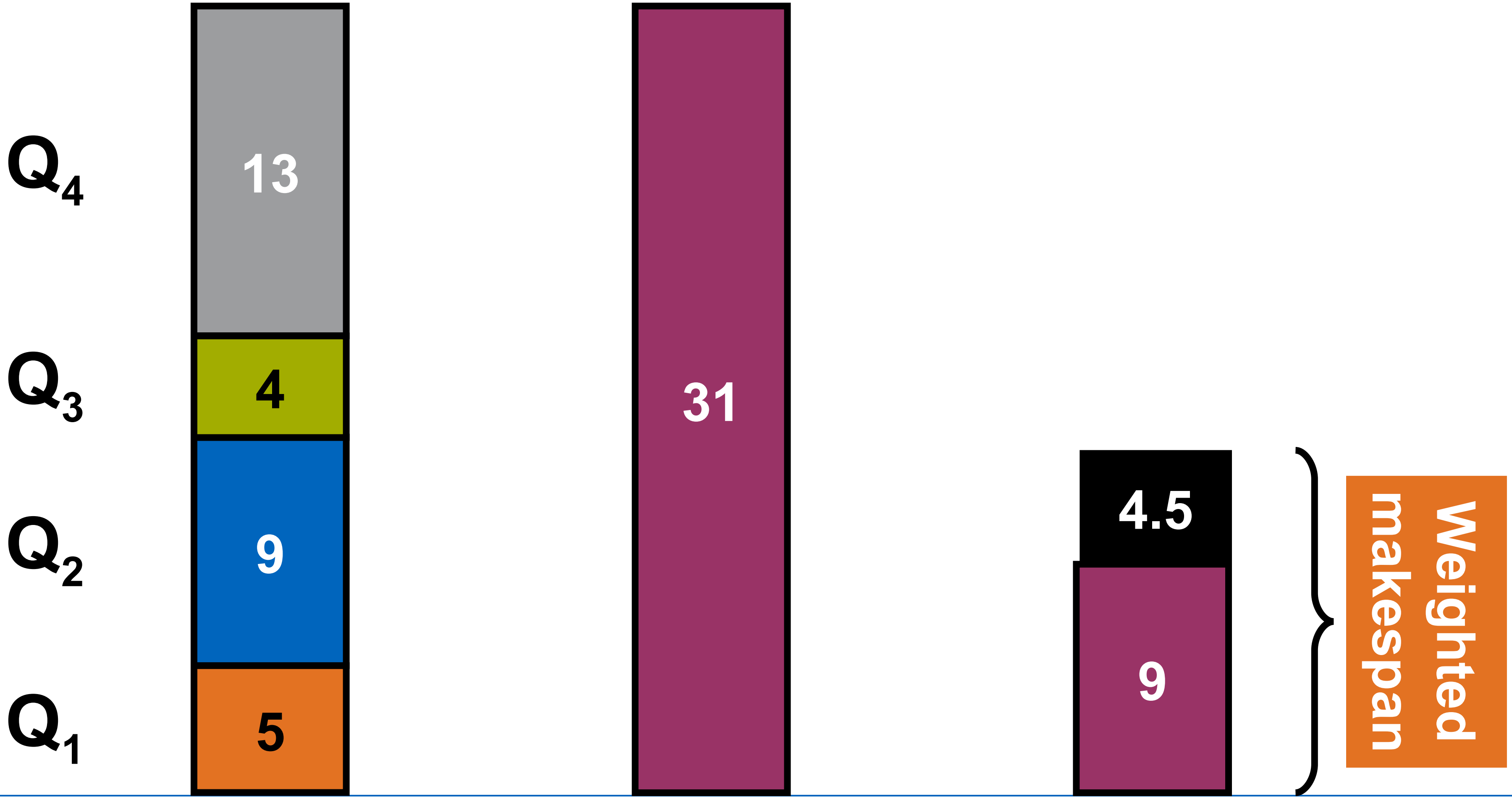
Makespan (Q_2 and Q_4 filtered)

