Tiresias:
Enabling Predictive Autonomous Storage and Indexing

Michael Abebe mtabebe@uwaterloo.ca
Horatiu Lazu
Khuzaima Daudjee

September 2022

tiny.cc/tiresias

UNIVERSITY OF WATERLOO

DBMS **storage** & **indexing choices**

have **trade-offs** based on **workload**

UNIVERSITY OF WATERLOO

Index Choice

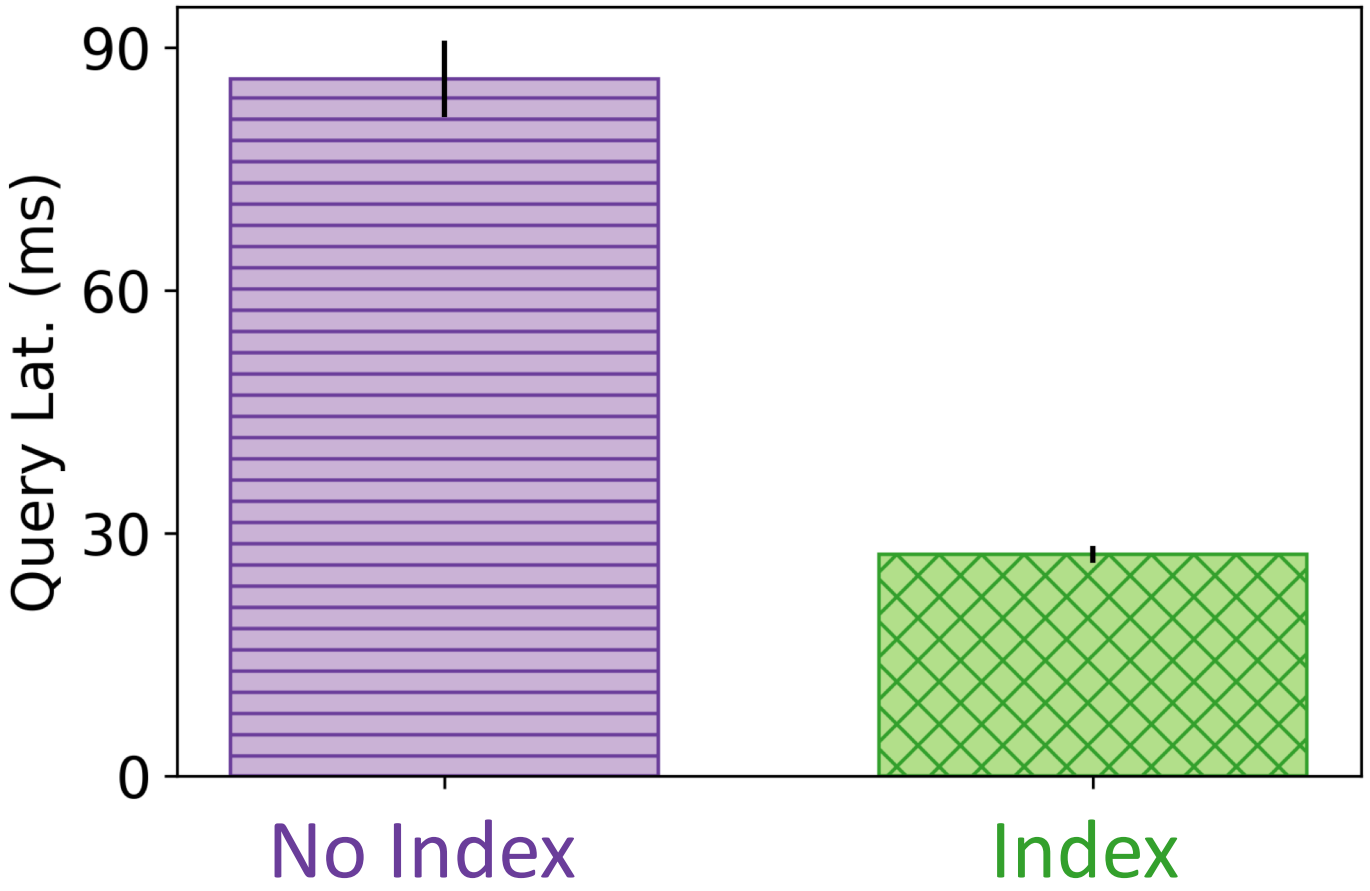
Order	Book	User	Qnt	$
100	HP	Alice	7	$15
101	Drac	Ted	90	$3
102	SQL	Geoff	1	$100
103	HP	Jean	3	$15

SELECT * WHERE
QNT > 15

Should we index?

Index Choice

Lower is better



SELECT * WHERE
QNT > 15

Should we index?

Yes

UNIVERSITY OF WATERLOO

Index Choice

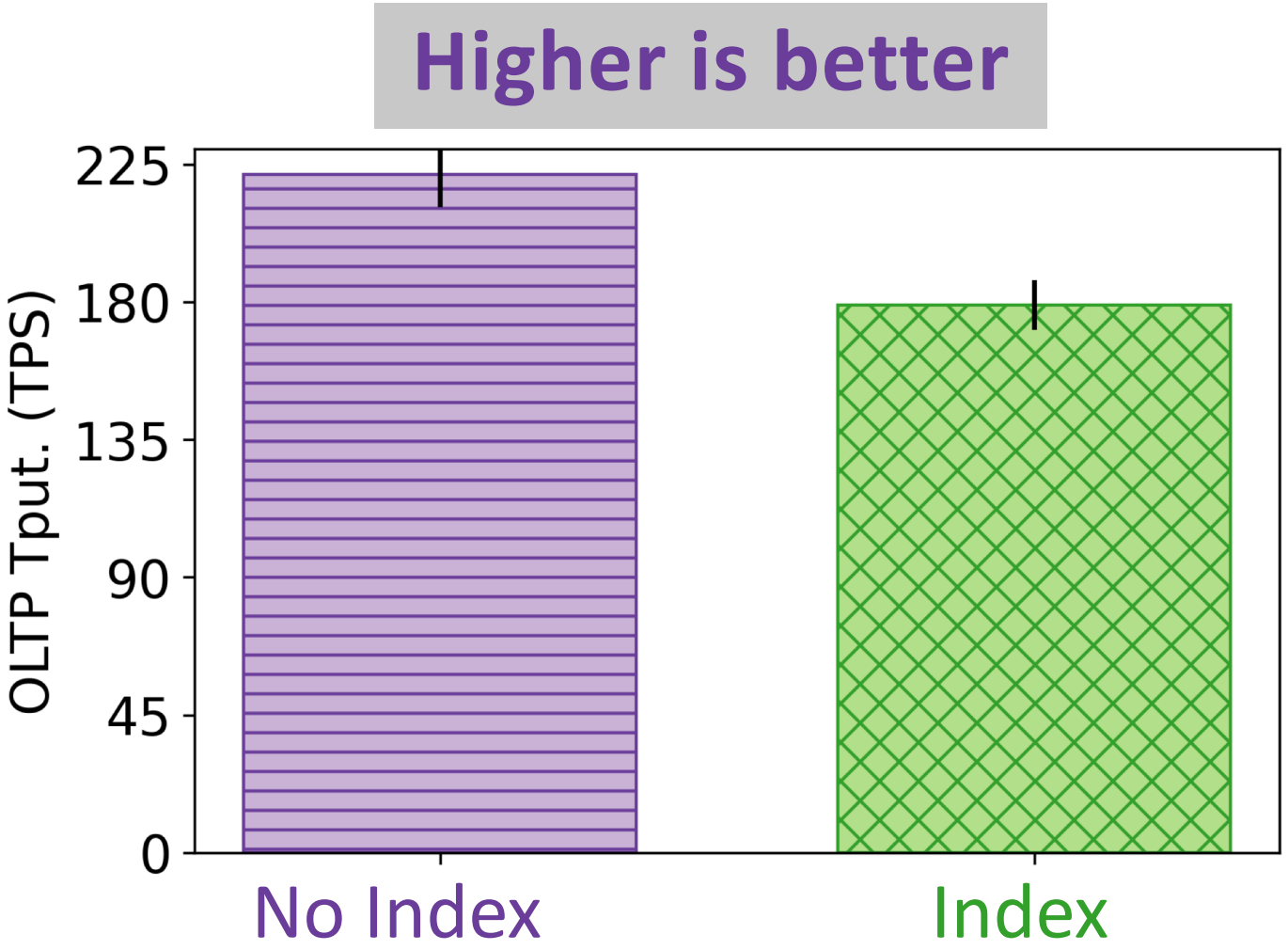
Order	Book	User	Qnt	$
100	HP	Alice	7	$15
101	Drac	Ted	90	$3
102	SQL	Geoff	1	$100
103	HP	Jean	3	$15
104	LoTR	John	2	$21

INSERT (104, LoTR, John, 2, $21)

Should we index?

UNIVERSITY OF WATERLOO

Index Choice



INSERT (104, LoTR, John, 2, $21)

Should we index?

No

Storage Choice

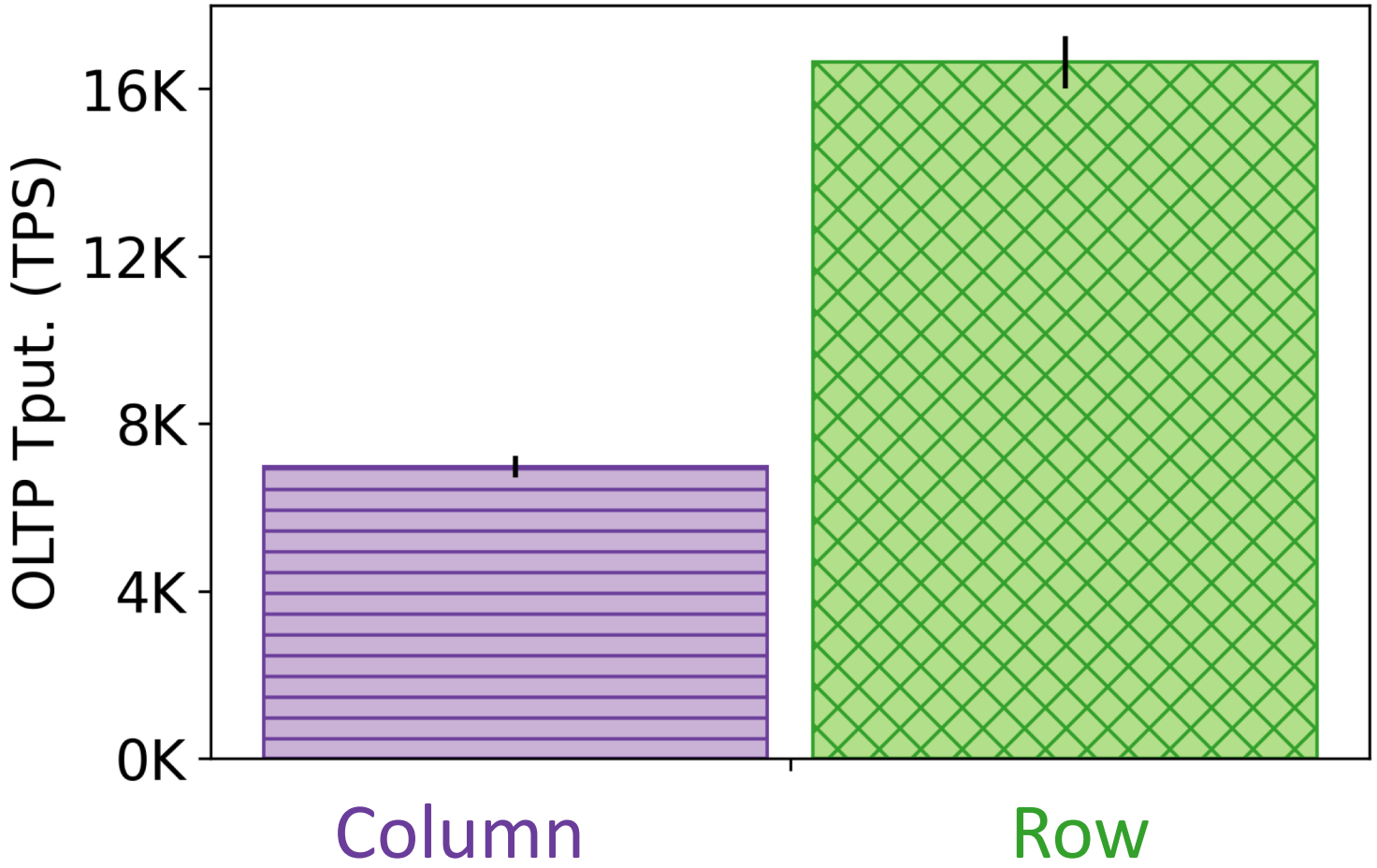
Order	Book	User	Qnt	$
100	HP	Alice	7	$15
101	Drac	Ted	90	$3
102	SQL	Geoff	1	$100
103	HP	Jean	3	$15
104	LoTR	John	2	$21

INSERT (104, LoTR, John, 2, $21)

UNIVERSITY OF WATERLOO

Storage Choice

Higher is better



INSERT (104, LoTR, John, 2, $21)

How should we store?

