Dynamic Workload Management for Very Large Data Warehouses

Juggling Feathers and Bowling Balls

Stefan Krompass
Alfons Kemper
TU München
Germany

Harumi Kuno
Umeshwar Dayal
HP Labs, Palo Alto, CA
USA

Outline

- Problem statement
- Proposed solution
- Evaluation
  - Approach and settings for experiments
  - Impact of problem queries on a workload
  - Impact of execution control
- Conclusion and ongoing work

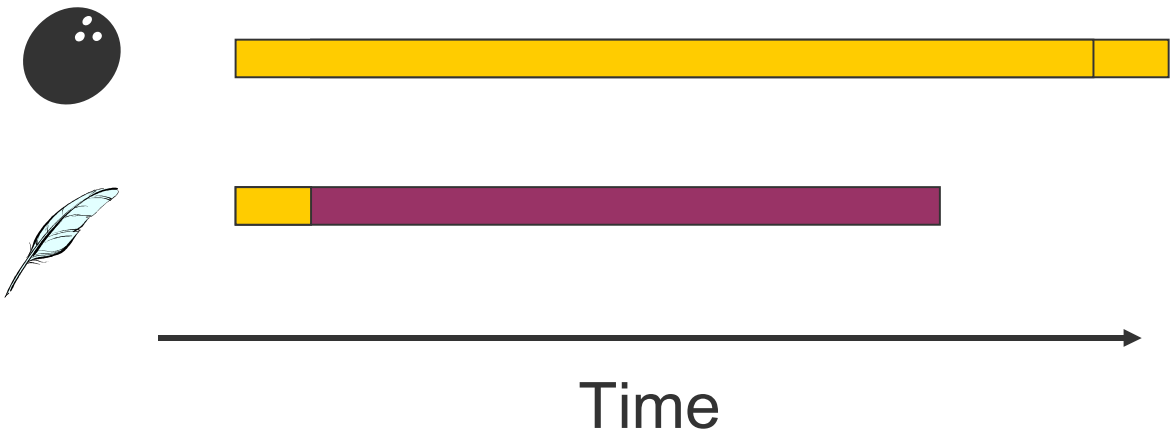
Background

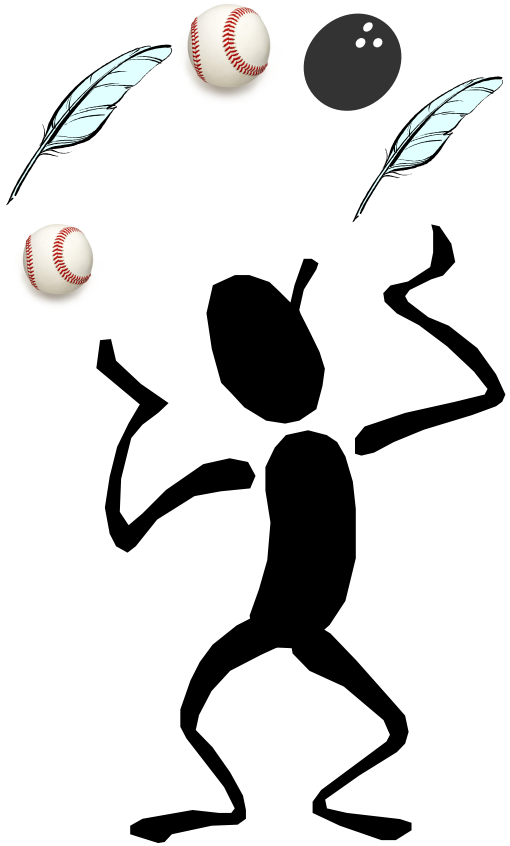
- HP has been building NeoView, a highly-parallel database engine for business intelligence
- Challenges for DBAs
  - How long should they wait to kill an unexpectedly long-running query?
  - When should they admit a newly arriving query if the currently executing batch of queries is in danger of missing its deadline?
  - What if the newly arrived query was submitted by the CEO?

➔ Automate workload management

Why BI Workloads Differ from OLTP Workloads

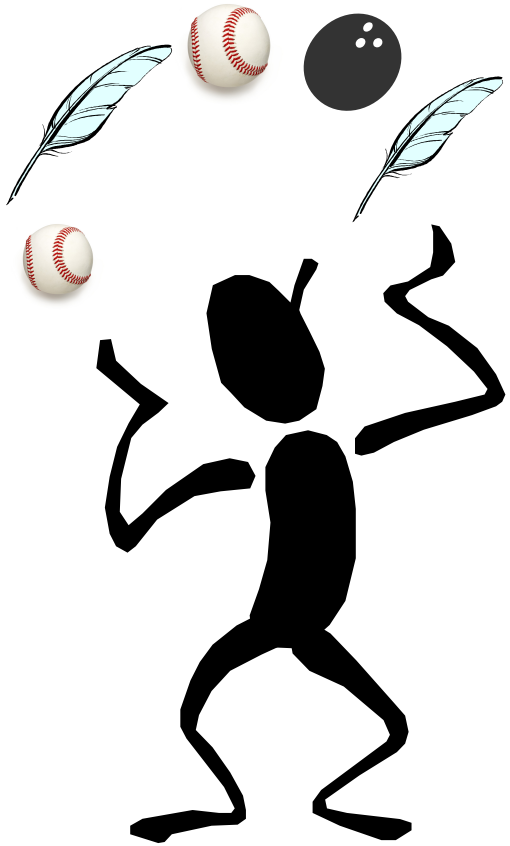
- Complexity
- Resource demands
- Different types of queries
- Unpredictability





Why BI Workloads Differ from OLTP Workloads

- Complexity
- Resource demands
- Different types of queries
- Unpredictability
- Problem queries
- Objectives



Vision: Automate Workload Management

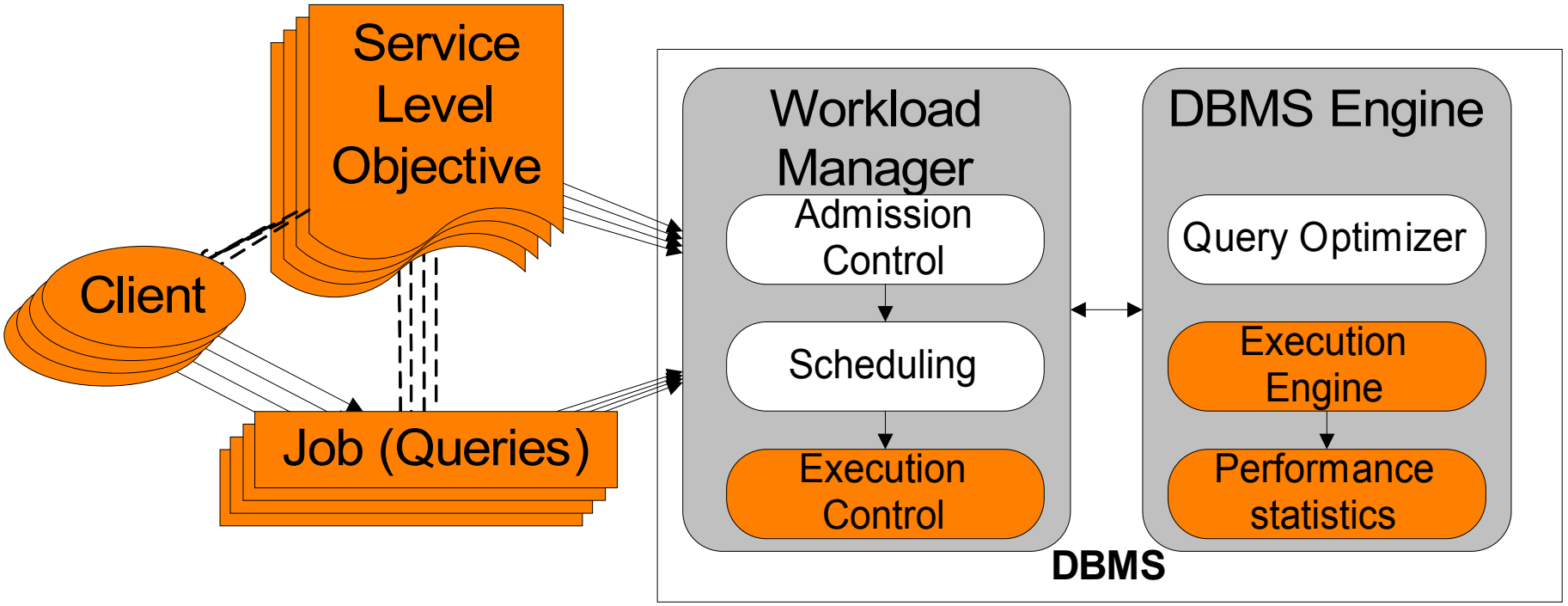
Our approach

- Optimize execution of workload subject to service level objectives
- Explicitly consider “problem” queries as an inherent part of the workload
- Propose an architecture that allows us to …
  - … model problem queries with different characteristics
  - … implement and test workload management actions for dealing with problem queries based on their observed behavior