Experimental measure - weighted makespan

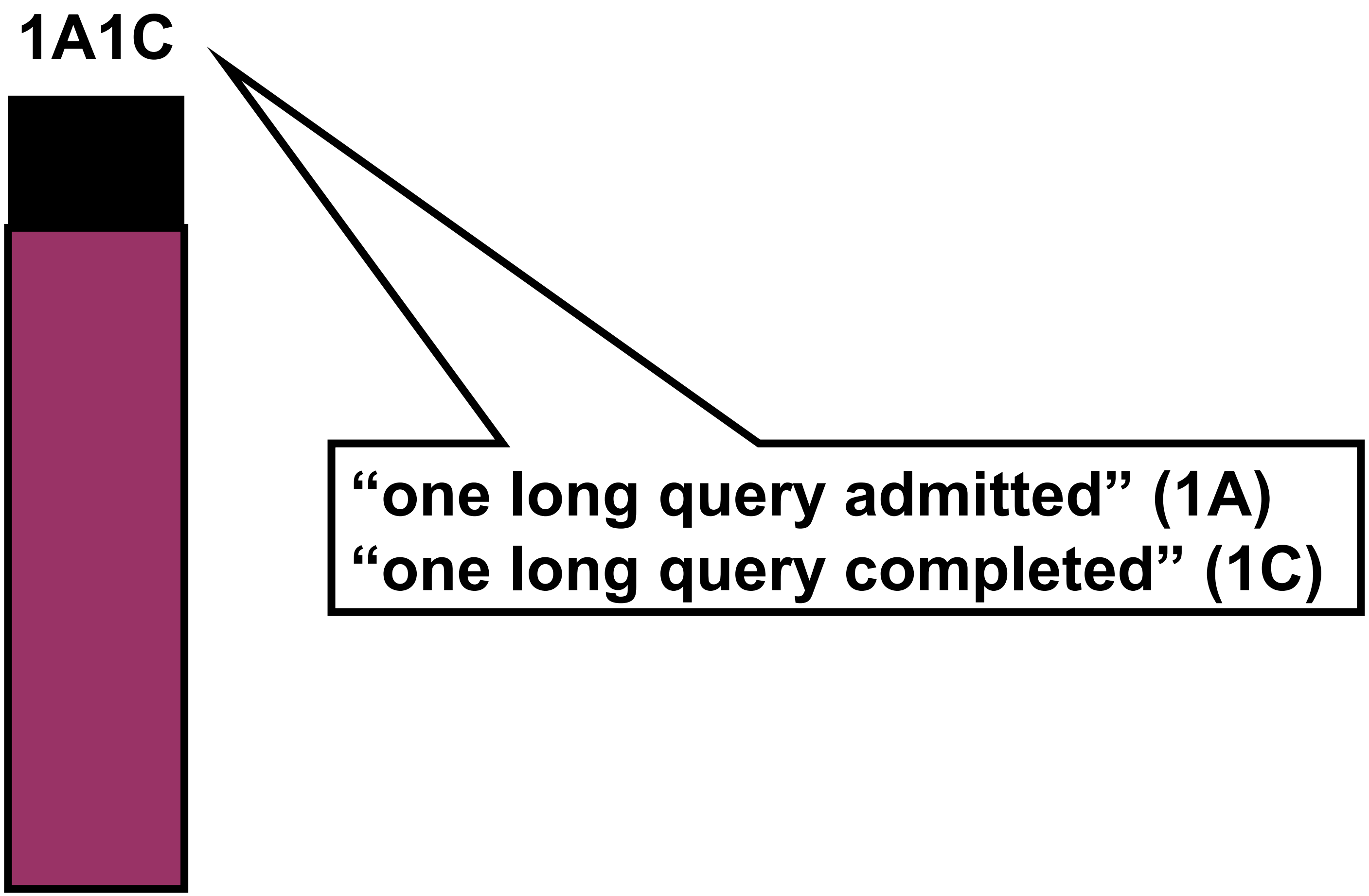


Penalty for treating Q_2 as false positive

Experimental measure - weighted makespan



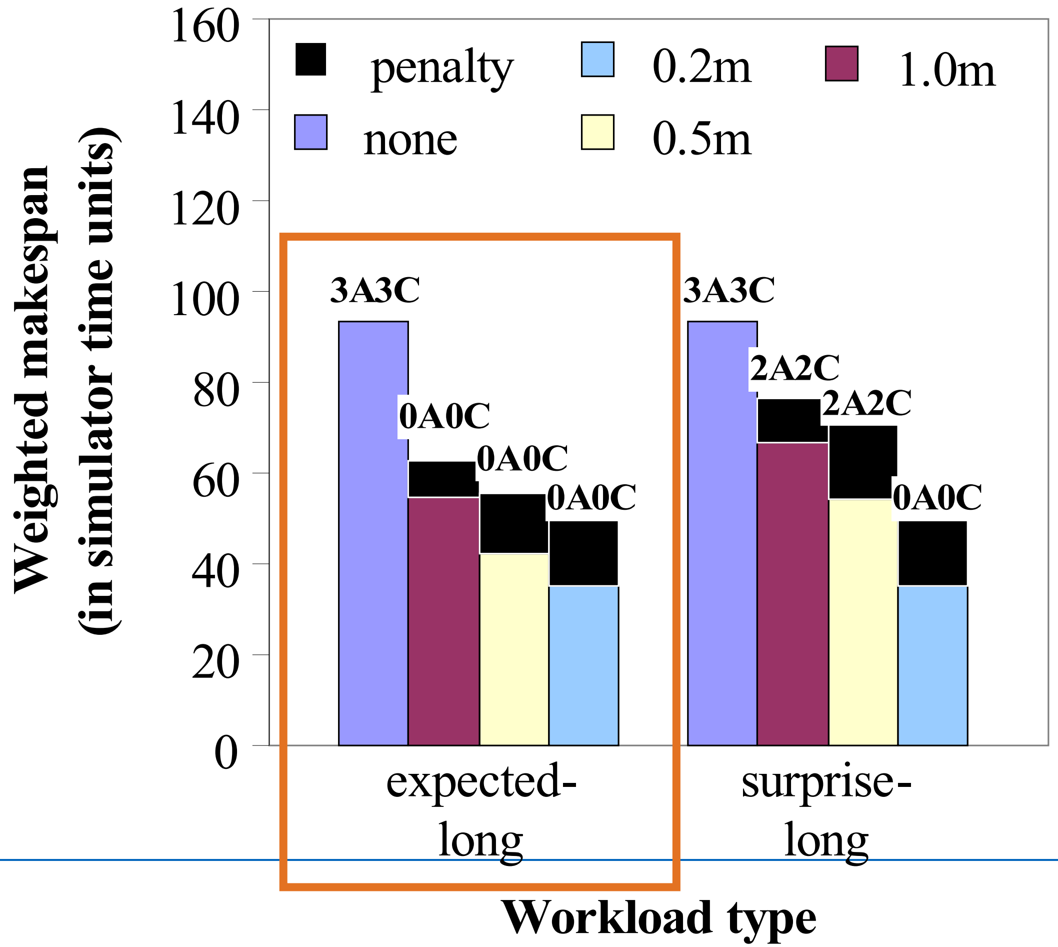
Experimental measure - weighted makespan



"one long query admitted" (1A)
"one long query completed" (1C)