Row

UNIVERSITY OF WATERLOO

Storage Choice

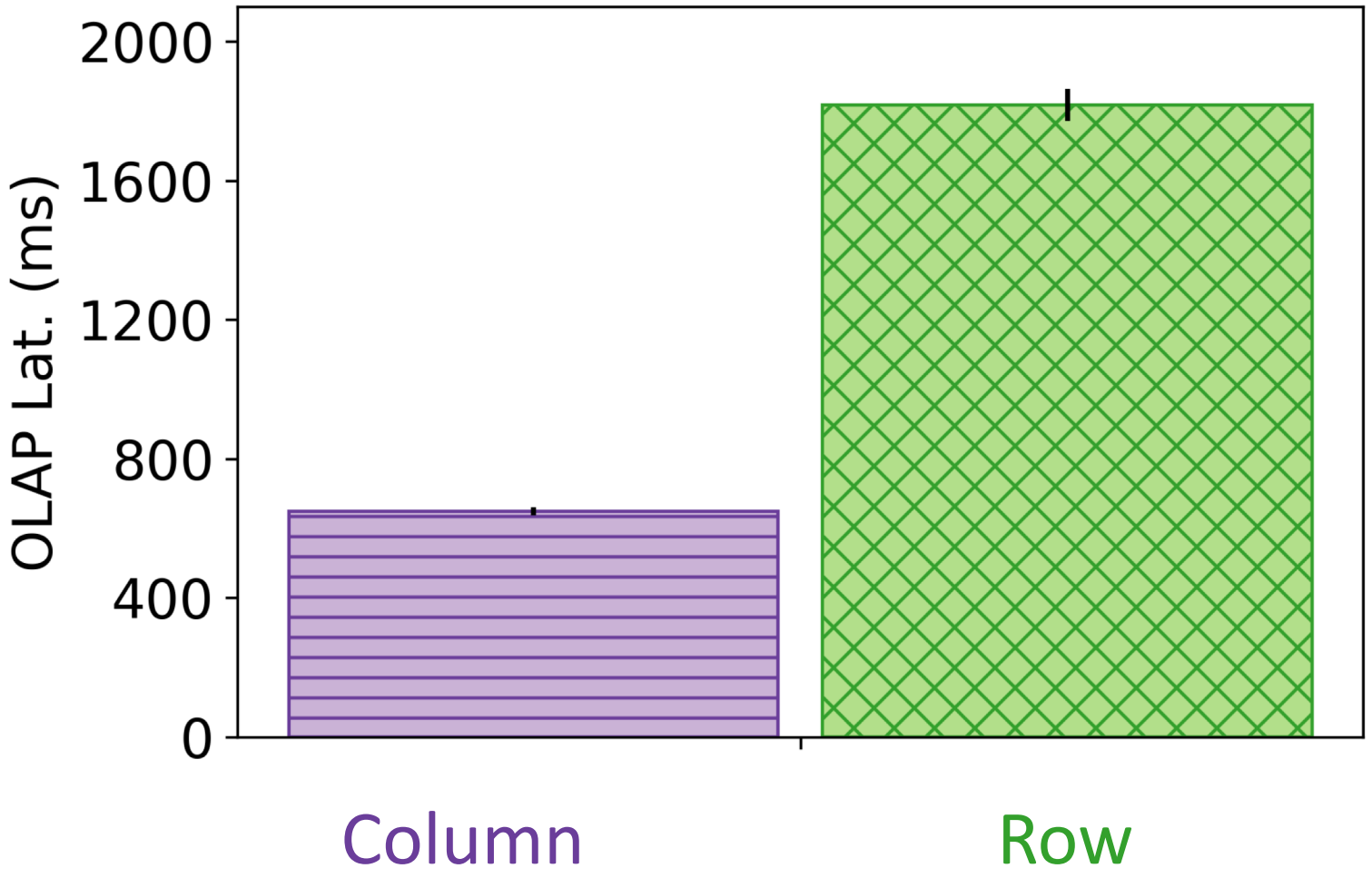
Order	Book	User	Qnt	$
100	HP	Alice	7	$15
101	Drac	Ted	90	$3
102	SQL	Geoff	1	$100
103	HP	Jean	3	$15
104	LoTR	John	2	$21

SELECT * **WHERE**
QNT > 15

UNIVERSITY OF WATERLOO

Storage Choice

Lower is better



SELECT * **WHERE**
QNT > 15

How should we store?

Column

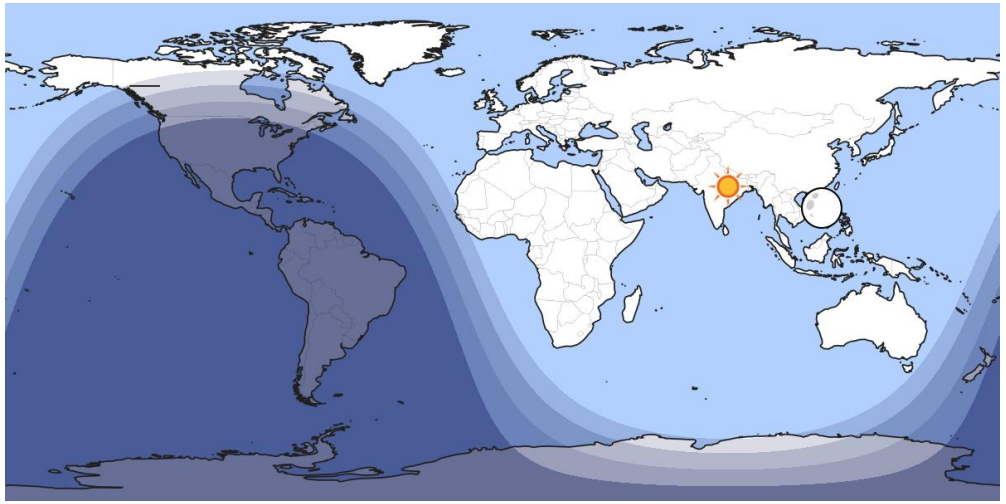
UNIVERSITY OF WATERLOO

DBMS **storage** & **indexing choices**

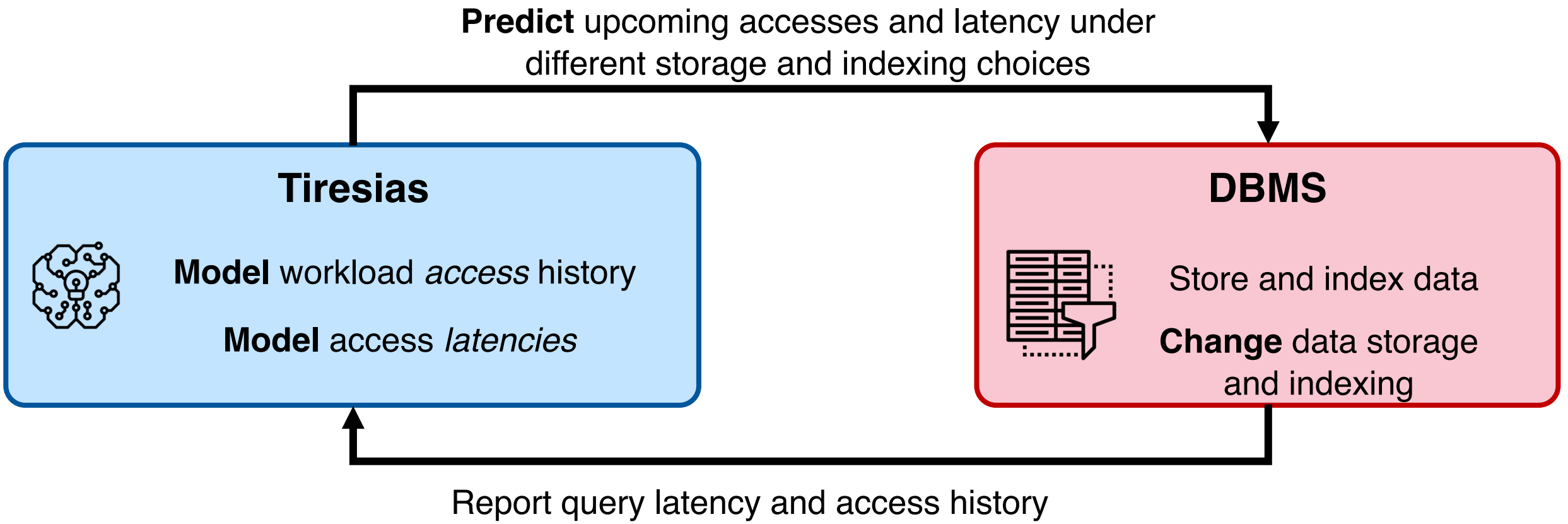
have **trade-offs** based on **workload**

so choices should be **adaptive**!

Workloads Change

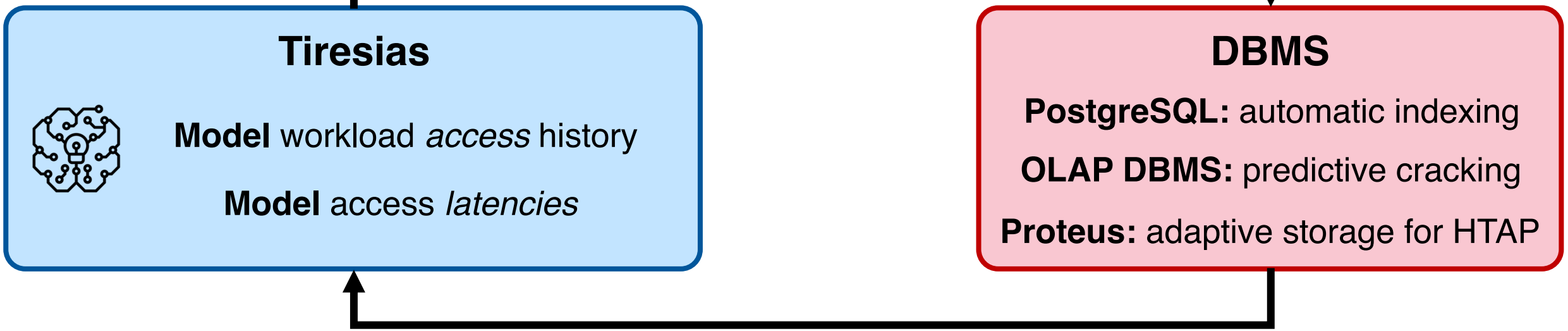


Tiresias


Predict upcoming accesses and latency under different storage and indexing choices
Tiresias
Model workload *access* history
Model access *latencies*
DBMS
Store and index data
Change data storage and indexing
Report query latency and access history