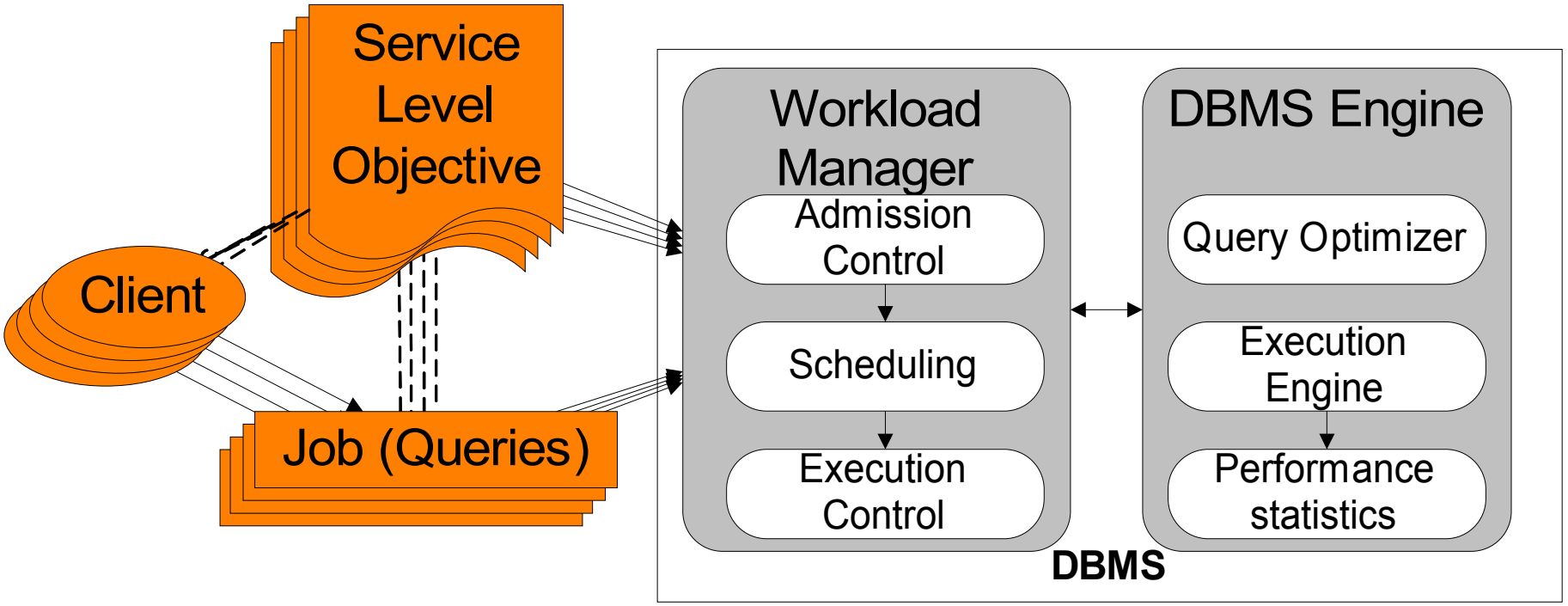
Outline

- Problem statement
- Proposed solution
- Evaluation
  - Implementation and settings for experiments
  - Impact of problem queries on a workload
  - Impact of execution control
- Conclusion and ongoing work

Workload Management Architecture



Service Level Objectives and Jobs



Service Level Objectives (SLOs)

- Job-facing SLOs (e.g., penalty functions used to optimize the scheduling of queries)

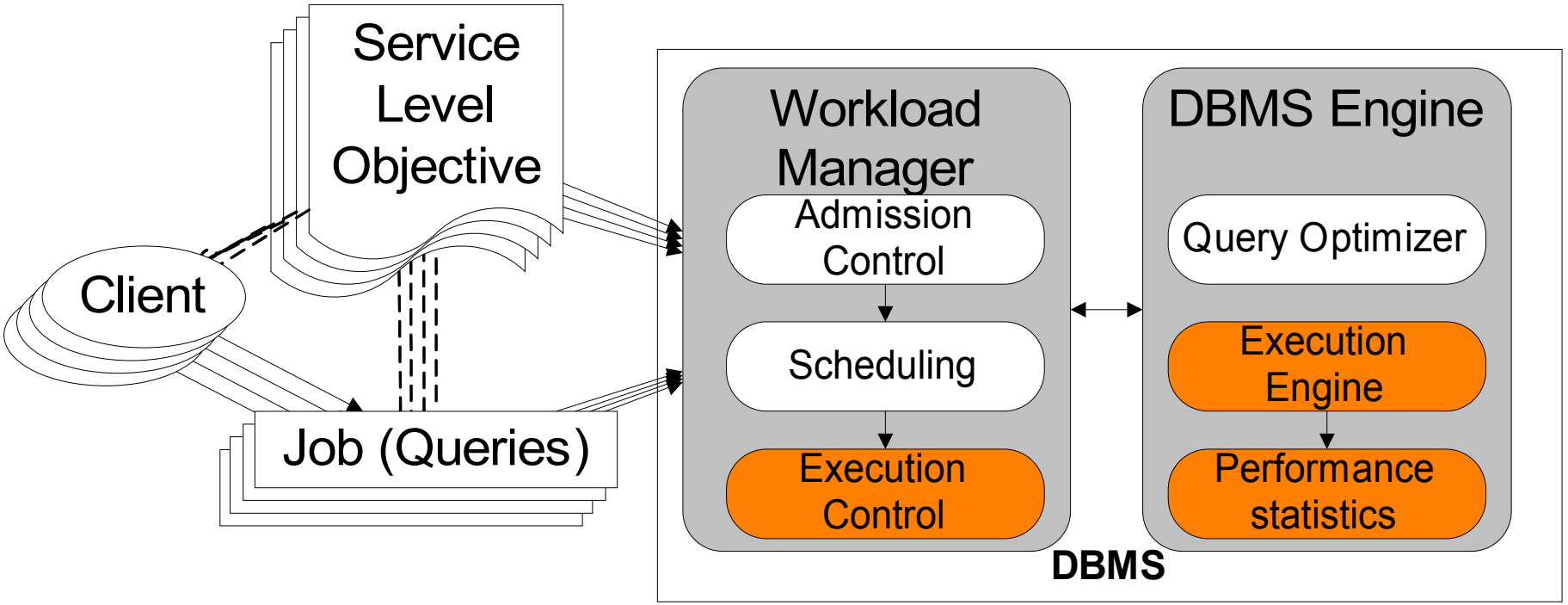
- Customer-facing SLOs
  - Minimize response time (derived from “challenges”)
  - Deadline-driven
  - Concrete quantities of computing time

Job Types

- Batch (e.g., reports)
  - Usually repetitive
  - All queries arrive at the database system at once
  - Queries may/may not have precedence constraints
  - SLO is deadline driven

- Interactive (e.g., business analysis)
  - All queries arrive at the database sequentially
  - Arrival time of the first query is not known in advance
  - SLO (“ASAP”)
    - Submitted by a special request for business reasons

Execution Engine



Workload Manger

- Admission Control
- Scheduling
- **Execution Control**
  - Set of actions that apply when certain conditions hold
  - Example:

```
IF relDBTime IS high AND progress IS low
 THEN cancel IS applicable
```