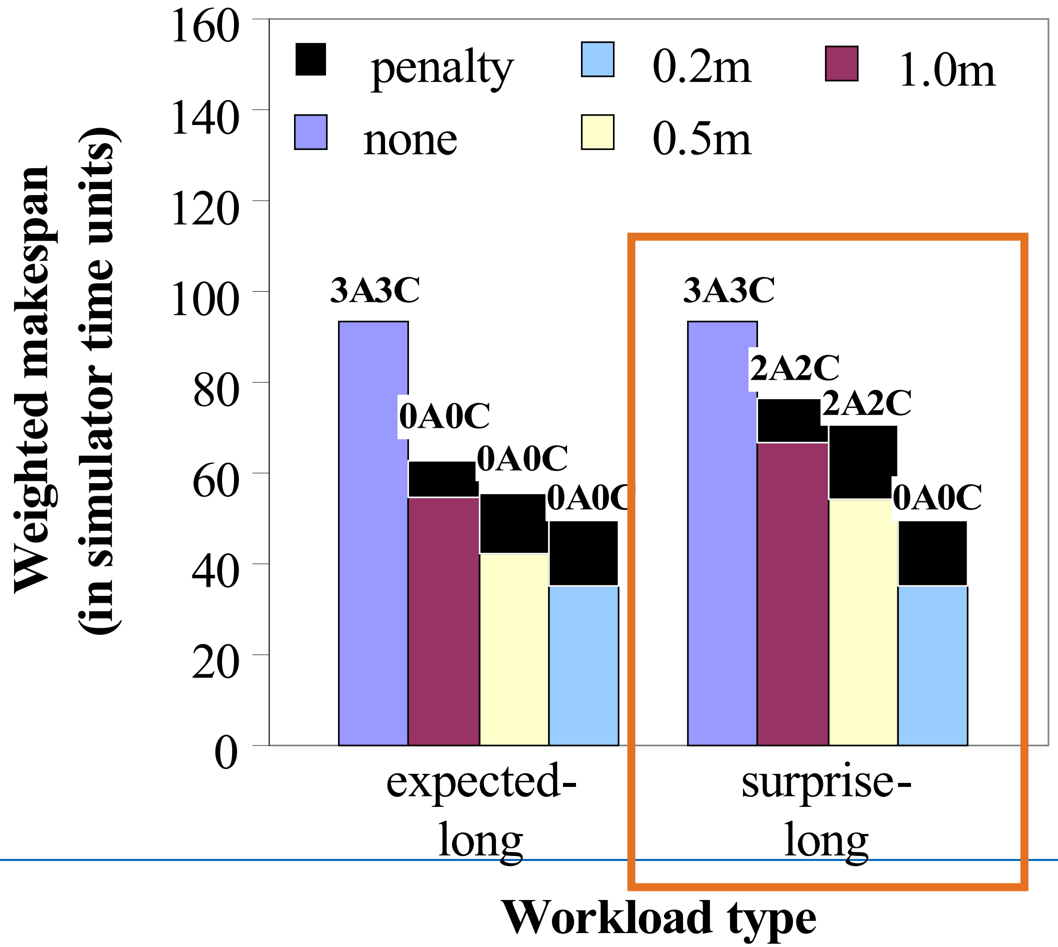
Can WM handle unreliable cost estimates?

Admission control thresholds



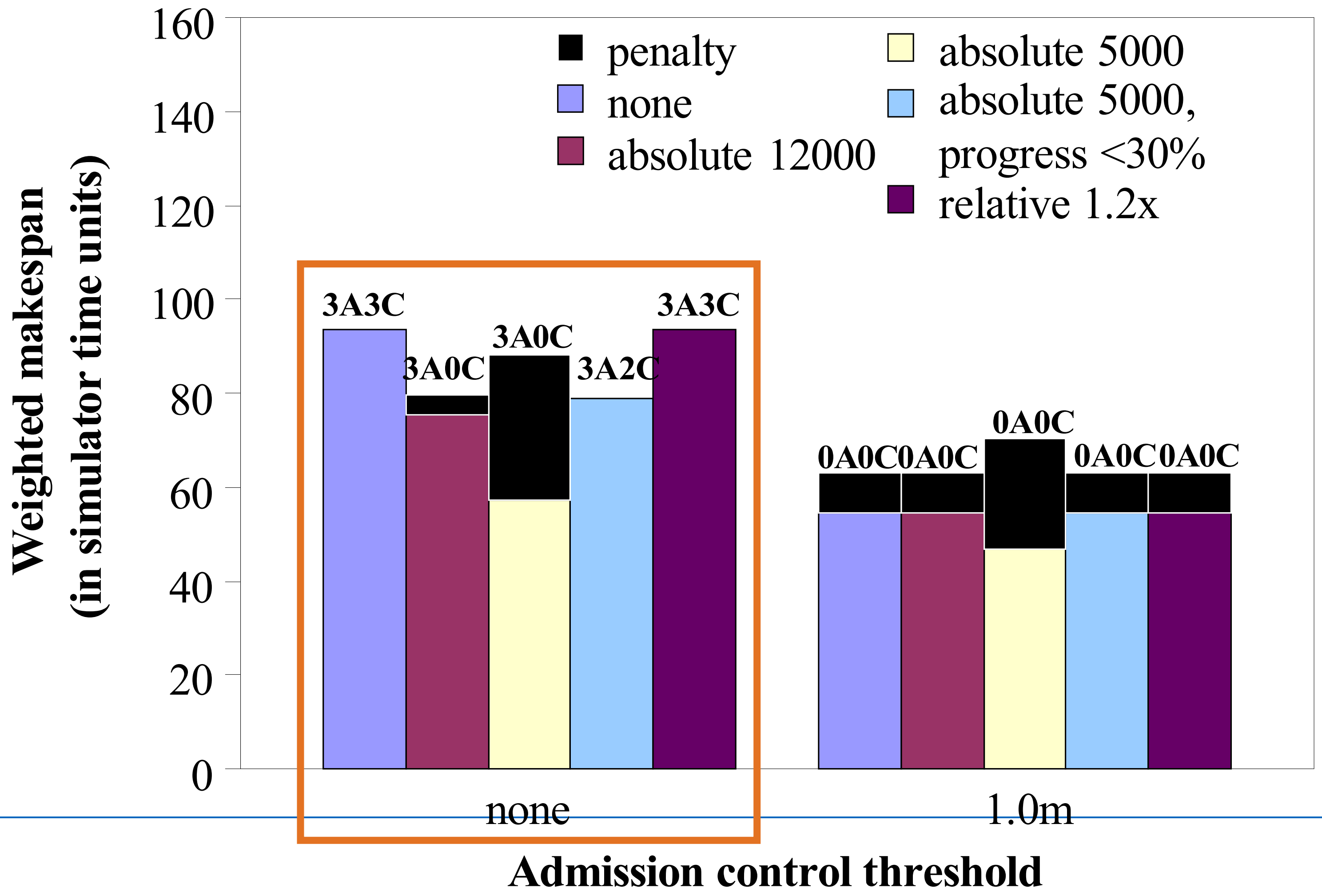
Can WM handle unreliable cost estimates?