Tiresias

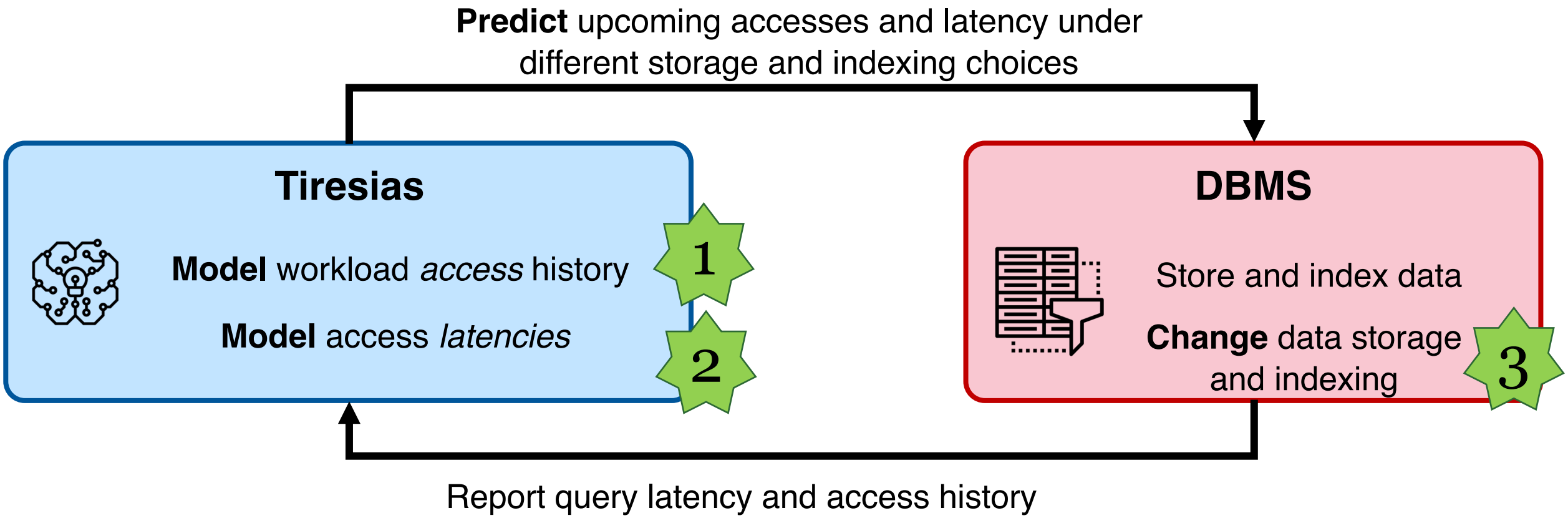
Predict upcoming accesses and latency under different storage and indexing choices



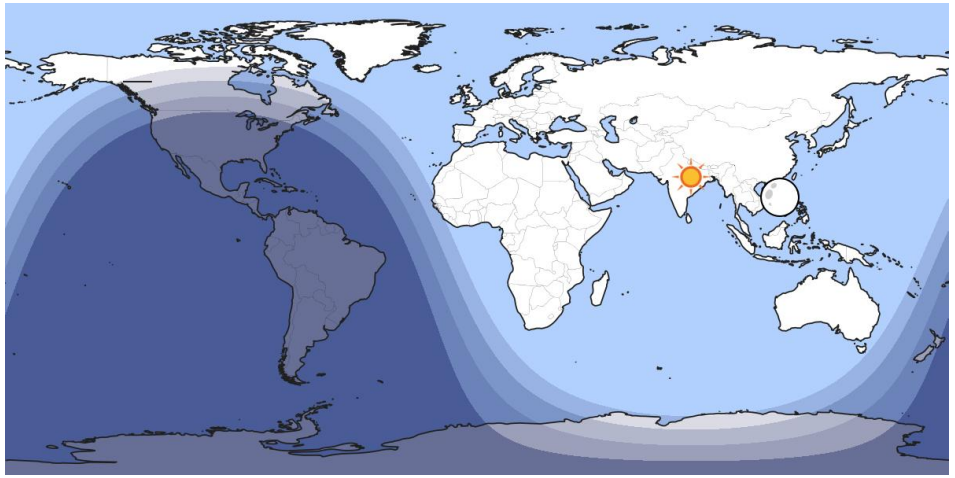
Report query latency and access history

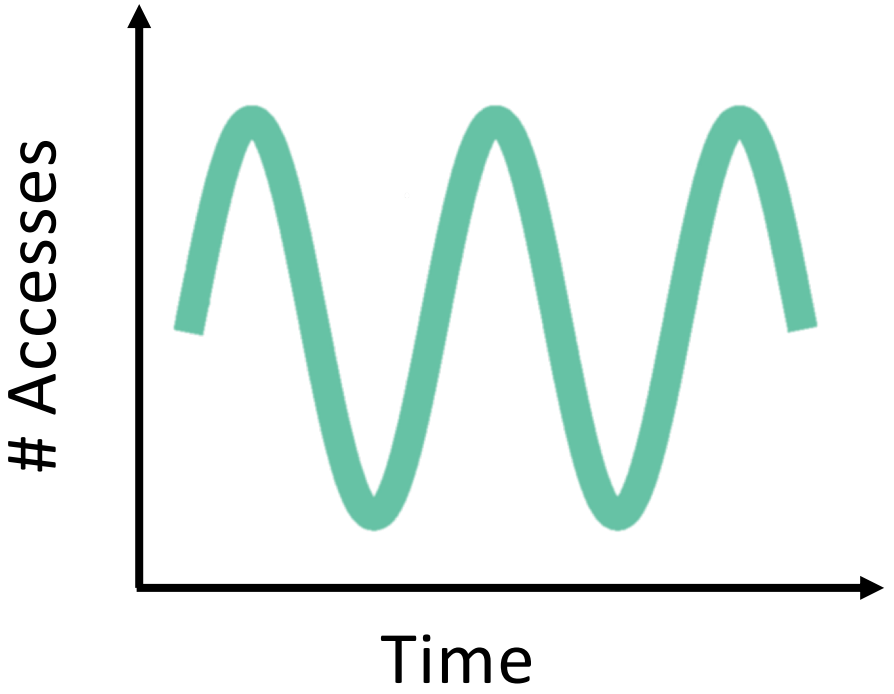
UNIVERSITY OF WATERLOO

Tiresias

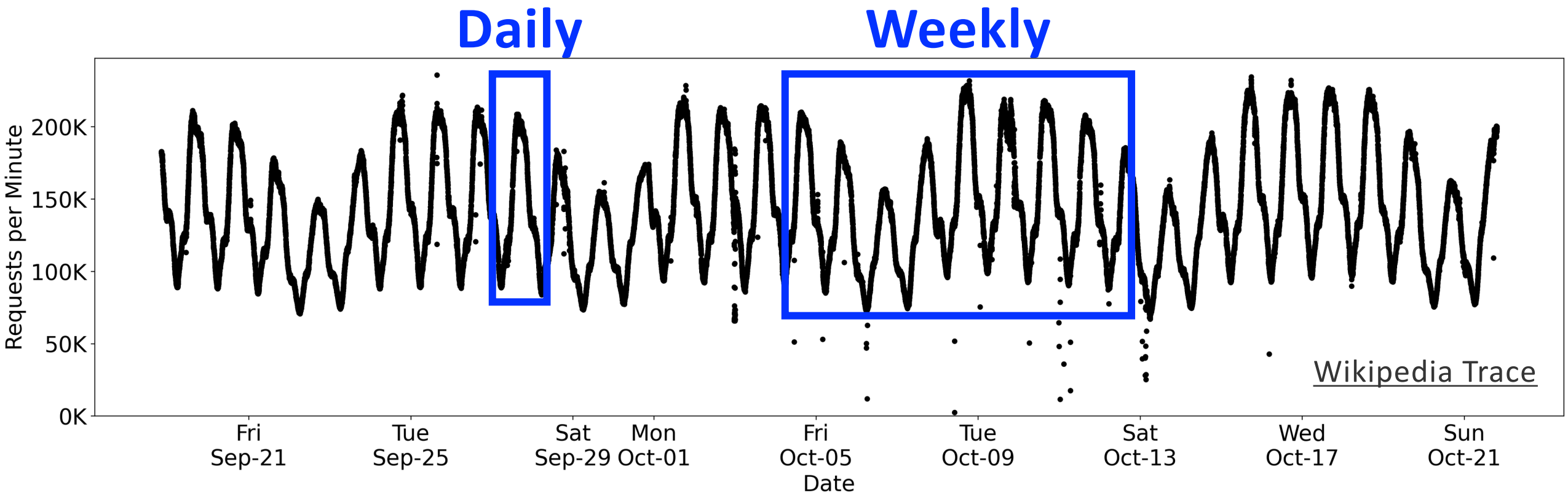

Predict upcoming accesses and latency under different storage and indexing choices
Tiresias
Model workload *access* history
Model access *latencies*
1
2
DBMS
Store and index data
Change data storage and indexing
3
Report query latency and access history