Workload Manger

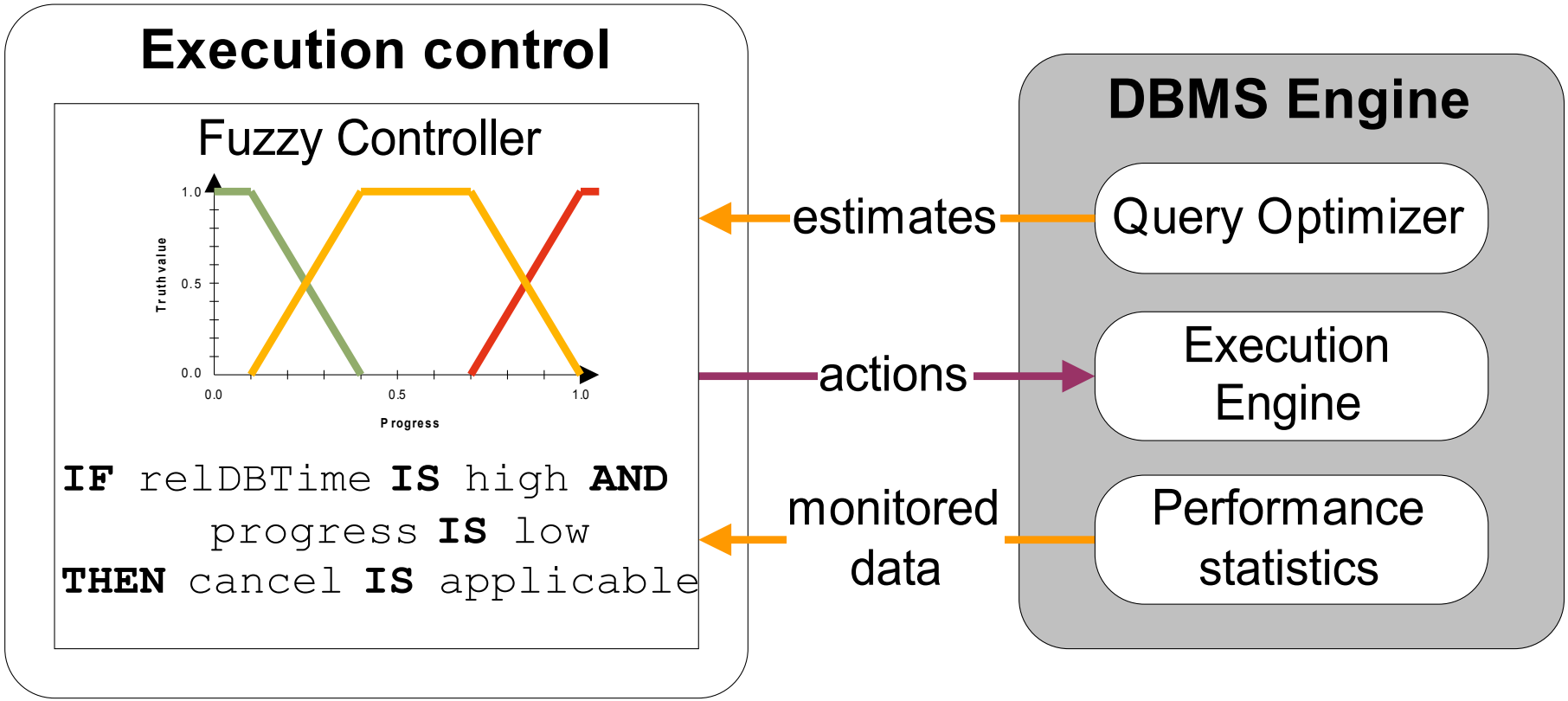
- Admission Control
- Scheduling
- **Execution Control**
  - Set of actions that apply when certain conditions hold
  - Example:

```
IF relDBTime IS high AND progress IS low
 THEN cancel IS applicable
```

Monitored Metrics

- Relative database time (derived from elapsed time of queries and processing time estimates)
- Query progress (derived from progress indicator)
- Number of cancellations
- Resource contention
- Priority

Monitored Metrics

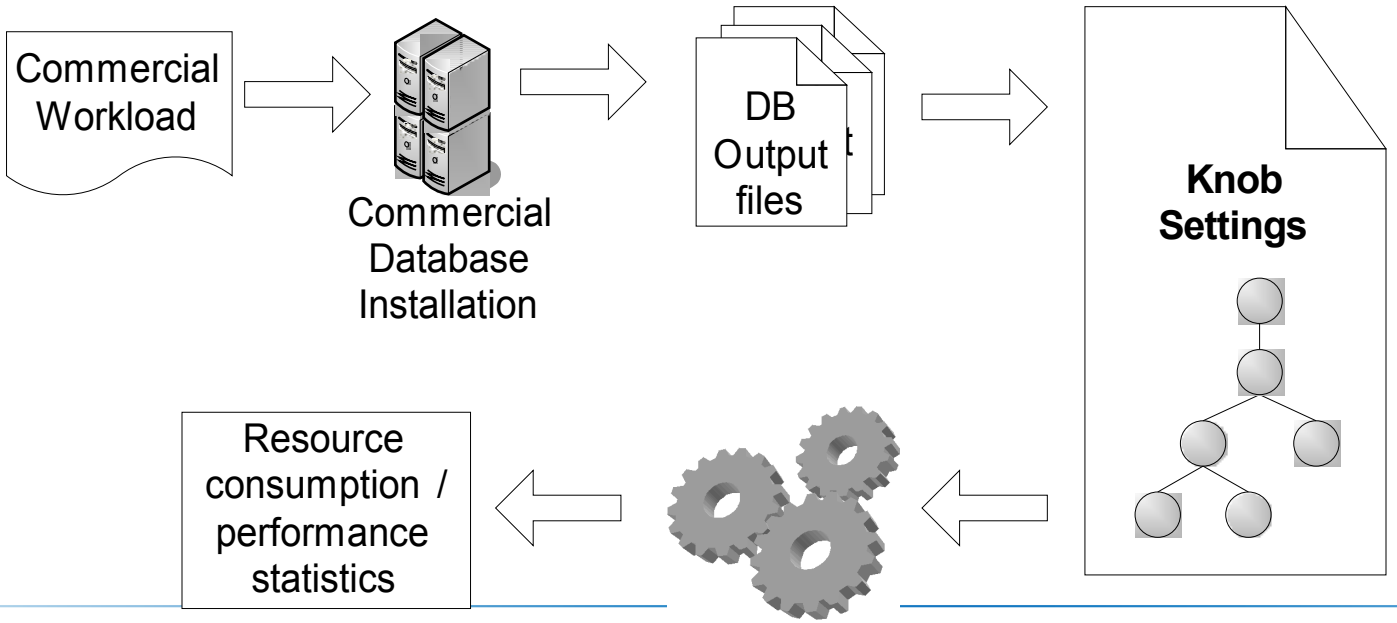


Outline

- Problem statement
- Proposed solution
- Evaluation
  - Implementation and settings for experiments
  - Impact of problem queries on a workload
  - Impact of execution control
- Conclusion and ongoing work

Implementation

- Use simulated execution engine instead of real database system installation
  - Inject problem queries
  - Real workloads can take days to process



- Number of queries in a job
- Number of jobs in a workload
- Number of problem queries
- …

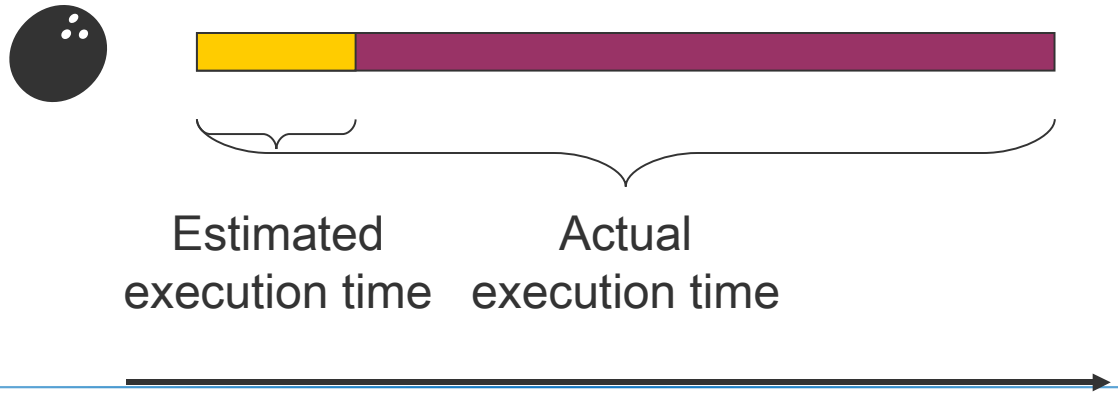
Settings for Experiments

- Interactive job
  - ~ 1100 feathers
  - Queries arrive sequentially
    - Inter-arrival time 0
    - Does not span entire workload interval
- Batch job
  - ~ 1700 feathers, baseballs, and bowling balls
  - Average execution time of batch queries ~1000 times higher than execution time of interactive queries

derived from commercial workload runs

Settings for Experiments

- Normal workload
  - Interactive and batch job executed in parallel
  - No problem queries
- Problem workload
  - Interactive and batch job executed in parallel
  - **Problem queries** injected into batch workload (75 queries with different “stretch factors”)