Admission control thresholds



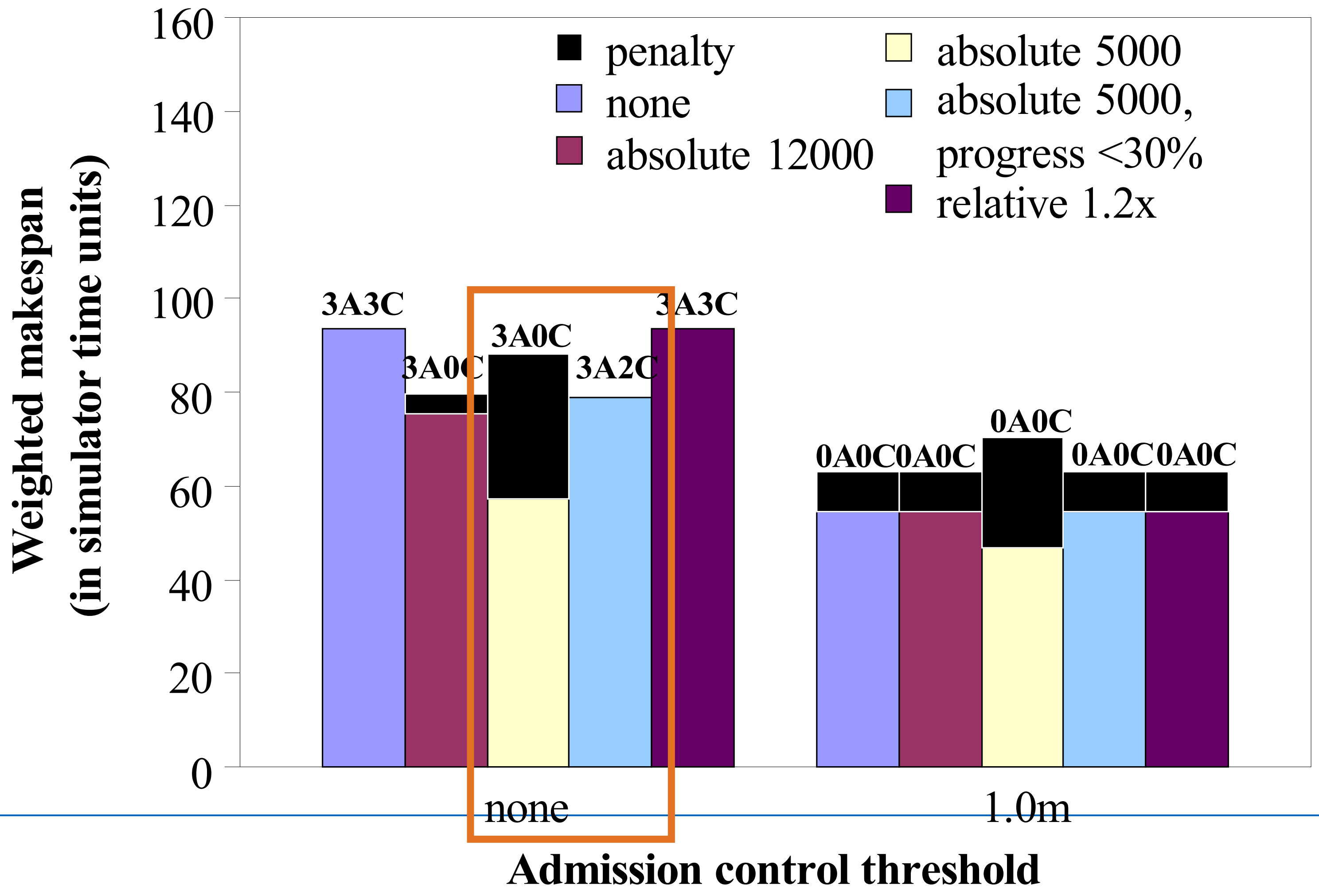
Can WM handle unreliable cost estimates?