Access Arrival Patterns

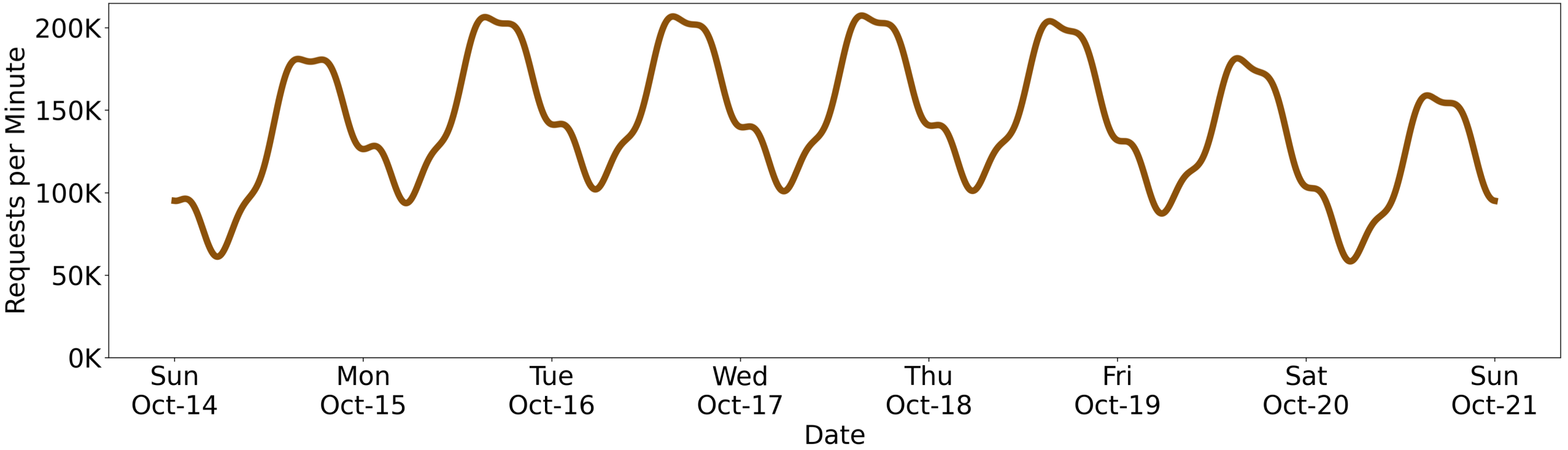




Access Arrival Patterns

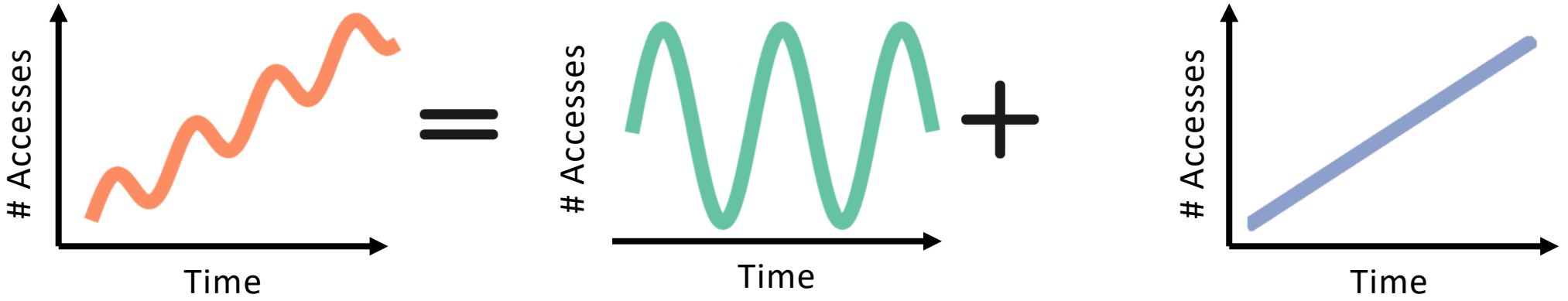


Learning Access Arrival Patterns



UNIVERSITY OF WATERLOO

Learning Access Arrival Patterns

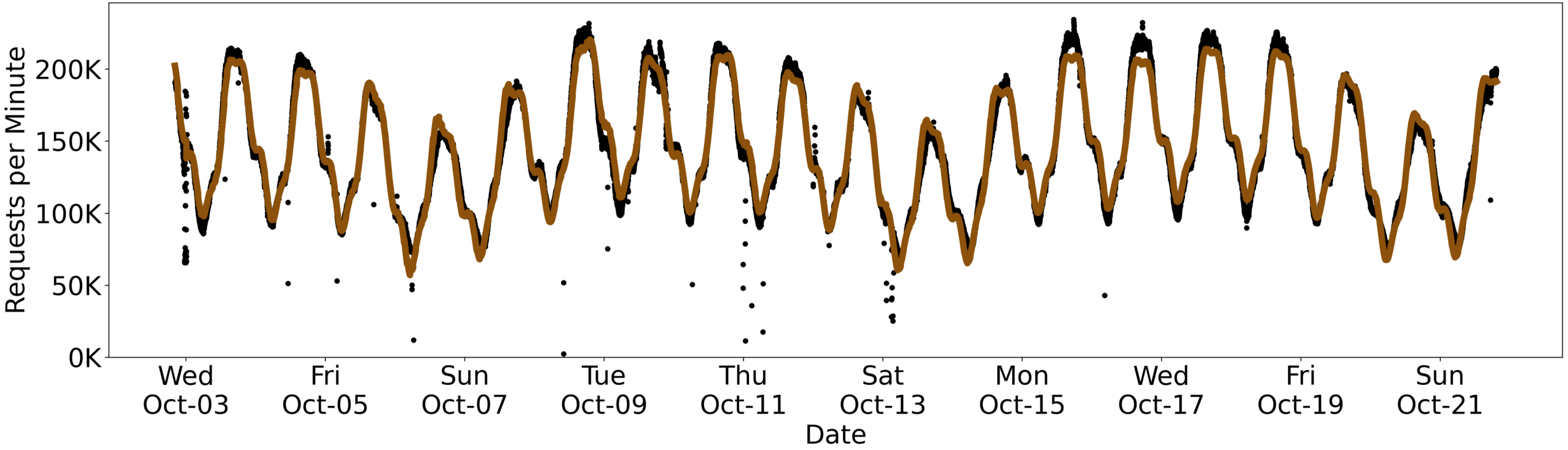


	Periodicity	Trend
SPAR	Avg. over user defined intervals	Avg. over interval
Hybrid-Ensemble	Recurrent Neural Network	Linear Regression