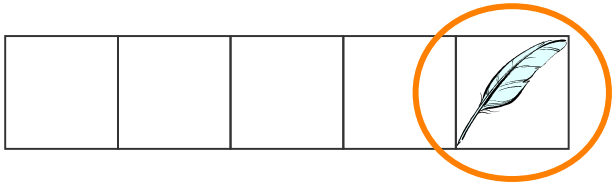


Settings for Experiments

- Normal workload
  - Interactive and batch job executed in parallel
  - No problem queries
- Problem workload
  - Interactive and batch job executed in parallel
  - **Problem queries** injected into batch workload (75 queries with different “stretch factors”)
  - Problem queries have a probability for showing the problem behavior after restarting them
- Admit interactive queries first

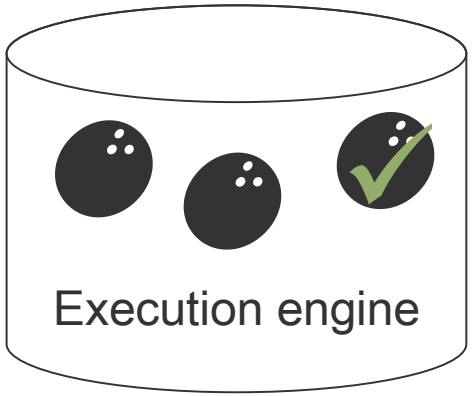
Admission Control: Admit Interactive First

Queue for interactive queries



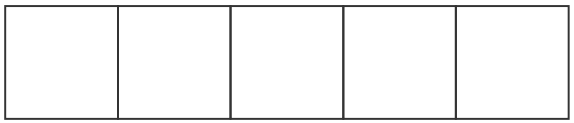
Admit query

Queue for batch queries

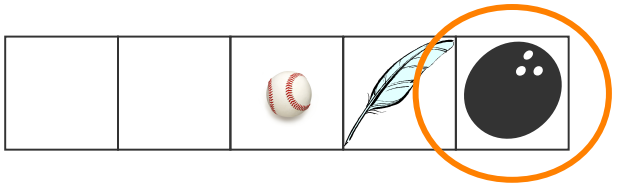


Admission Control: Admit Interactive First

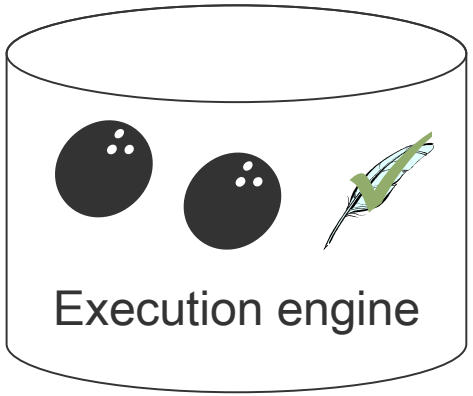
Queue for interactive queries



Queue for batch queries



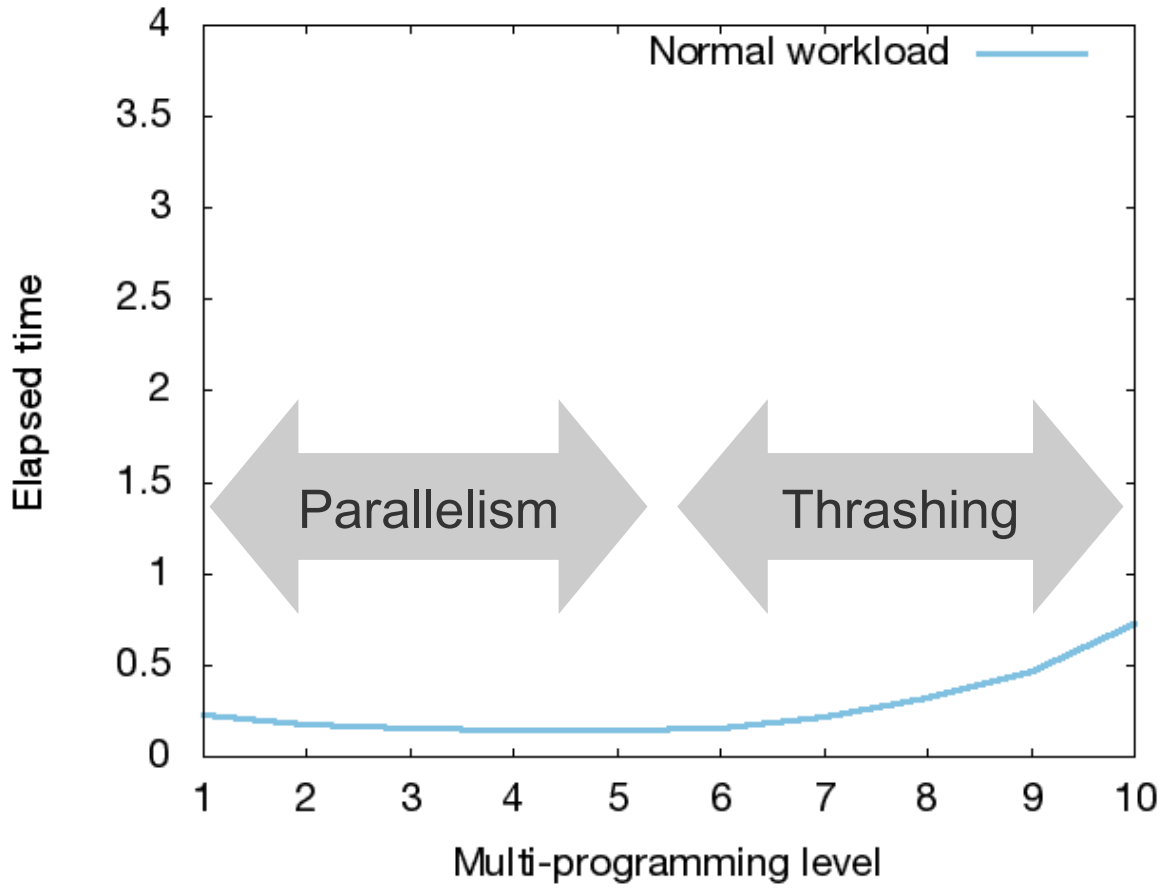
Admit query



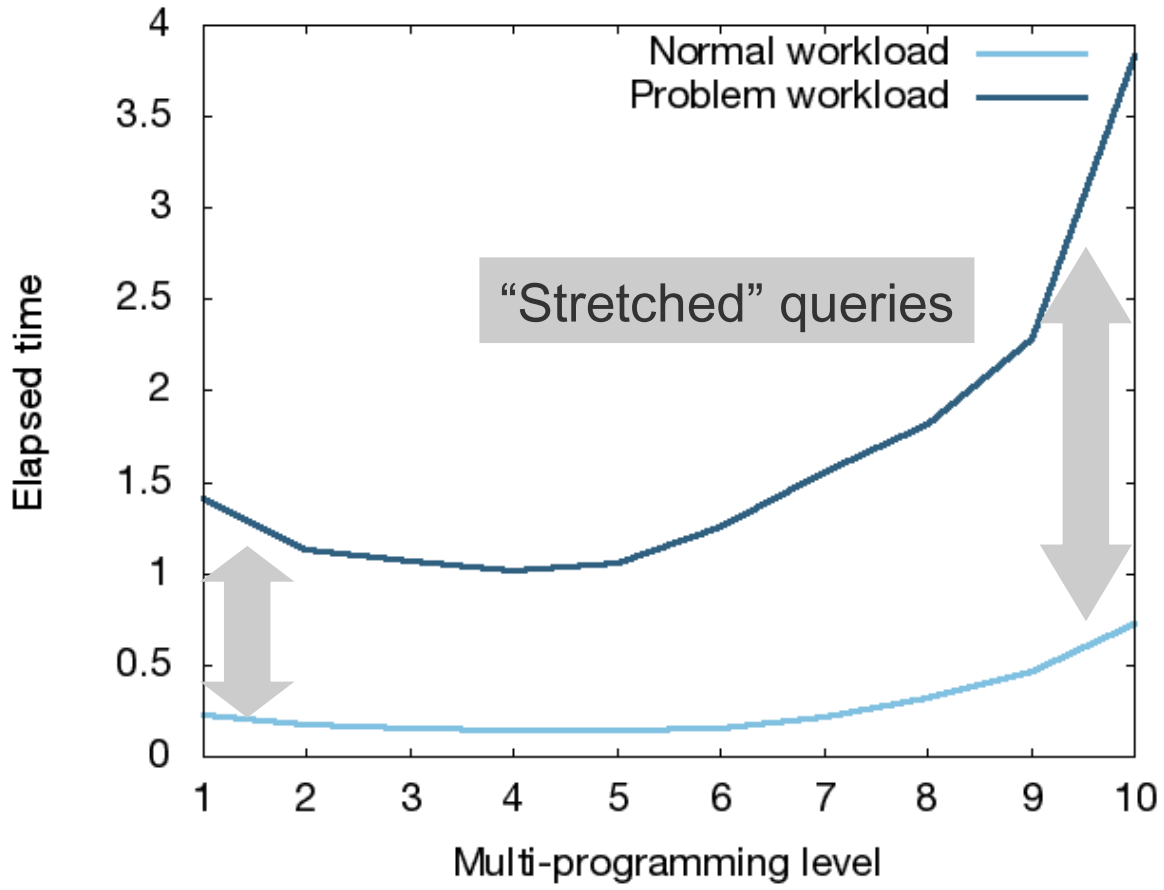
Outline

- Problem statement
- Proposed solution
- Evaluation
  - Implementation and settings for experiments
  - Impact of problem queries on a workload
  - Impact of execution control
- Conclusion and ongoing work