Adm ctl + exec ctl with different kill thresholds



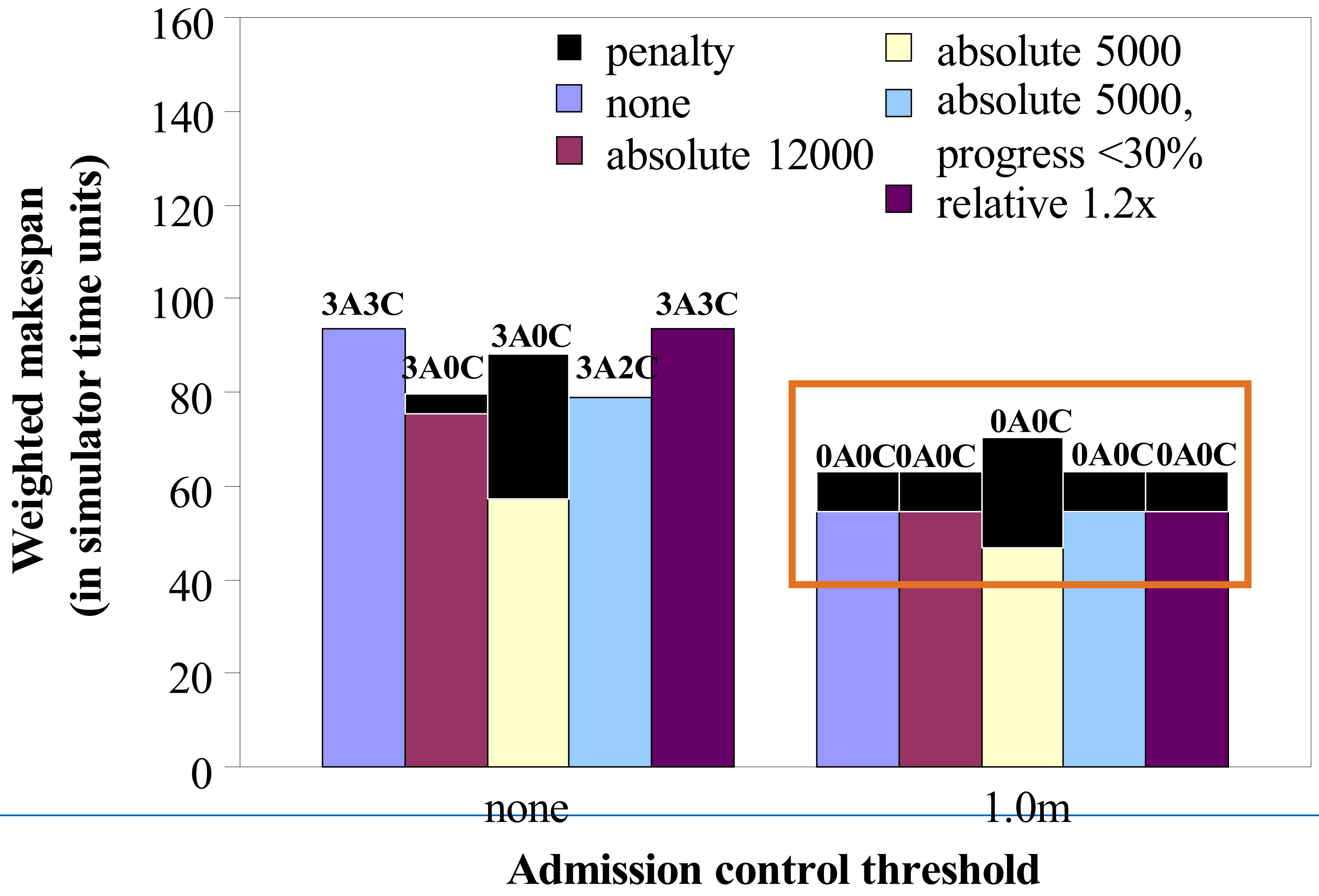
Can WM handle unreliable cost estimates?

Adm ctl + exec ctl with different kill thresholds



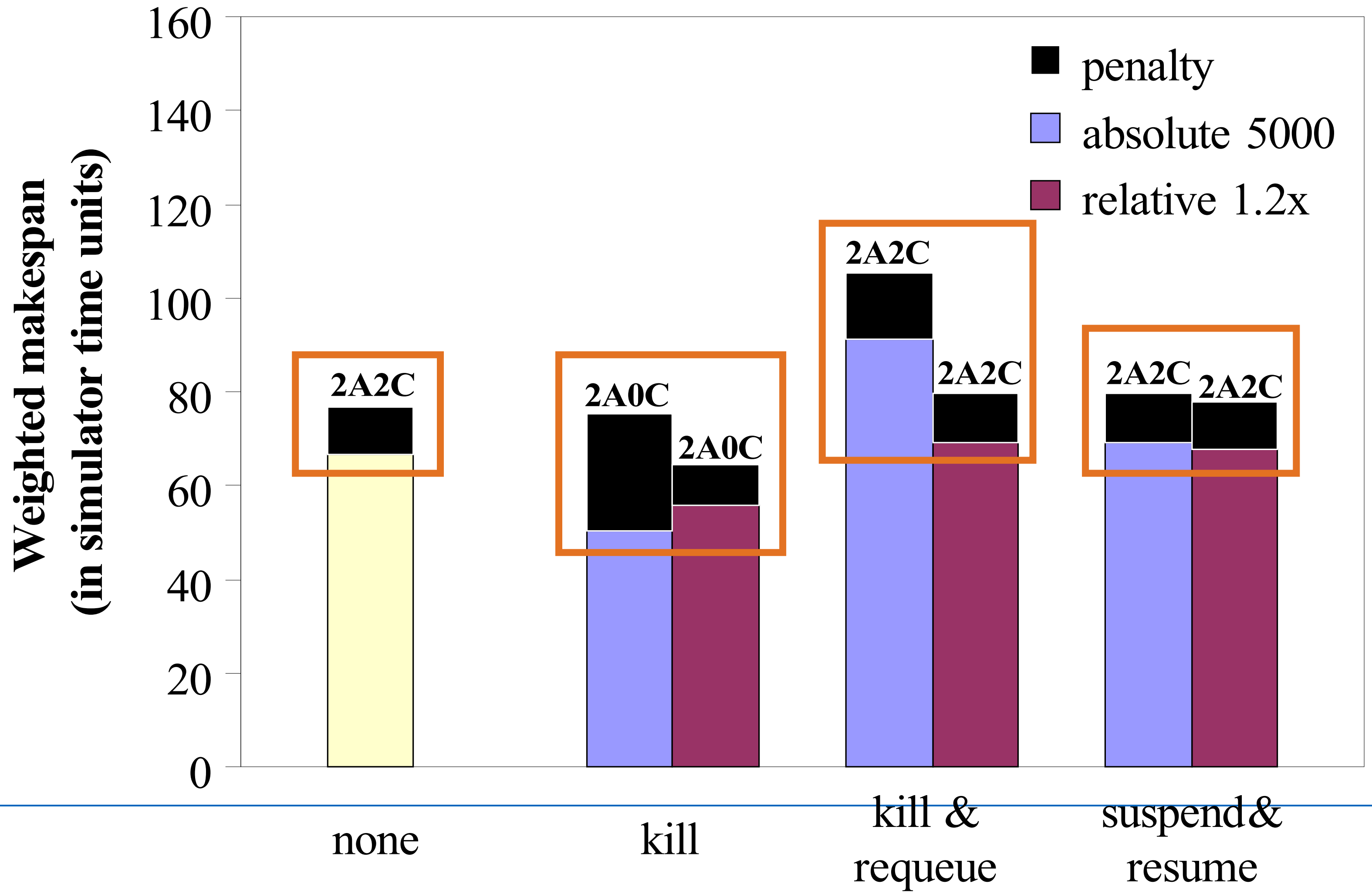
Can WM handle unreliable cost estimates?