UNIVERSITY OF WATERLOO

Access Arrival Patterns - Results

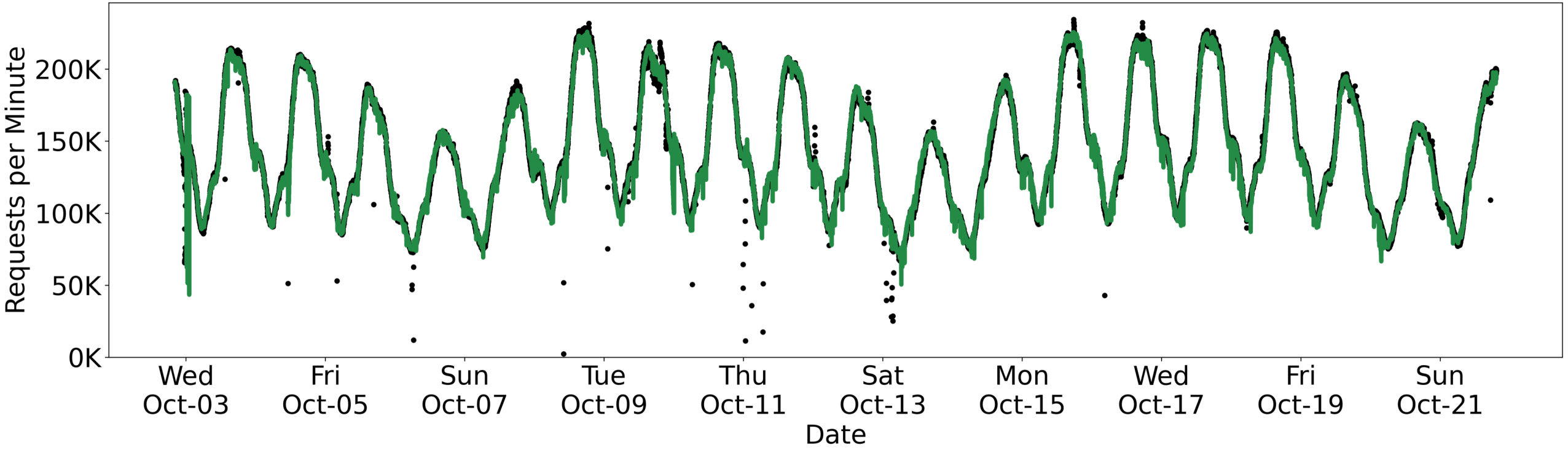
Hybrid-Ensemble



UNIVERSITY OF WATERLOO

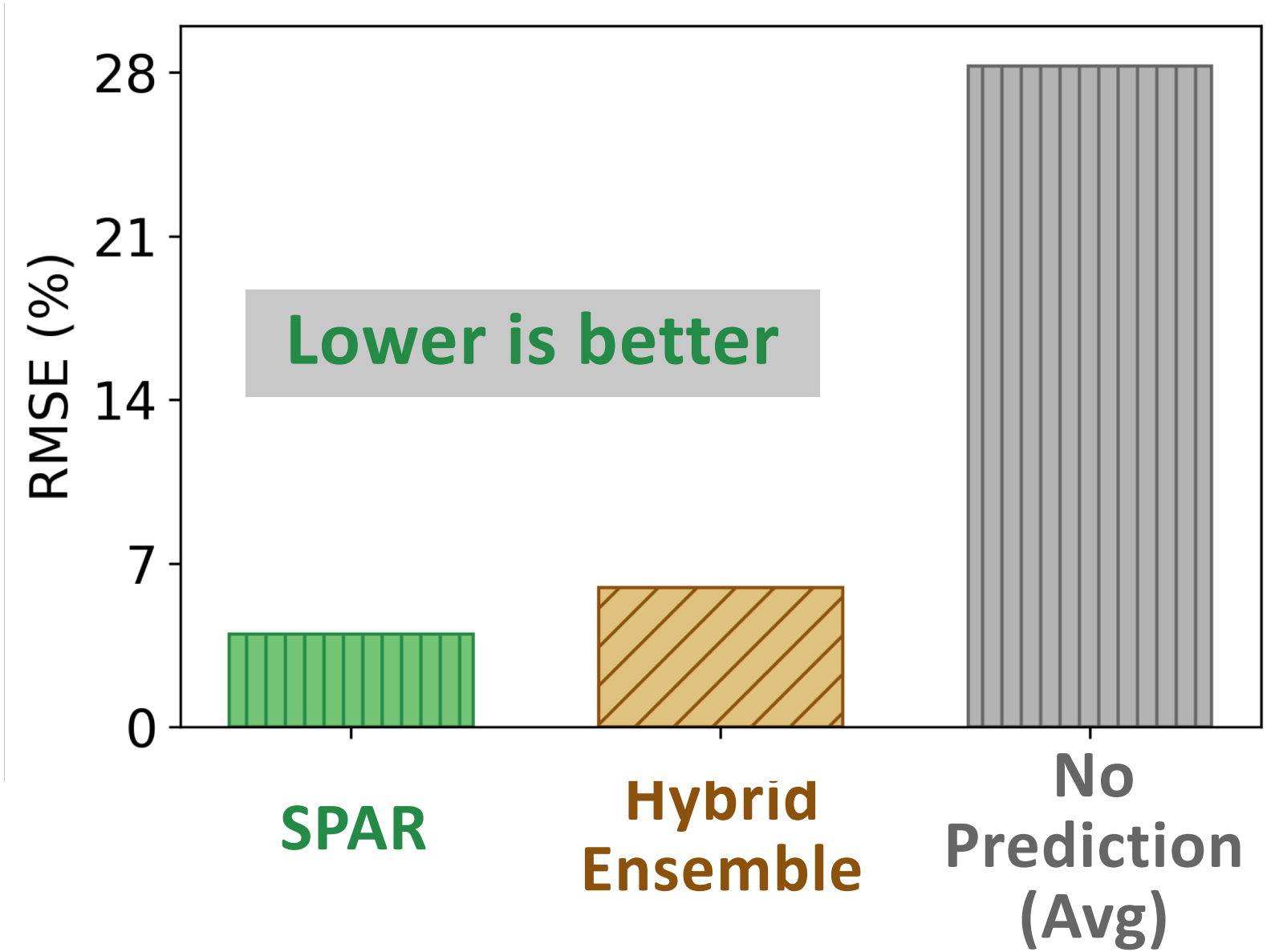
Access Arrival Patterns - Results

SPAR



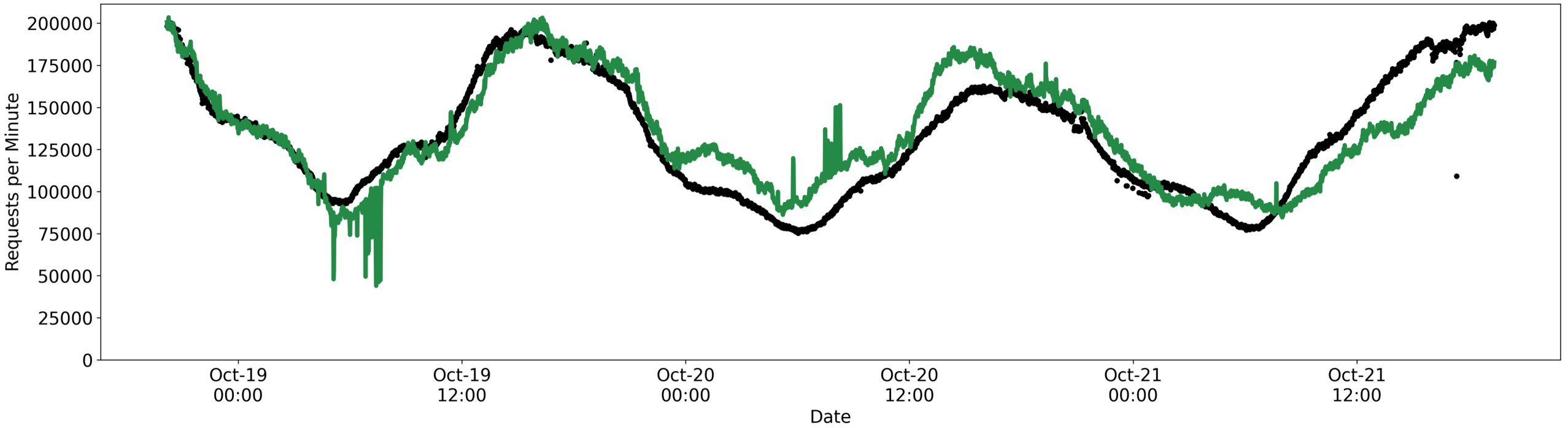
UNIVERSITY OF WATERLOO

Access Arrival Patterns - Accuracy



Access Arrival Patterns - Results

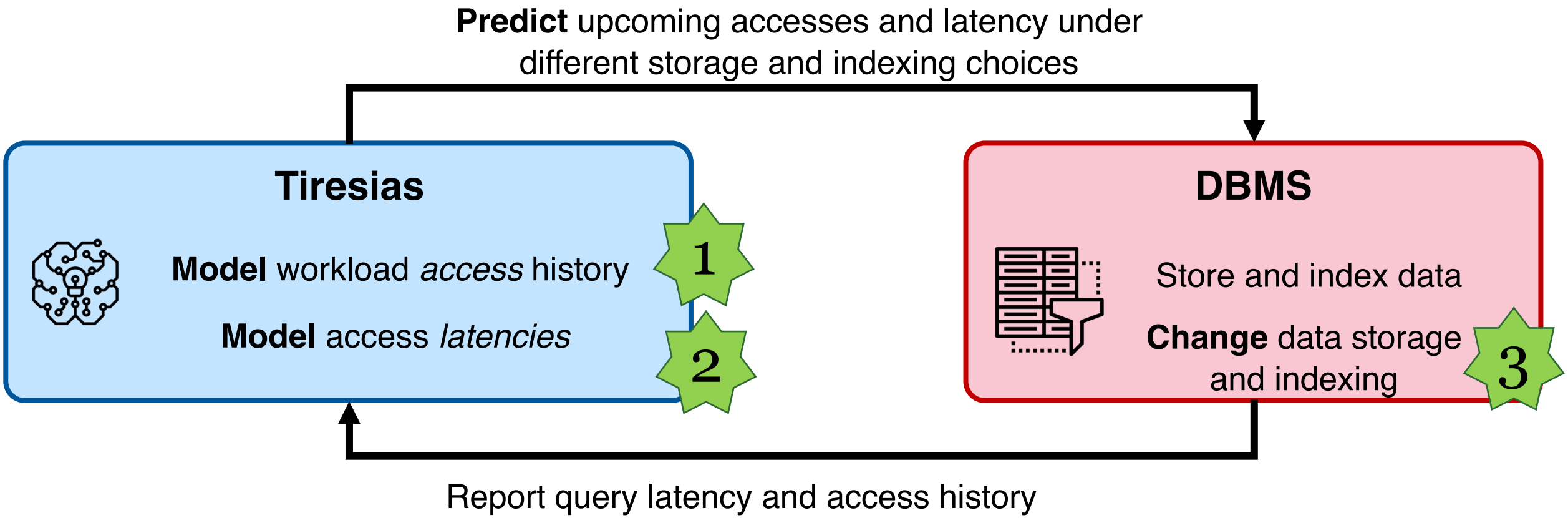
SPAR (Mis-Configured)



Incorrect definition of period **decreases** accuracy!

UNIVERSITY OF WATERLOO

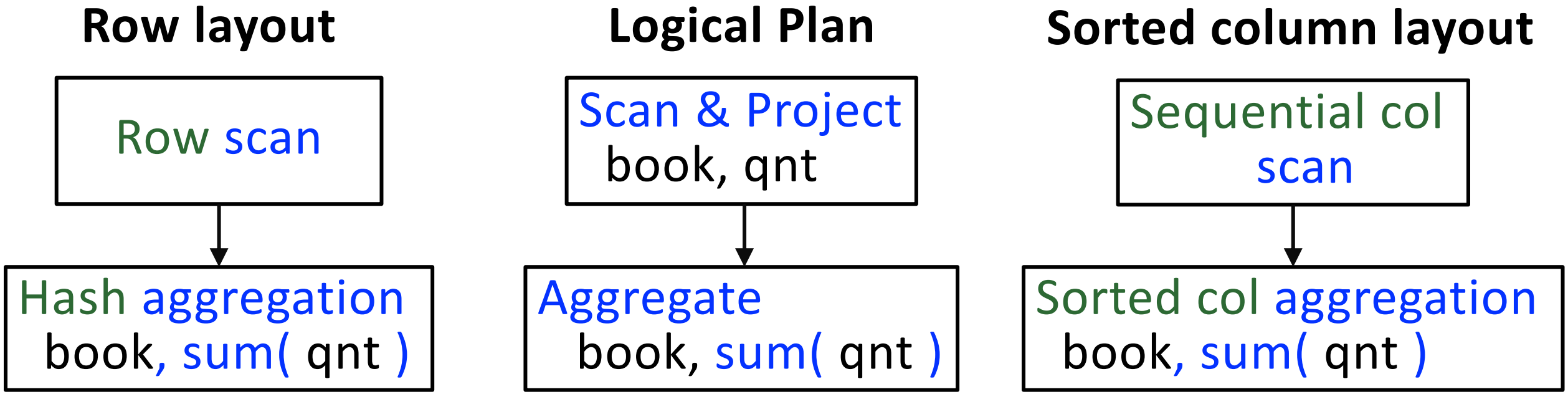
Tiresias


Predict upcoming accesses and latency under different storage and indexing choices
Tiresias
Model workload *access* history
Model access *latencies*
1
2
DBMS
Store and index data
Change data storage and indexing
3
Report query latency and access history