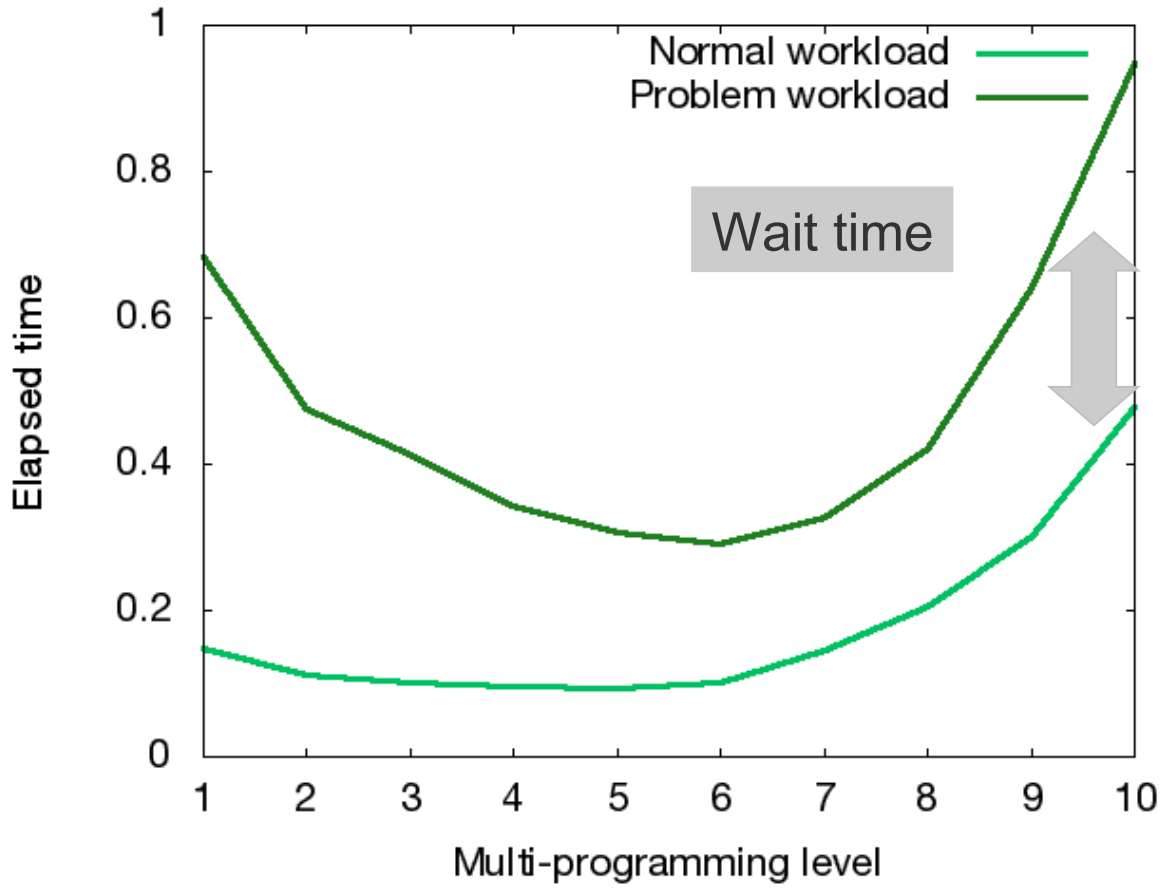
Impact of Problem Queries on Batch Job



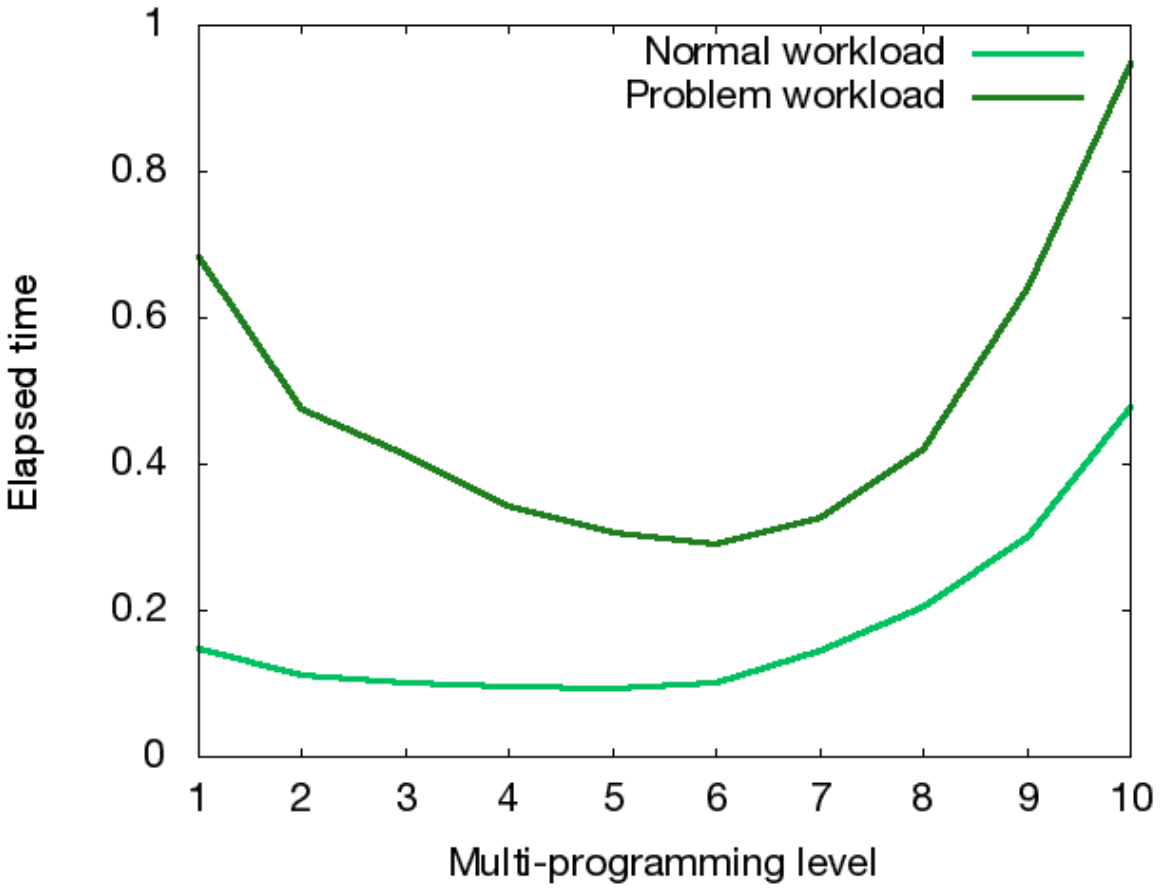
Impact of Problem Queries on Batch Job

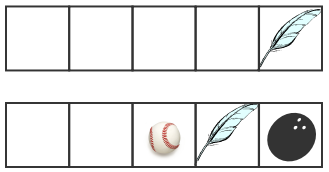


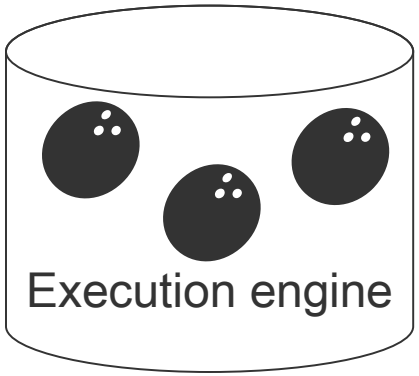
Impact of Problem Queries on Interactive Job



