# Toward Green Data Lake Management and Analysis through a CTMC Model

Marzieh Derakhshannia[1], Julien Grange[2], Nihal Pekergin[2]

[1]IRIT, Université Toulouse III-Paul Sabatier, France
[2]LACL, Université Paris-Est Créteil, France

VECoS'24
October 15th-19th, 2024

# Motivation

Data lakes:

- **heterogeneous** data management platforms

# Motivation

Data lakes:

- **heterogeneous** data management platforms
- power intensive

# Motivation

Data lakes:

- **heterogeneous** data management platforms
- power intensive

**Goal**

We want to strike a trade-off between

- power consumption
- quality of service, i.e. data absorption and query response time

# Modeling

Data lake abstraction:

- a set of nodes

**Assumption**

All nodes are assumed identical

# Modeling

Data lake abstraction:

- a set of nodes
- each node can service up to some fixed number of jobs

# Modeling

Data lake abstraction:

- a set of nodes
- each node can service up to some fixed number of jobs

Each node can be in one of three regimes:

($R_0$) Stand-by regime

($R_1$) Low regime

($R_2$) High regime

with increasing

- storage bandwidth
- computing power
- power consumption

# Modeling

Two events:

- query arrival

**Assumption**

Each query gives rise to some number of jobs
We have analyzed datasets of typical queries to get a distribution of query complexity (number of jobs = number of operations)

# Modeling

Two events:

- query arrival
- job completion

**Assumption**

All jobs are identical
We don't record a connection between a job and the query it comes from

# Modeling

Two events:

- query arrival
- job completion

**Assumption**

Each kind of event occurs

- identically and independently
- with a time-homogeneous exponential rate

$\rightarrow$ we can use a **CTMC** (continuous-time Markov chain) model

# Modeling

Internal state of the CTMC at time $t$:

$J(t)$: number of active jobs

$R_0(t), R_1(t), R_2(t)$: number of nodes in regime $(R_0), (R_1), (R_2)$

# Modeling

Internal state of the CTMC at time $t$:

$J(t)$: number of active jobs

$R_0(t), R_1(t), R_2(t)$: number of nodes in regime $(R_0), (R_1), (R_2)$

**Assumption**

The rate of job completion takes into account

- the number and regime of nodes
- the number of jobs
- job contention when several jobs run on a single node

# Strategies

Two **pure** strategies: **left** and **right**

When the capacity is exceeded, preferentially...

**(left)** ...push a node from ($R_0$) to ($R_1$)

**(right)** ...push a node from ($R_1$) to ($R_2$)

# Strategies

Two **pure** strategies: **left** and **right**

When the capacity is exceeded, preferentially...

**(left)** ...push a node from $(R_0)$ to $(R_1)$

**(right)** ...push a node from $(R_1)$ to $(R_2)$

We consider **probabilistic** strategies

# Numerical evaluation

For each strategy we evaluate

- expected number of jobs

in the steady-state distribution of the CTMC

**Fact**

When the data lake is not overloaded, the expected number of jobs accurately represents the expected response time per job

# Numerical evaluation

For each strategy we evaluate

- expected number of jobs
- expected query loss

in the steady-state distribution of the CTMC

# Numerical evaluation

For each strategy we evaluate

- expected number of jobs
- expected query loss
- energy consumption

in the steady-state distribution of the CTMC

# Numerical evaluation

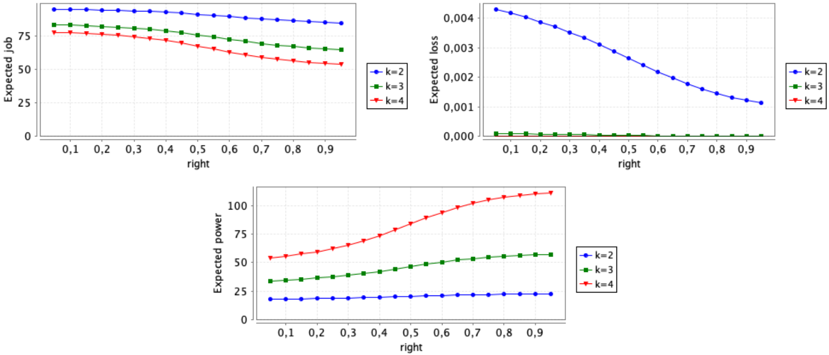
For each strategy we evaluate

- expected number of jobs
- expected query loss
- energy consumption

in the steady-state distribution of the CTMC

We used the probabilistic model checker PRISM to construct and analyse our CTMCs

# Numerical evaluation




Performance measures in the steady-state

$k$: ratio between frequency of ($R_2$) and ($R_1$)

Power consumption ratio between ($R_2$) and ($R_1$) is cubic in $k$ (DVFS)

# Conclusion

We have

1. modeled performance and power consumption in data lakes via CTMCs
2. proposed a family of strategies to strike a balance between both
3. implemented a way to evaluate experimentally these strategies

# Conclusion

We have

1. modeled performance and power consumption in data lakes via CTMCs
2. proposed a family of strategies to strike a balance between both
3. implemented a way to evaluate experimentally these strategies

**Futur work**

Go from CTCM to *Markov decision processes*, to optimize directly on the decision variables