Predicting Latency

Transactions are composed of **physical operators**

SELECT book, **SUM**(qnt) **GROUP BY** book



UNIVERSITY OF WATERLOO

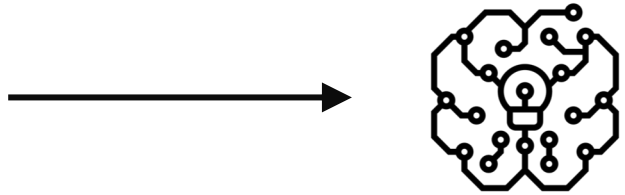
Predicting Latency

Transactions are composed of **physical operators**

Predict physical operator latency

Per layout with workload stats as parameters

Cardinality
Data Width
Est Selectivity



Predicted Latency

Seq col scan

UNIVERSITY OF WATERLOO

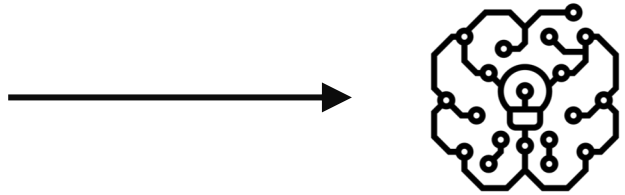
Predicting Latency

Transactions are composed of **physical operators**

Predict physical operator latency

Per layout with workload stats as parameters

Cardinality
Data Width
Est Selectivity



Predicted Latency

Row scan

UNIVERSITY OF WATERLOO

Predicting Latency

Transactions are composed of **physical operators**

Predict physical operator latency

Per layout with workload stats as parameters

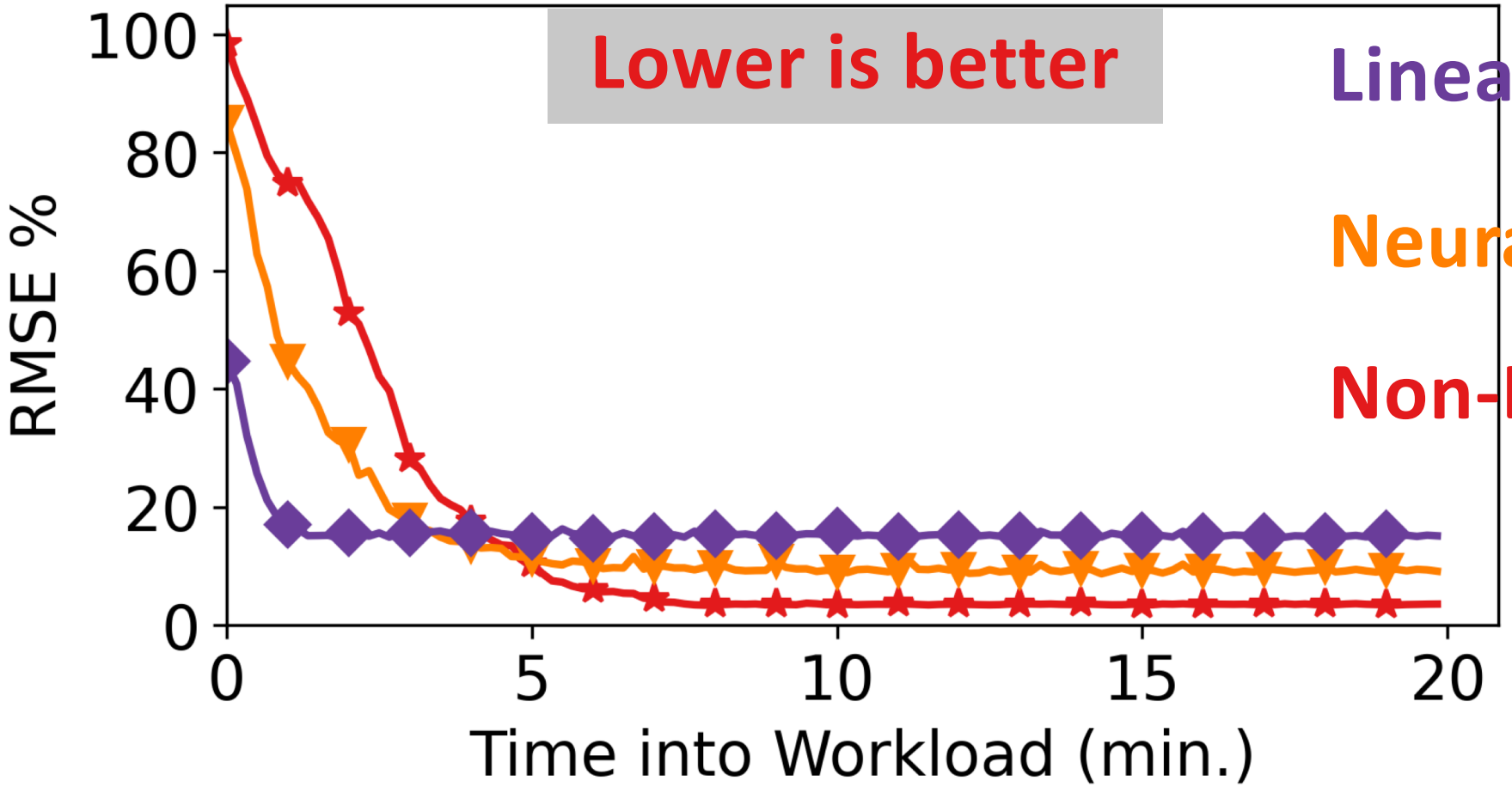
Linear Regressor

Neural Network

Non-Linear Regressor

UNIVERSITY OF WATERLOO

Predicting Latency Accuracy



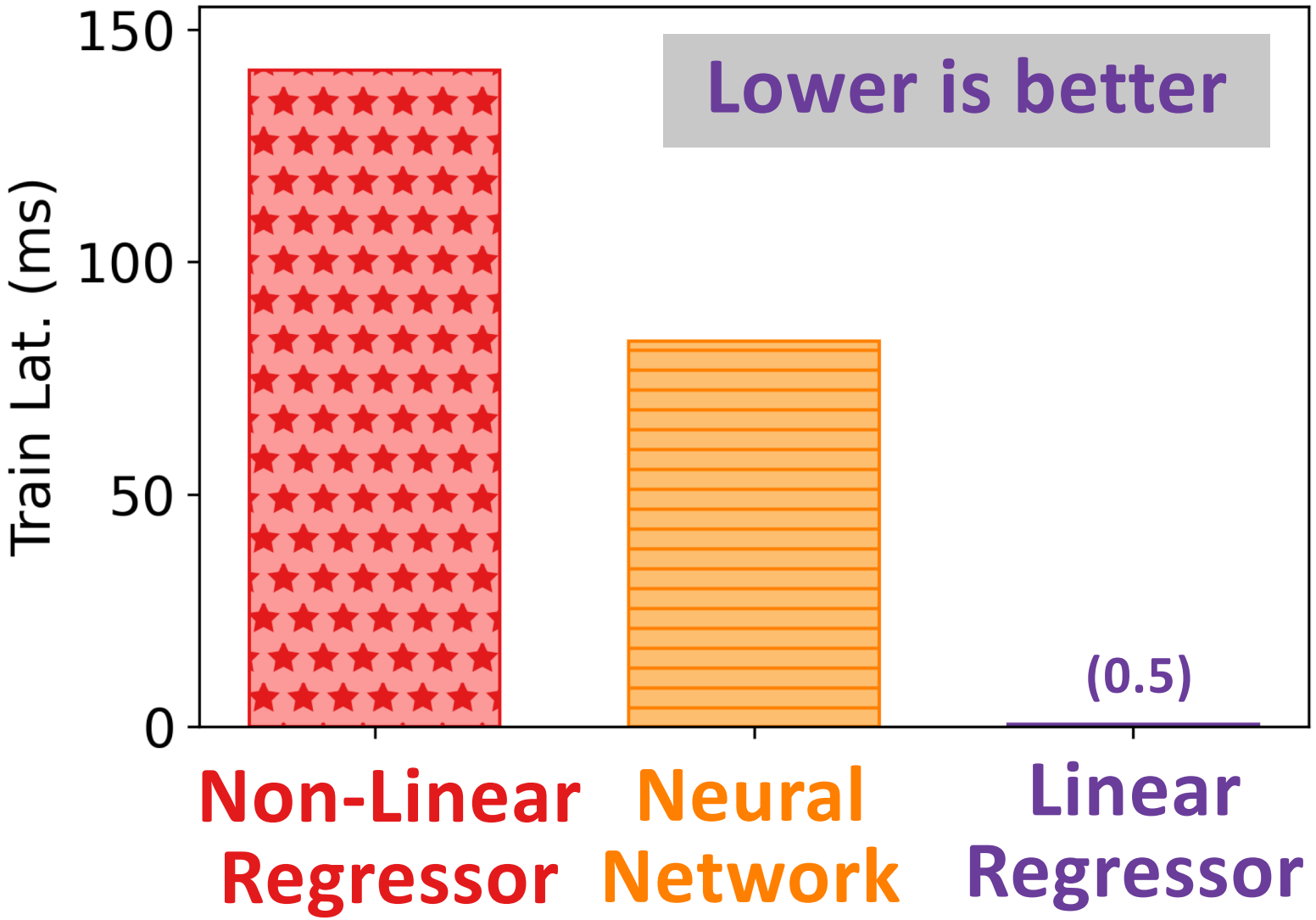
Linear Regressor

Neural Network

Non-Linear Regressor

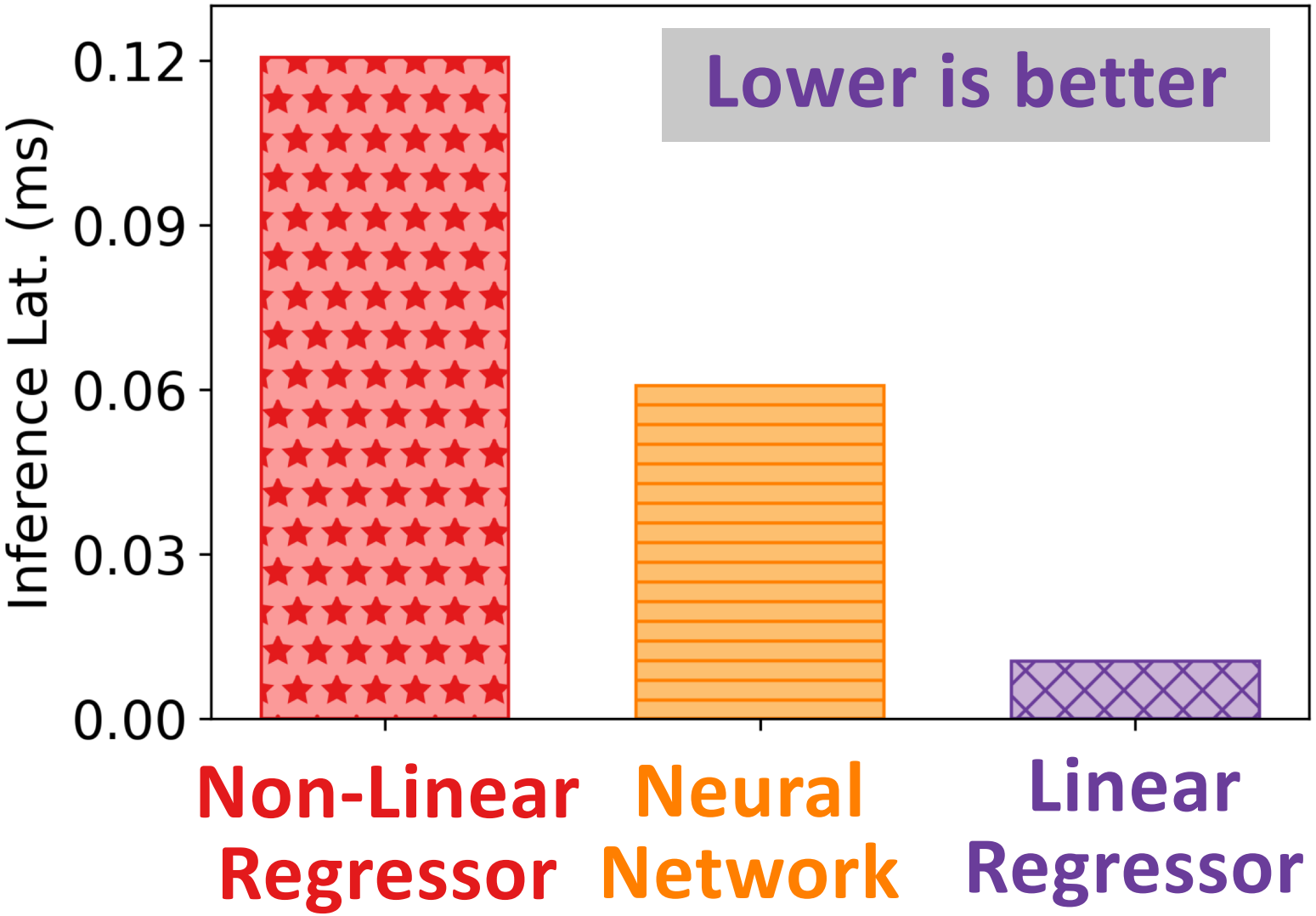
UNIVERSITY OF WATERLOO

Predicting Latency - Training



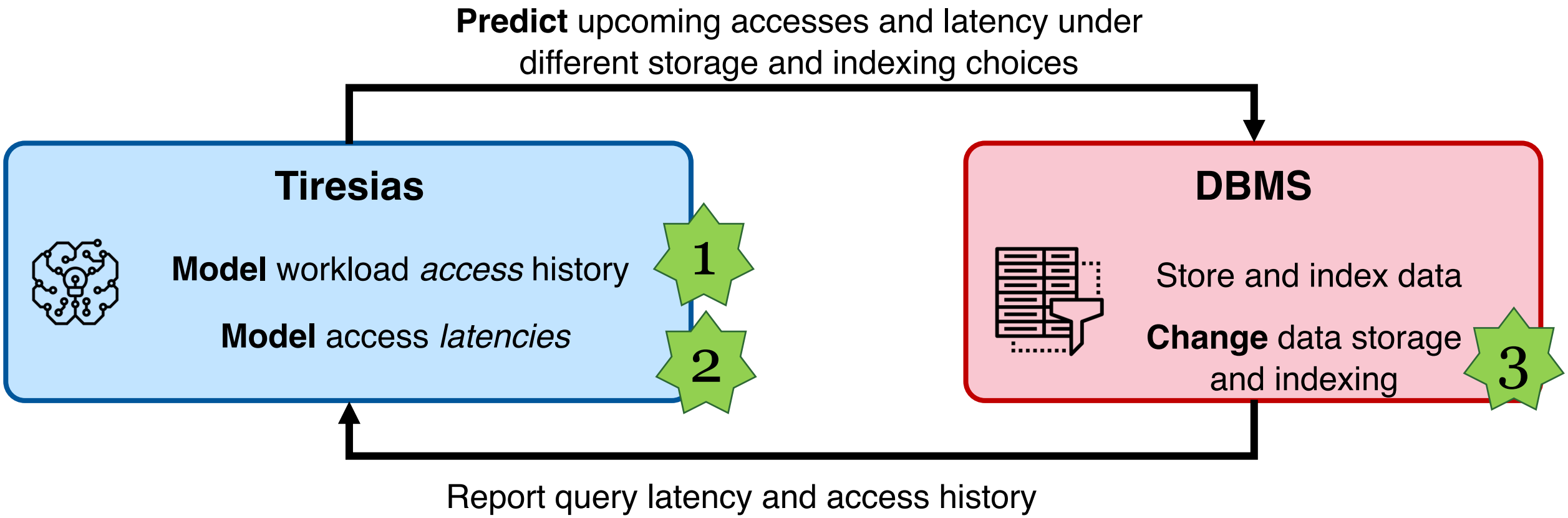
UNIVERSITY OF WATERLOO

Predicting Latency - Inference



UNIVERSITY OF WATERLOO

Tiresias


Predict upcoming accesses and latency under different storage and indexing choices
Tiresias
Model workload *access* history
Model access *latencies*
1
2
DBMS
Store and index data
Change data storage and indexing
3
Report query latency and access history

Storage and Index Changes

Expected Benefit of Change

Predicted access **latency** under:
current & **proposed** storage/indexing choice