Impact of Problem Queries on Interactive Job







Outline

- Problem statement
- Proposed solution
- Evaluation
  - Implementation and settings for experiments
  - Impact of problem queries on a workload
  - Impact of execution control
- Conclusion and ongoing work

Workload Management Policies

- Fix the MPL at 5
- Varying aggressiveness
  - If query exceeds estimated database time, take action

```
relative database time=
  actual database time/estimated database time
```

  - If query is almost finished, do not execute action
- Queries identified as problems are killed and immediately resubmitted (“cancel”)
- Canceled queries get two more chances to run to completion
- If queries do not complete, they are killed (“aborted”)

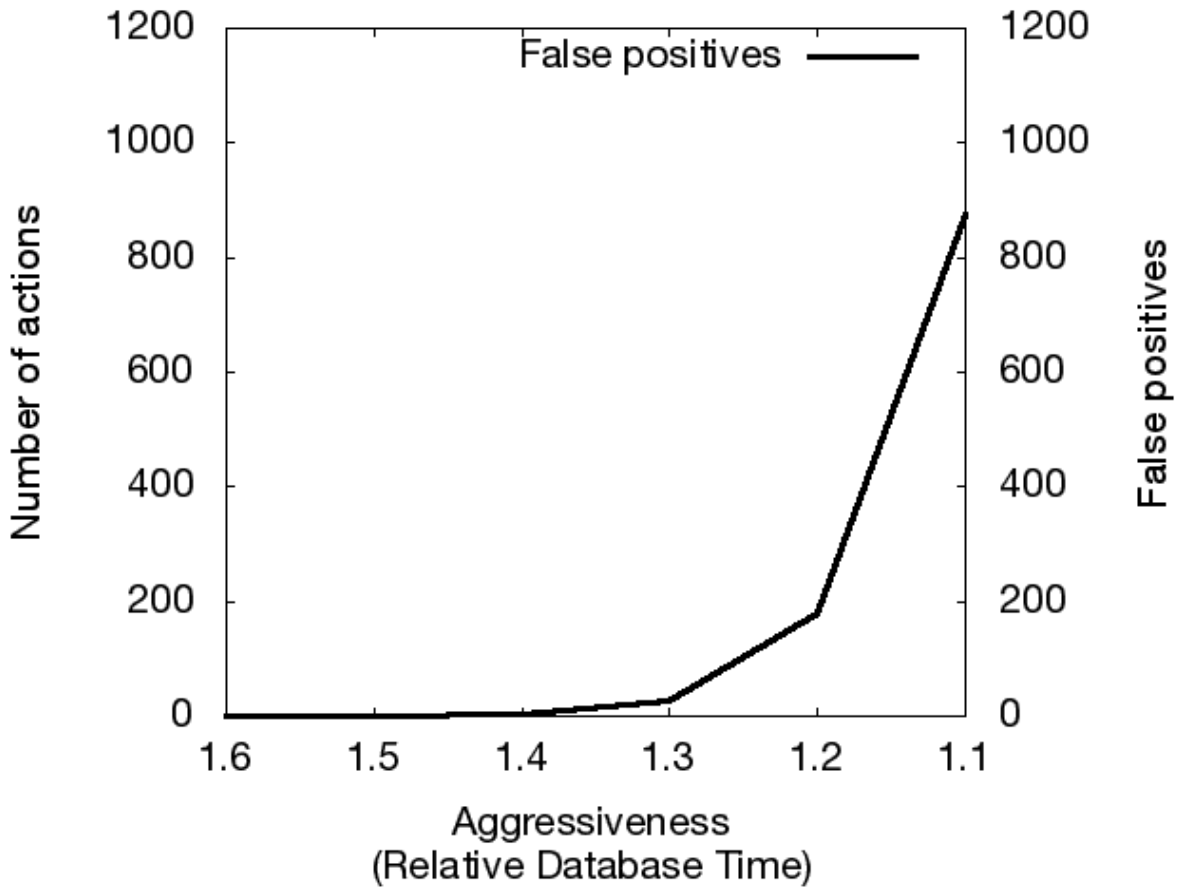
Impact of Workload Management Actions

- Batch job: Reduce elapsed time by 81% (problem queries)
- Interactive job: Reduce wait time by 67% (wait time)

- But…

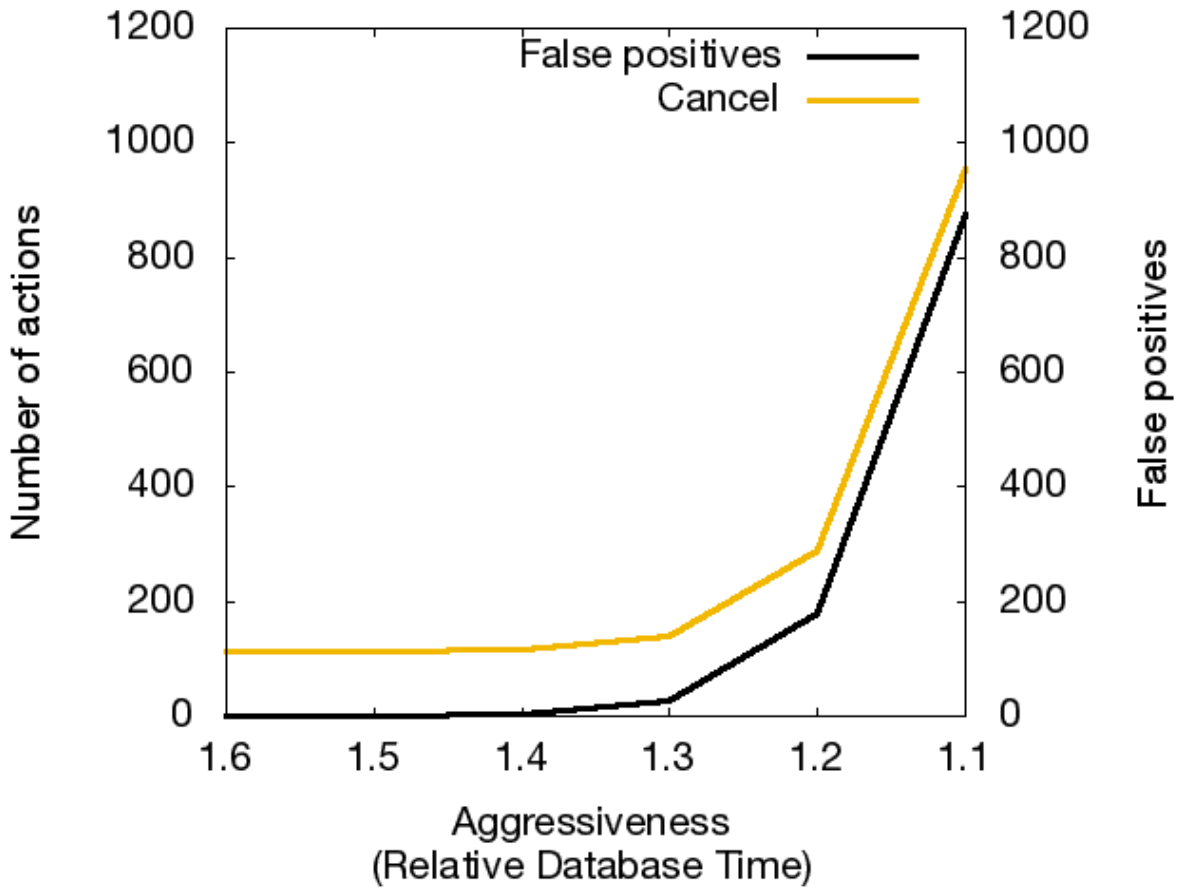
False Positives Lead to Unnecessary Actions