Adm ctl + exec ctl with different kill thresholds



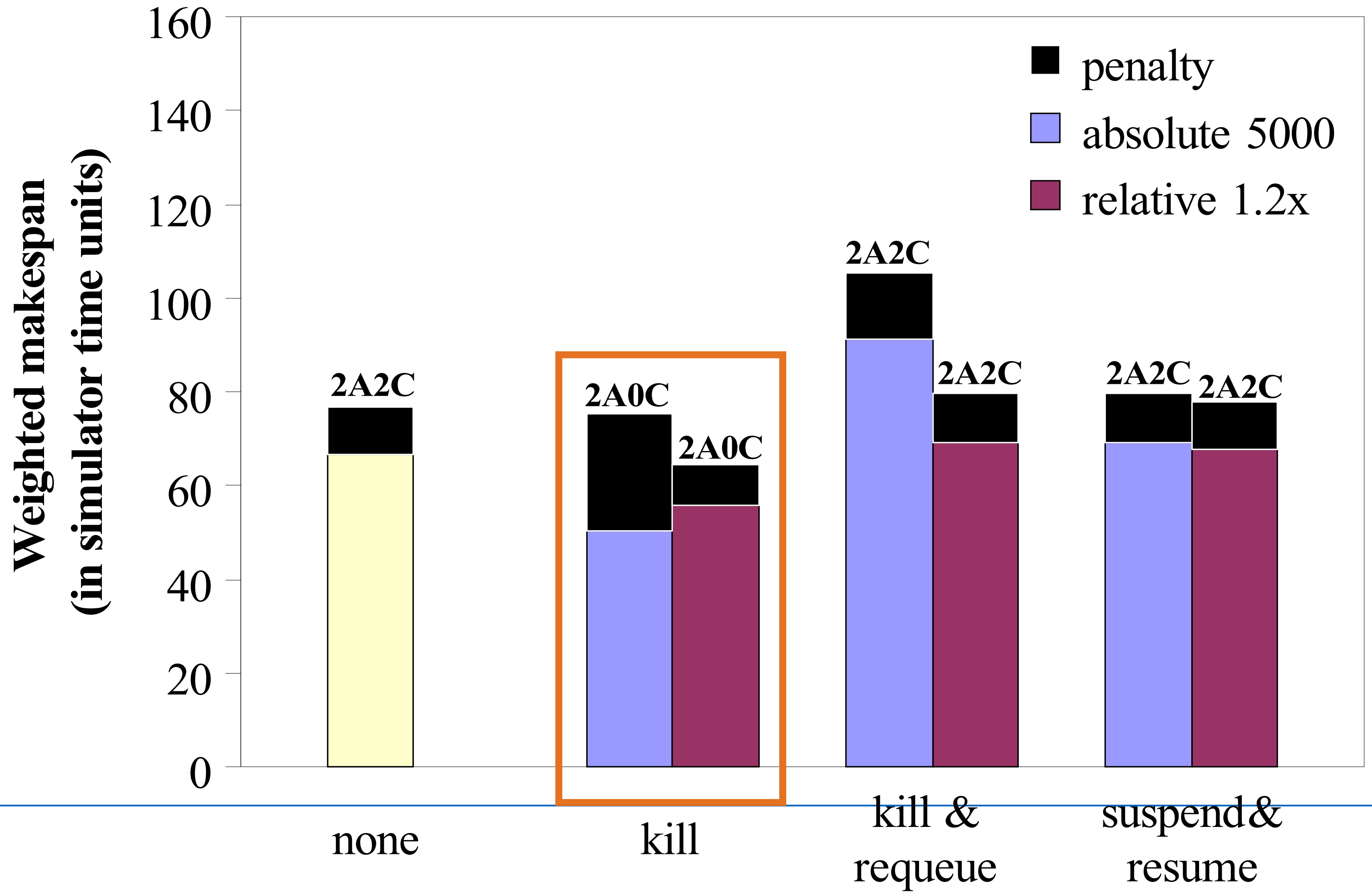
Can WM handle unreliable cost estimates?

Execution control actions (w/ admission control 1.0m)