Weighted by **likelihood** of **access**

Cost incurred to perform change

UNIVERSITY OF WATERLOO

Tiresias End-to-End Evaluation

DBMS

PostgreSQL: automatic indexing

OLAP DBMS: predictive cracking

Proteus: adaptive storage for HTAP

Adapt with **Tiresias**

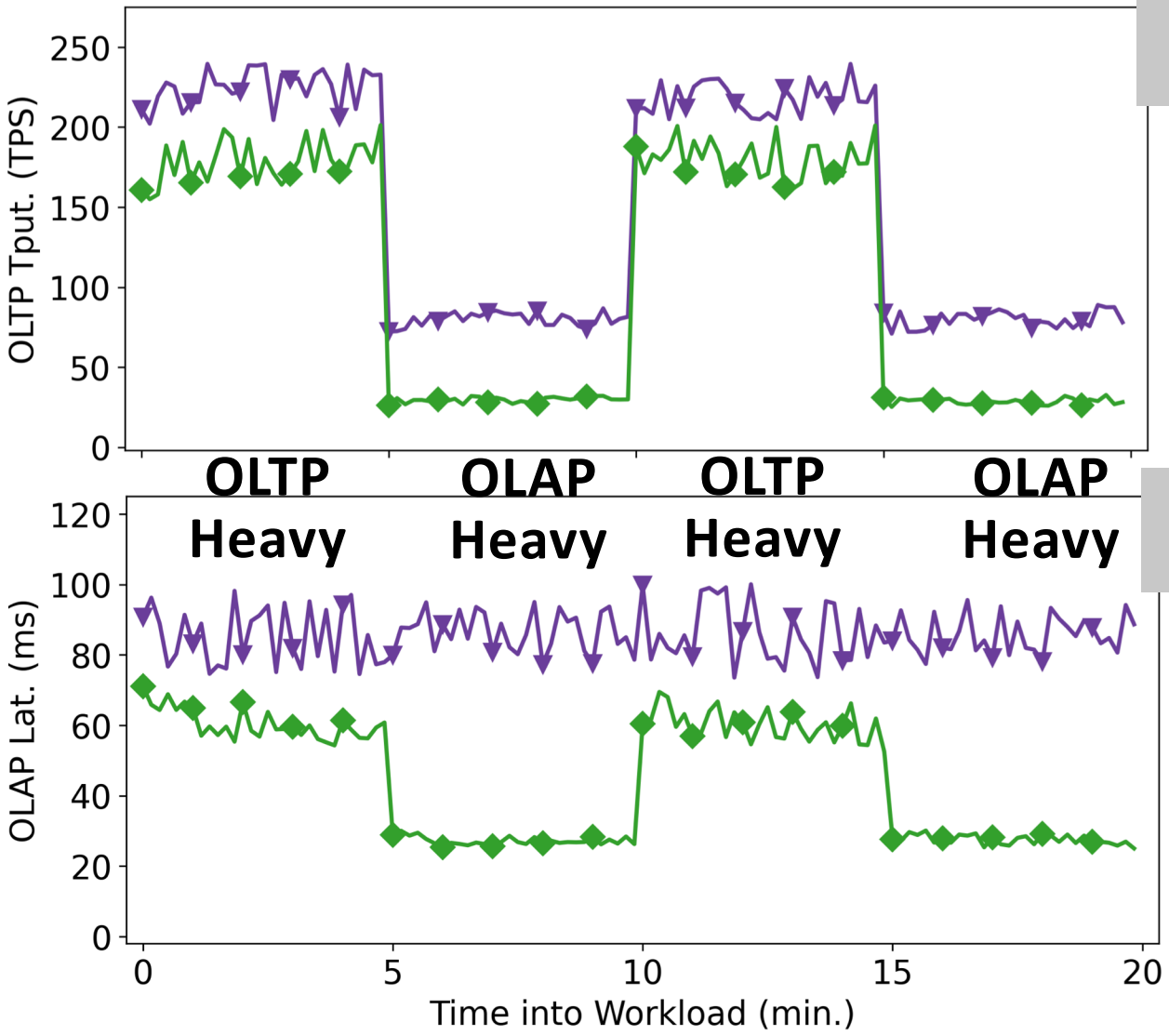
Static choices **without** Tiresias

Shifting Workloads

Skew (hotspots)

Mix (OLTP or OLAP heavy)