Relative Database Time



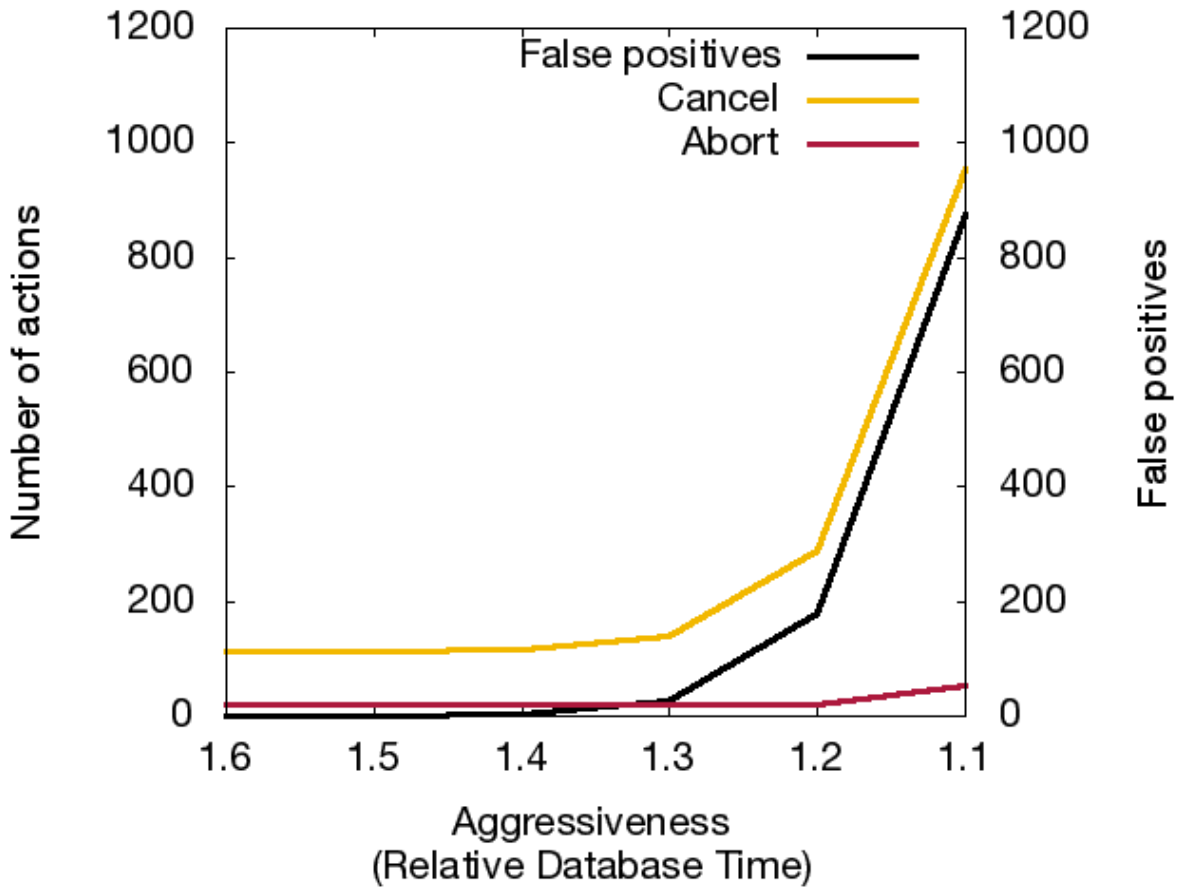
False Positives Lead to Unnecessary Actions

Relative Database Time



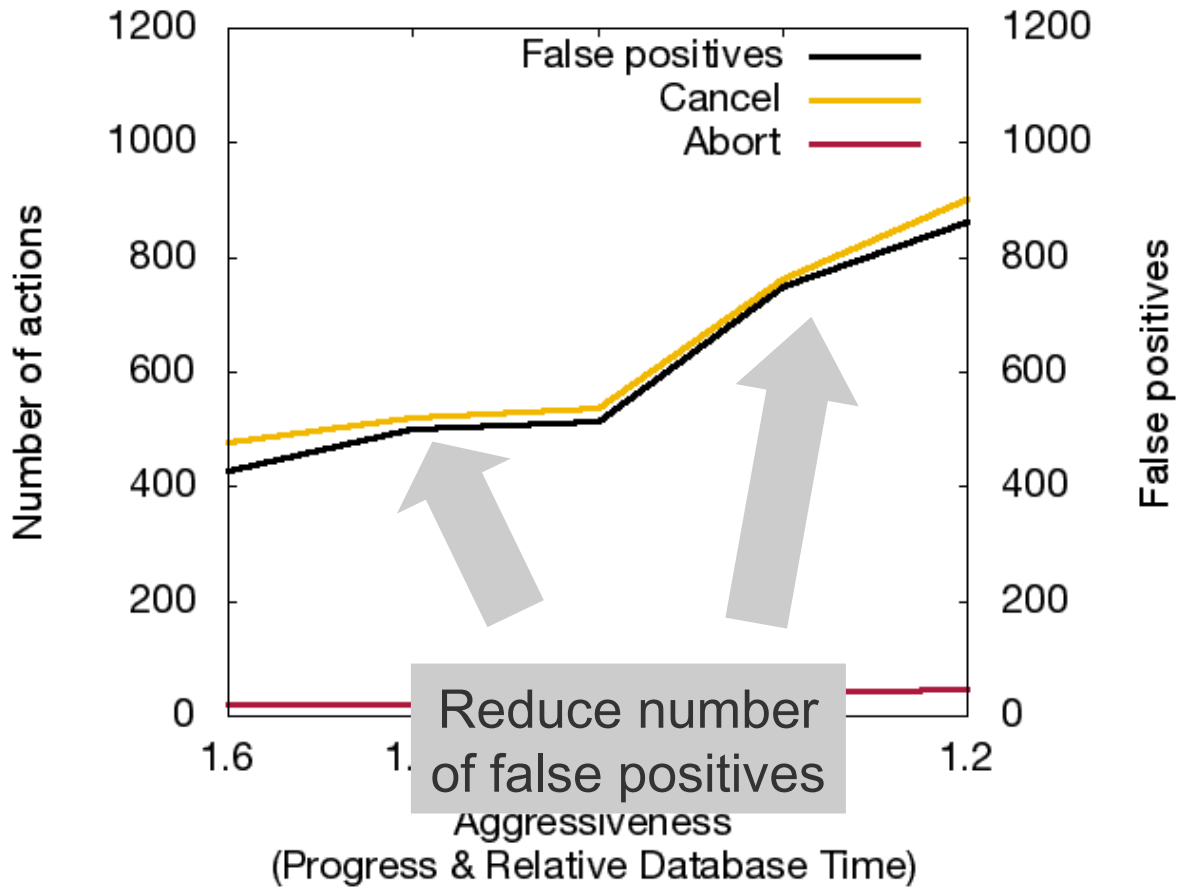
False Positives Lead to Unnecessary Actions