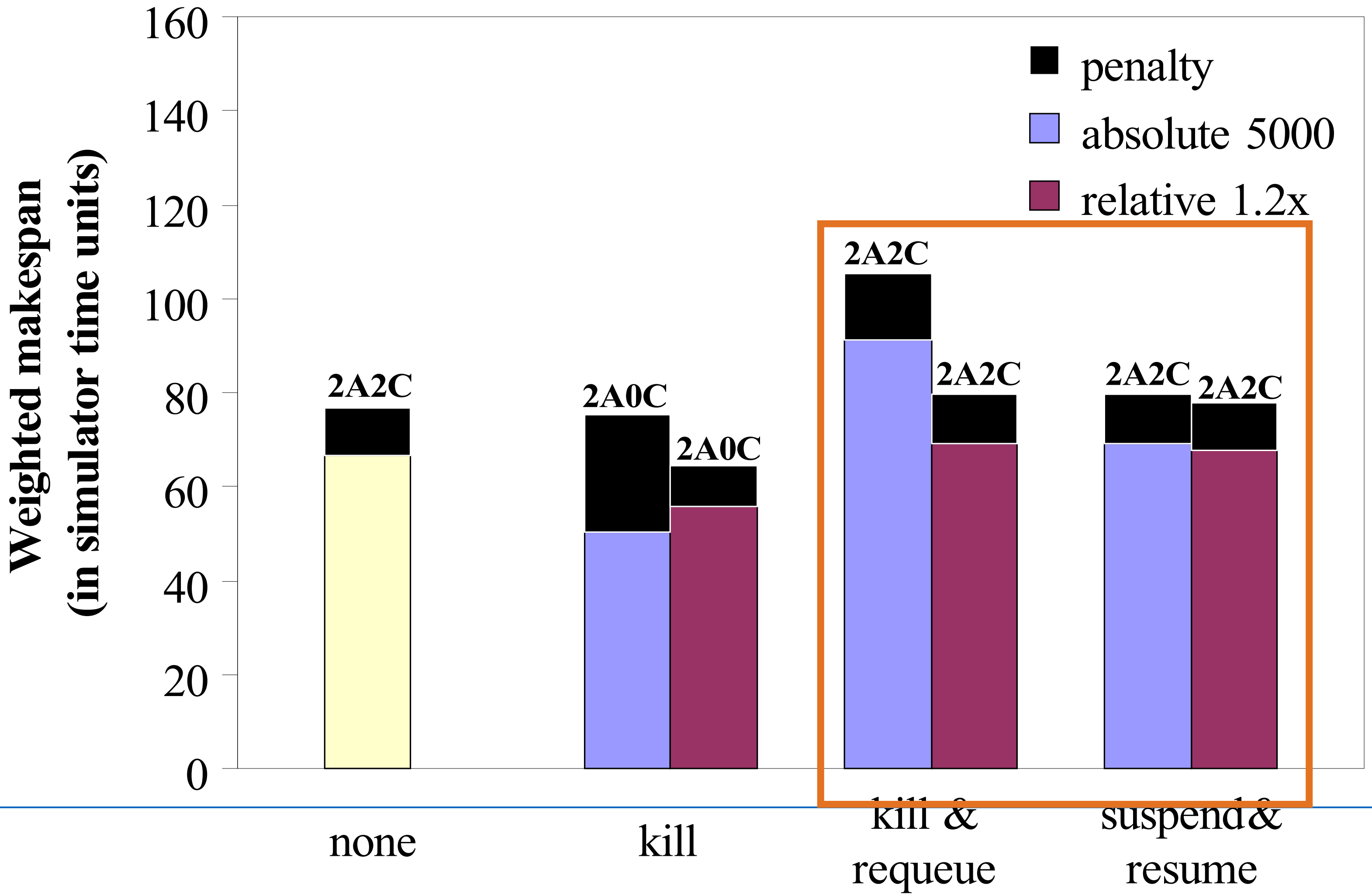
Can WM handle unreliable cost estimates?

Execution control actions (w/ admission control 1.0m)

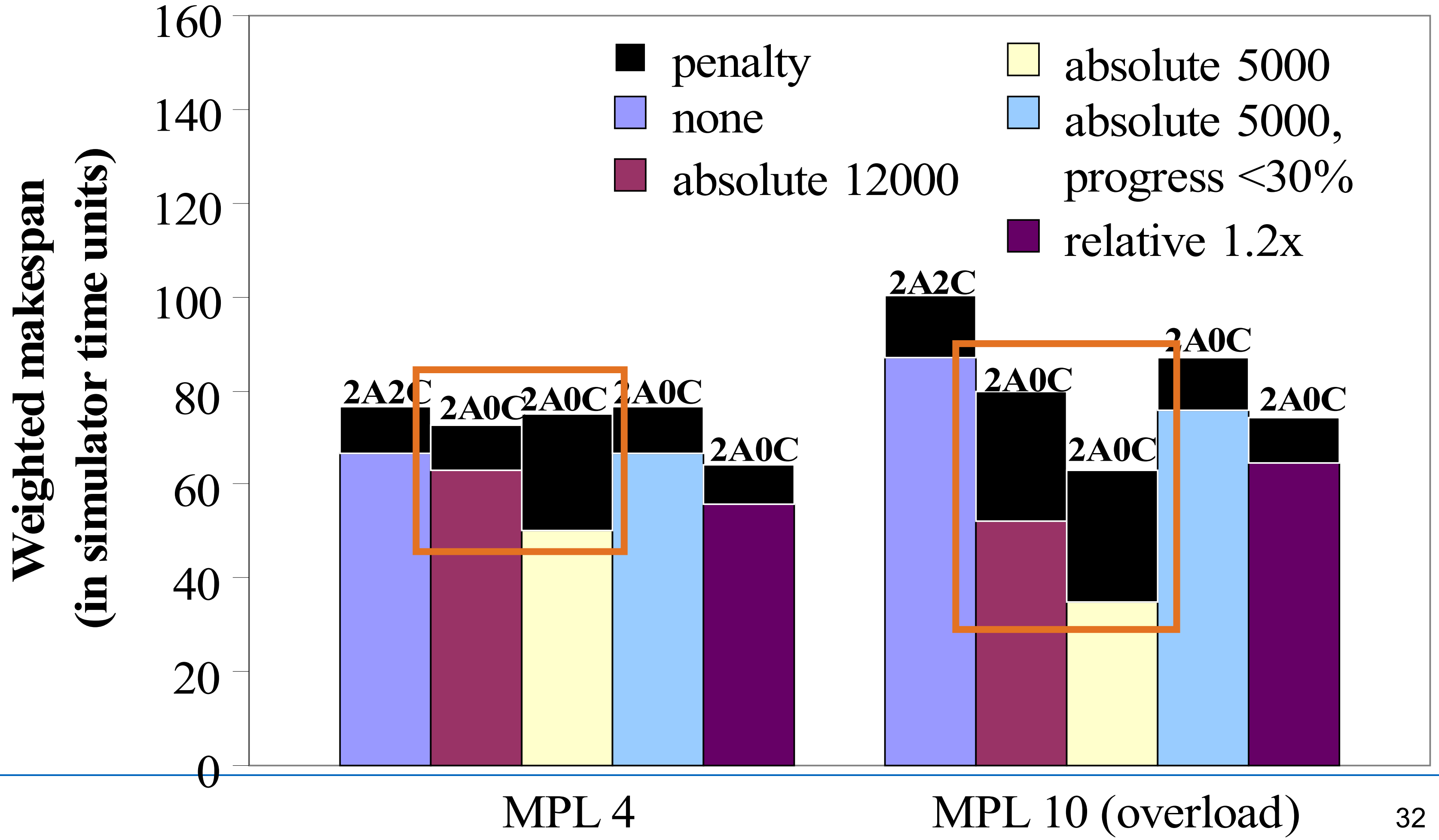


Can WM handle unreliable cost estimates?