Relative Database Time

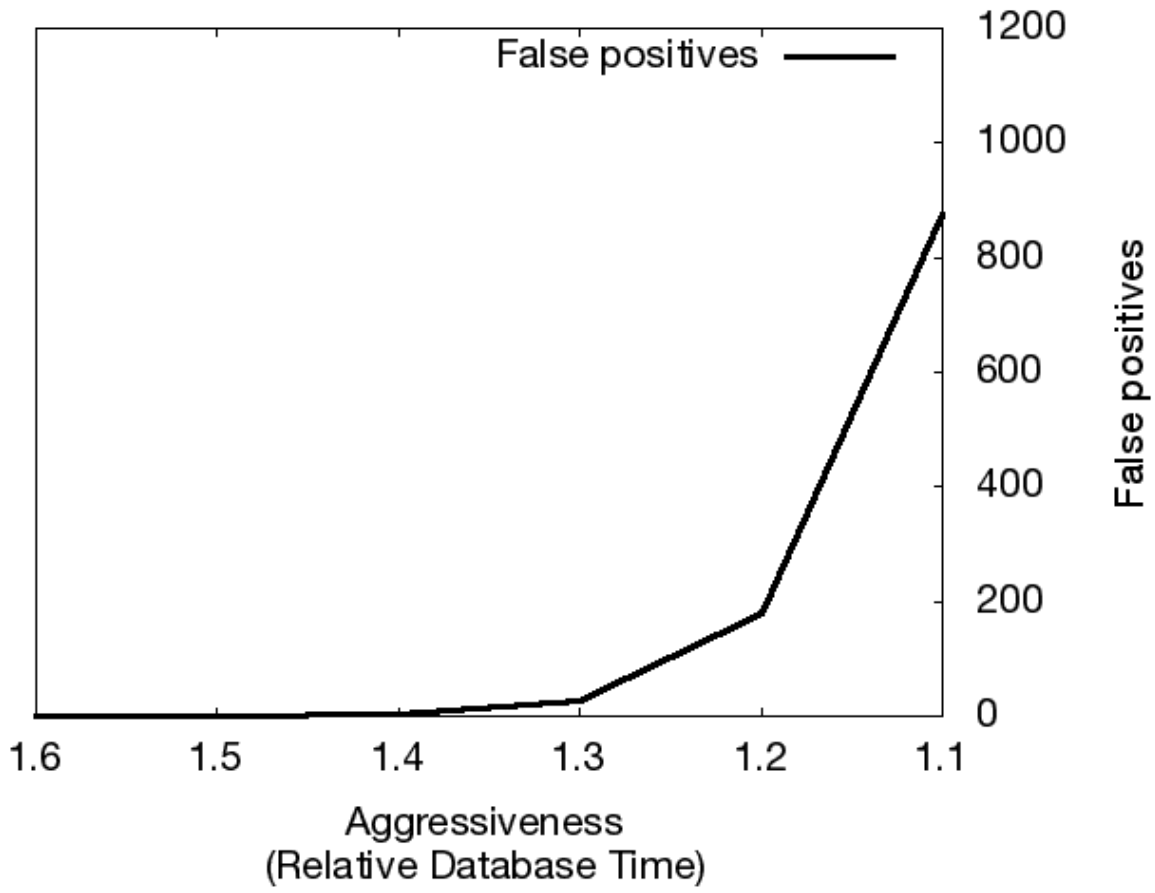


Number of False Positives and Actions Executed

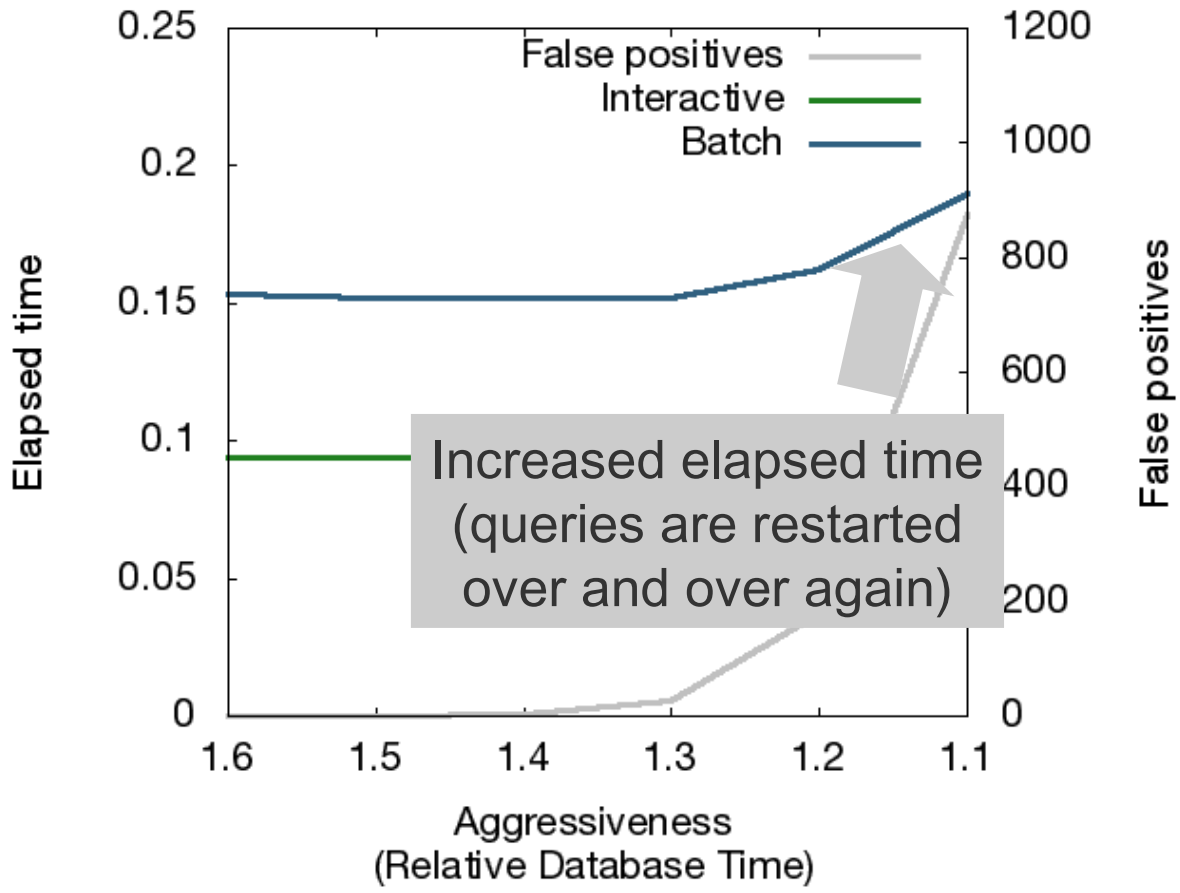
Progress



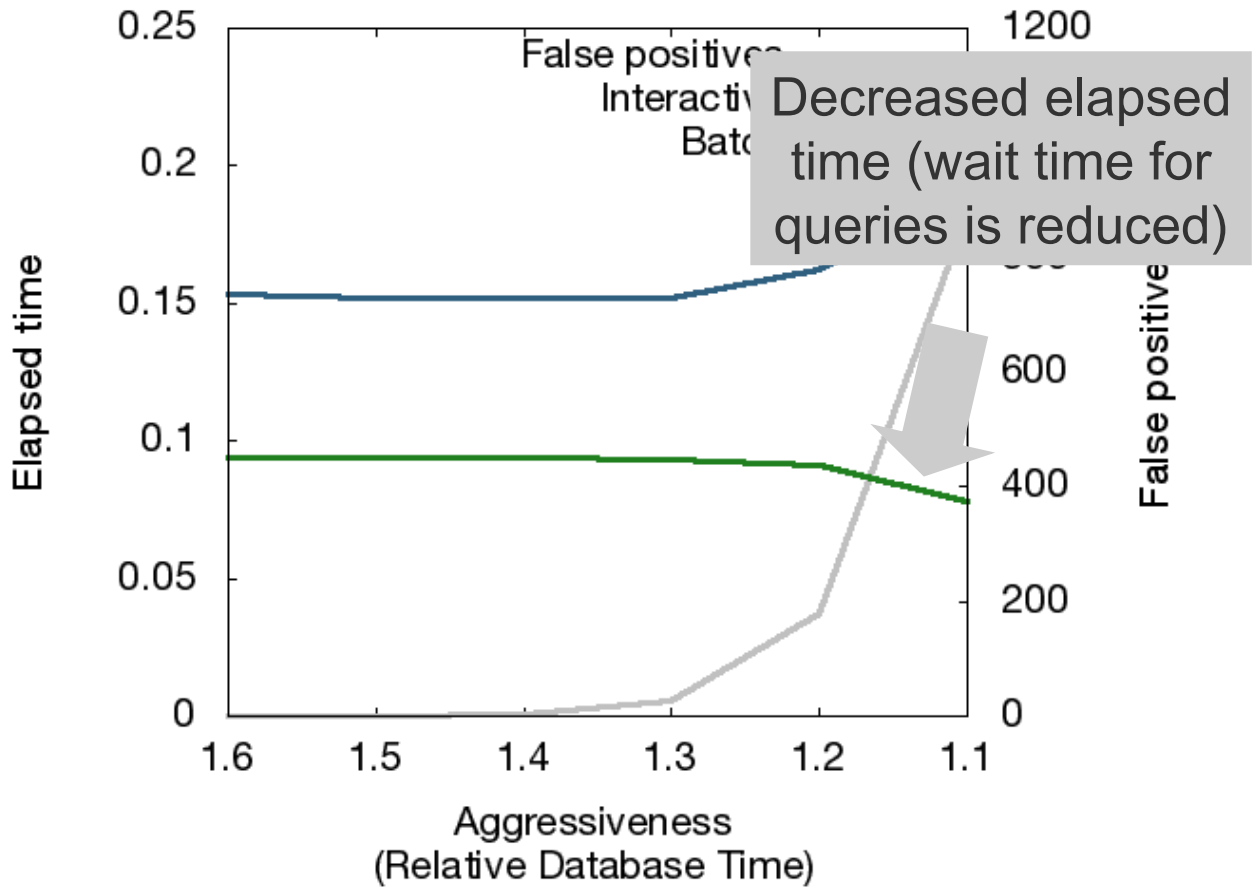
Elapsed Time for Batch and Interactive Jobs



Elapsed Time for Batch and Interactive Jobs



Elapsed Time for Batch and Interactive Jobs



Outline

- Problem statement
- Proposed solution
- Evaluation
  - Implementation and settings for experiments
  - Impact of problem queries on a workload
  - Impact of execution control
- Conclusion and ongoing work

Conclusion

- We implemented a workload management test bed
- Our experiments show that …
  - … even few problem queries have a significant impact on the execution of a mixed workload
  - … the number of false positives leads to an increase in execution time
- Lessons we learned
  - Applying actions too aggressively leads to unnecessary actions
  - Use controller and adjust parameters to right level of aggression

Ongoing Work

- Evaluate impact of admission control and scheduling of BI workloads
- Model query execution on a more detailed level
- Model additional problem types
- Evaluate new workload management techniques

Any Questions?