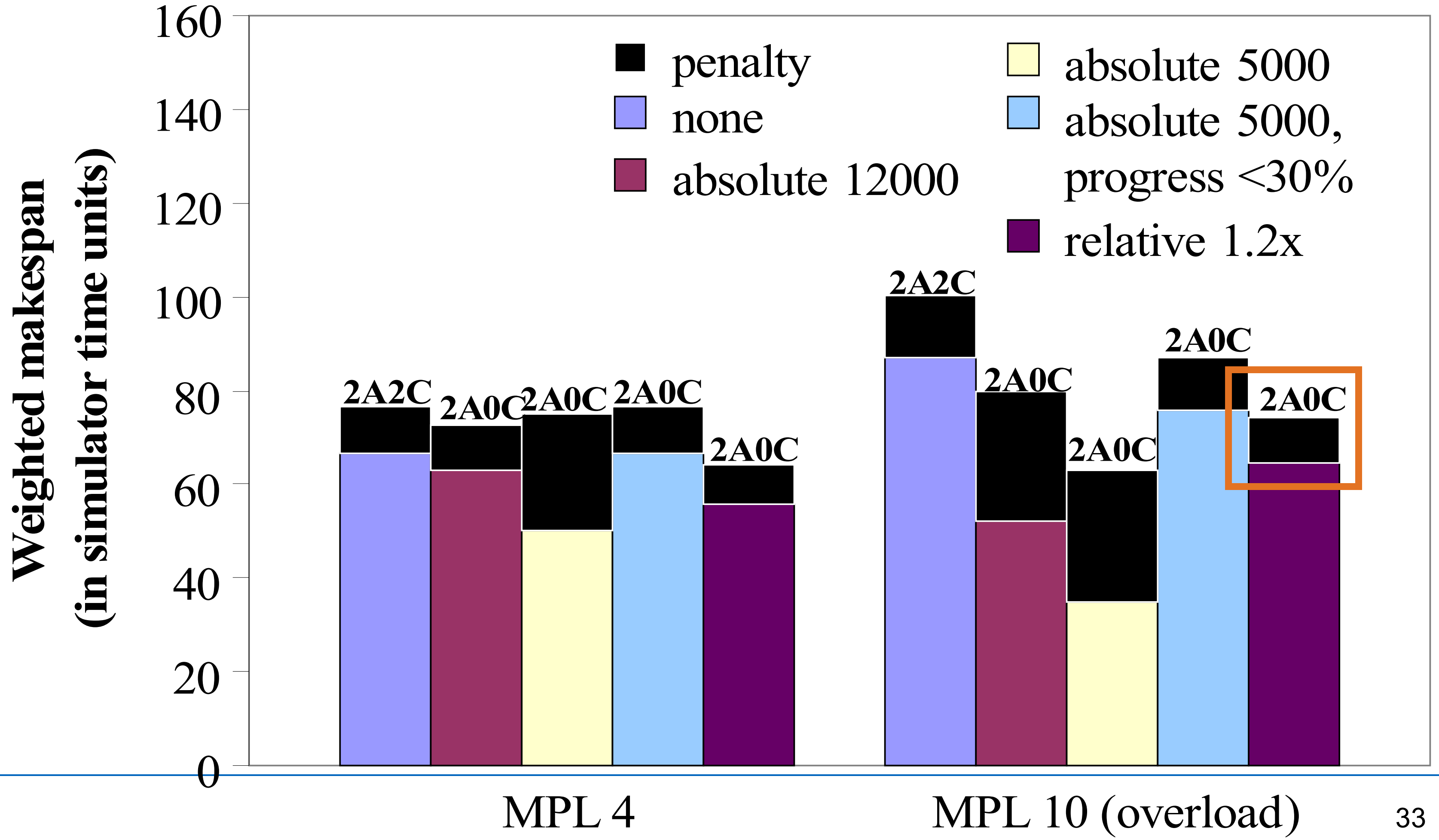
Execution control actions (w/ admission control 1.0m)



Can workload management handle system overload?



Can workload management handle system overload?



Conclusion

- Systematic study of workload management policies to mitigate the impact of long-running queries
- Can workload management handle…
  - unreliable cost estimates ✓
  - unobserved resource contention ✓
  - system overload ✗

- Value of this work: experimental framework for studying more challenging workload management problems

Related work (excerpt)

D. G. Benoit. *Automated Diagnosis and Control of DBMS Resources*. EDBT PhD. Workshop, 2000

S. Chaudhuri, R. Kaushik, and R. Ramamurthy. *When Can We Trust Progress Estimators for SQL Queries?* SIGMOD 2005

S. Chaudhuri, R. Kaushik, R. Ramamurthy, and A. Pol. *Stop-and-Restart Style Execution for Long Running Decision Support Queries*. VLDB 2007

S. Krompass, H. Kuno, U. Dayal, and A. Kemper. *Dynamic Workload Management for Very Large Data Warehouses: Juggling Feathers and Bowling Balls*. *VLDB* 2007

G. Luo, J. F. Naughton, and P. S. Yu. *Multi-query SQL Progress Indicators*, EDBT 2006

Workload management tools: HP Neoview, IBM Workload Manager for DB2, Microsoft SQL Server, Oracle Database Resource Manager, Teradata Dynamic Workload Manager