End-to-End PostgreSQL Indexes



Higher is better

No Index

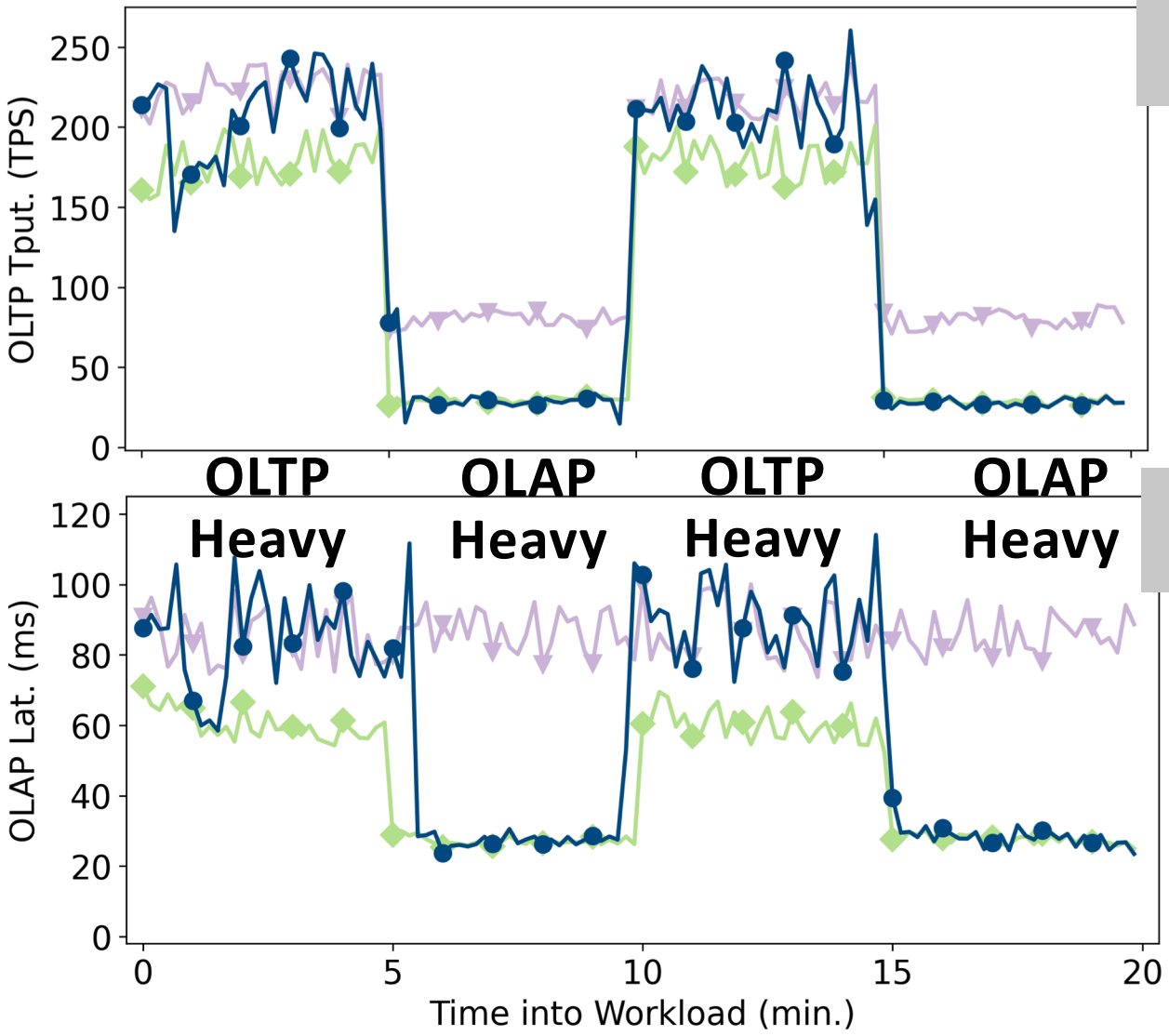
Index

Lower is better

No Index

Index

End-to-End PostgreSQL Indexes



Higher is better

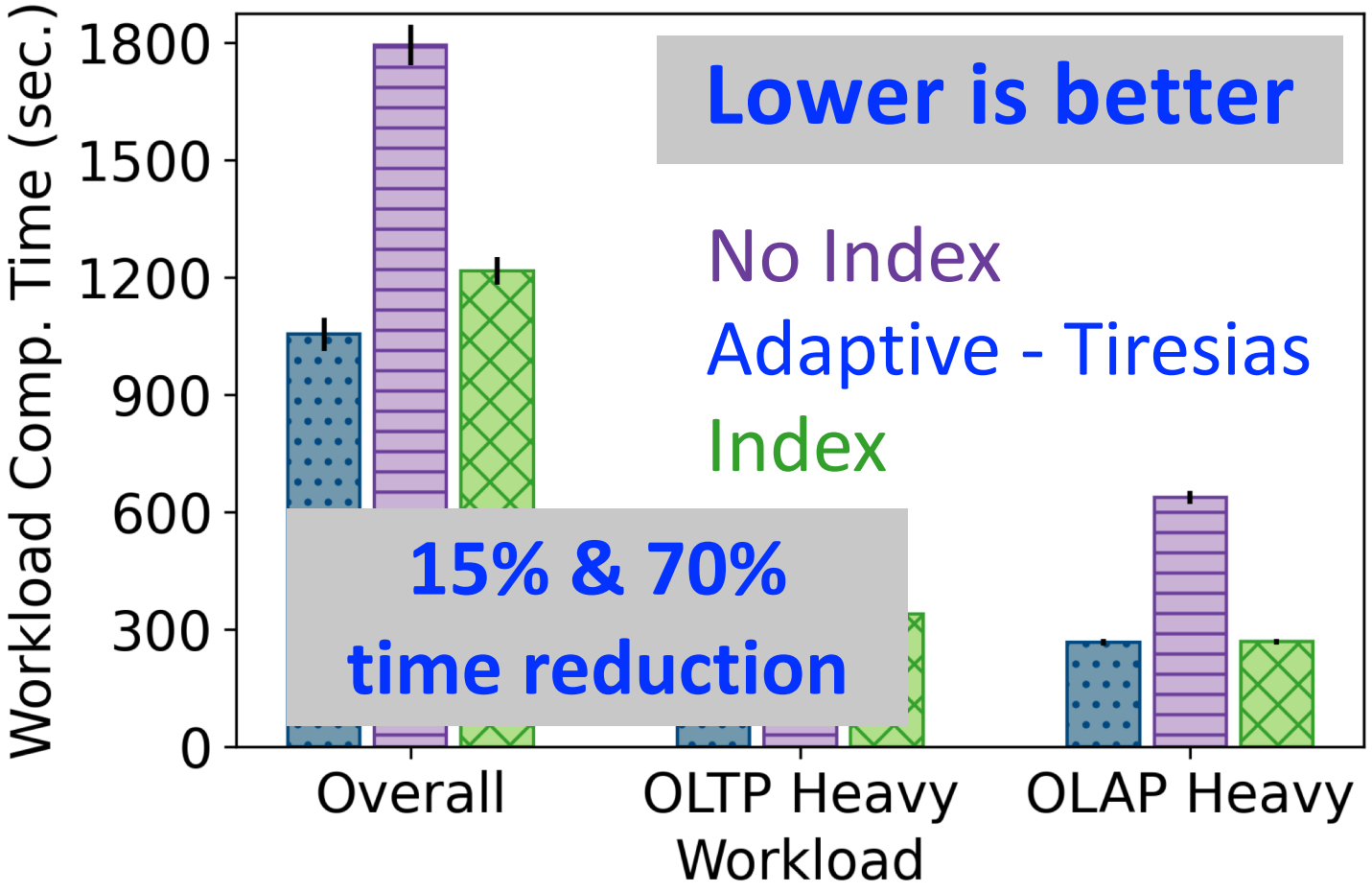
No Index
Adaptive - Tiresias
Index

Lower is better

No Index
Adaptive - Tiresias
Index

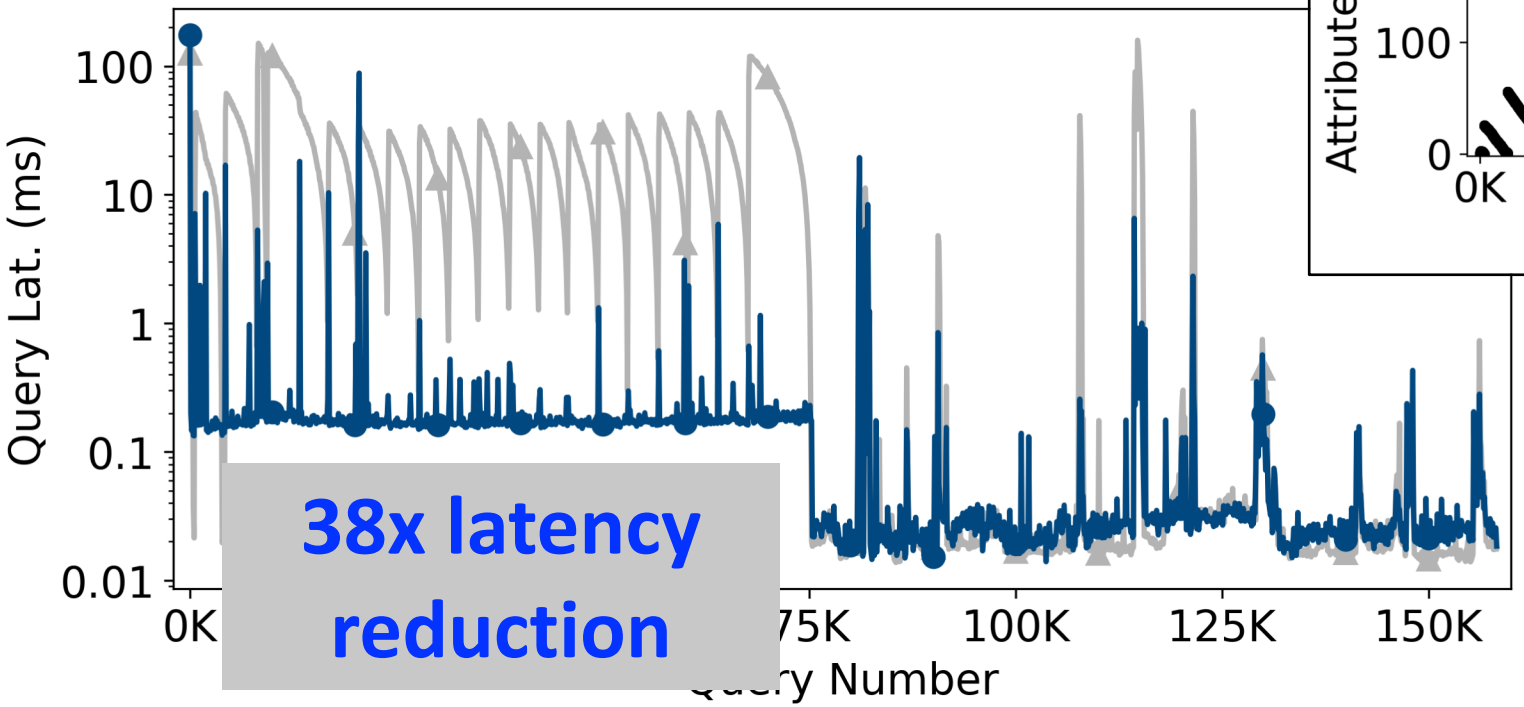
UNIVERSITY OF WATERLOO

End-to-End PostgreSQL Indexes

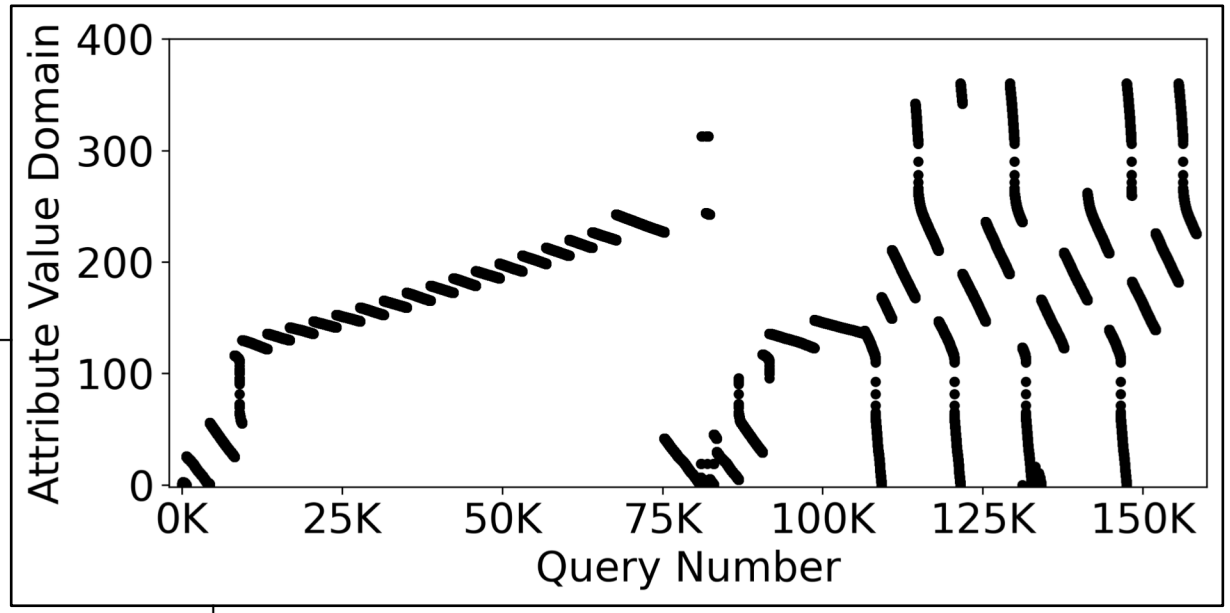


End-to-End Cracking

Lower is better



38x latency reduction

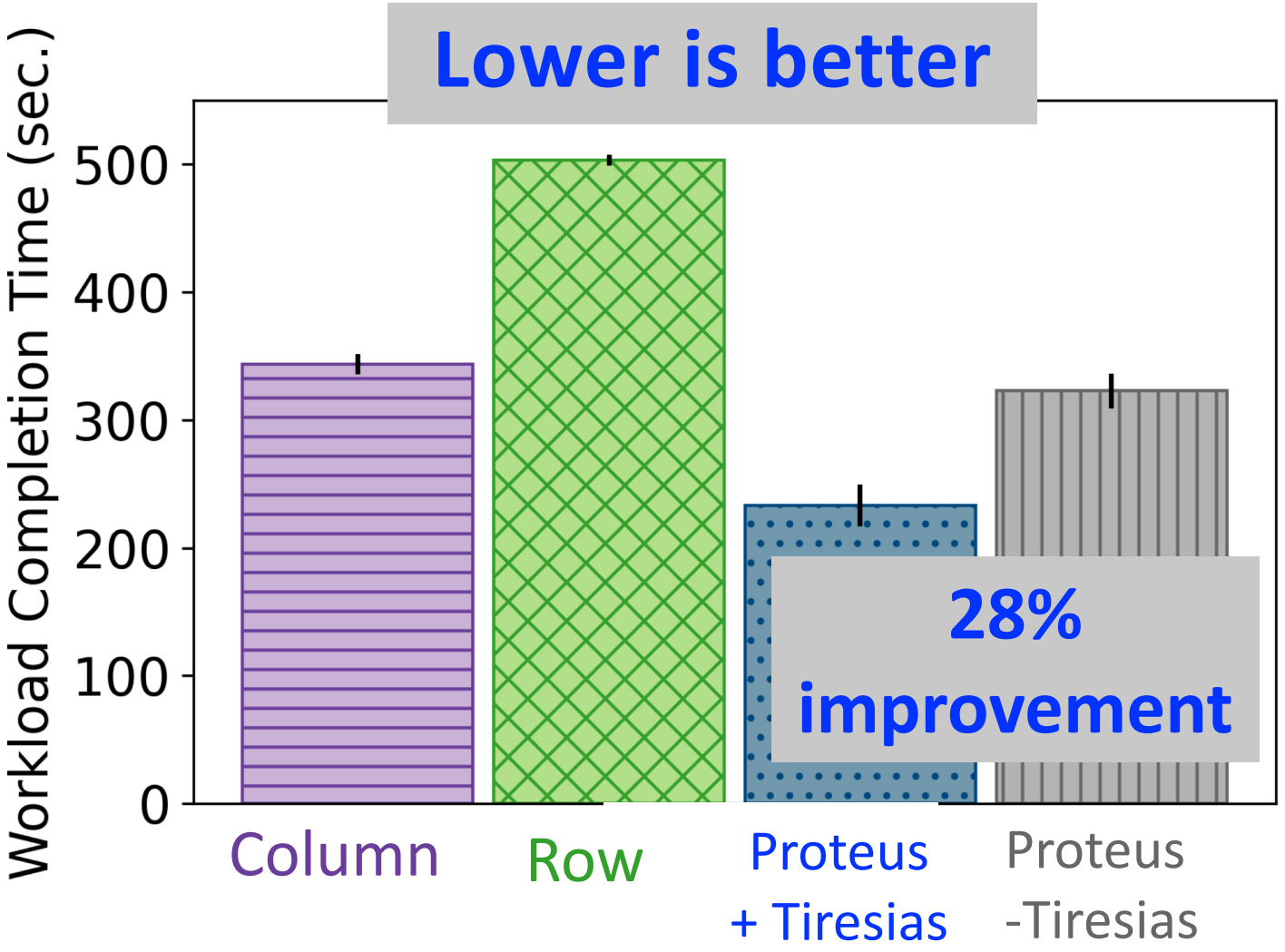


Cracking (No Tiresias)

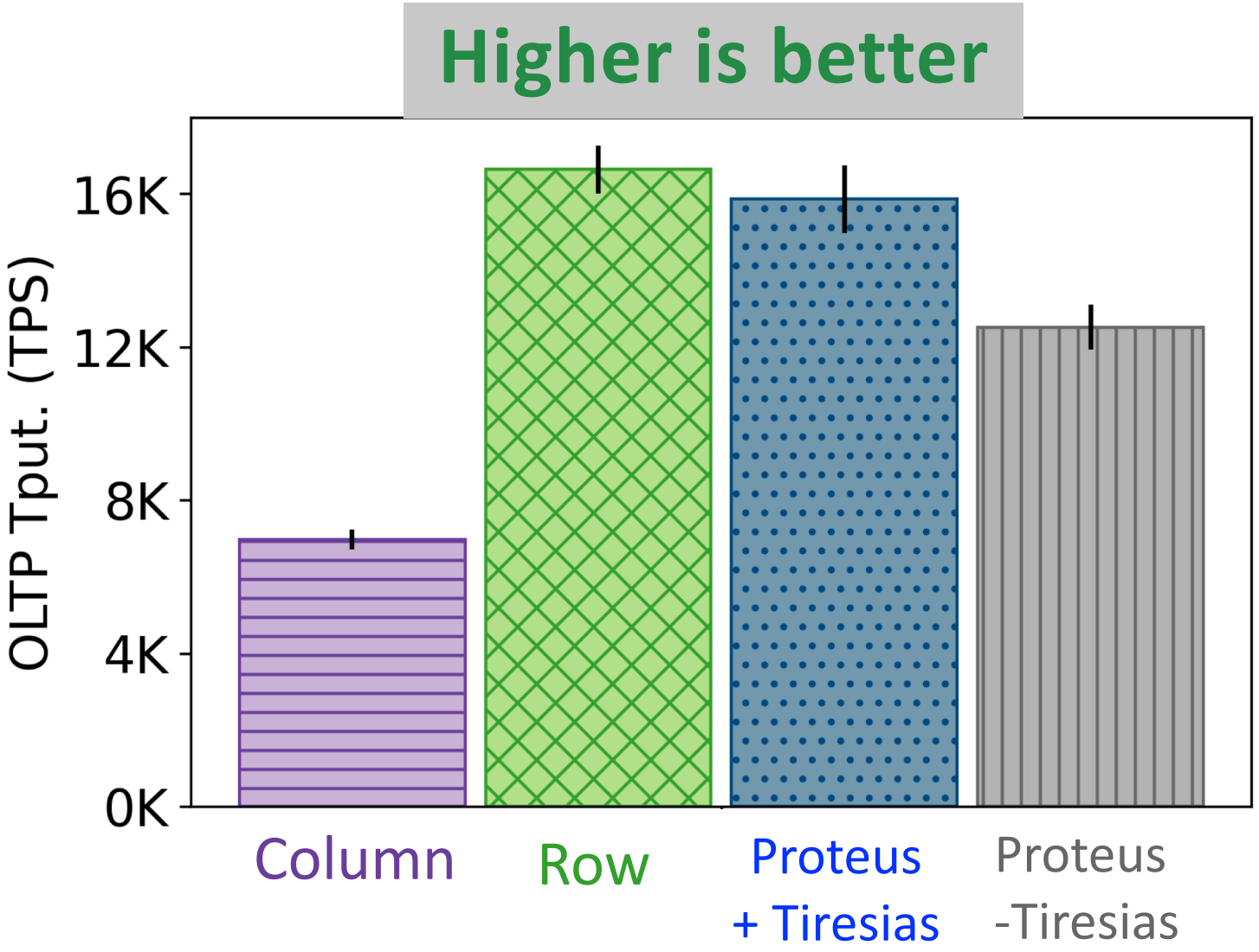
Predictive Cracking: Tiresias

UNIVERSITY OF WATERLOO

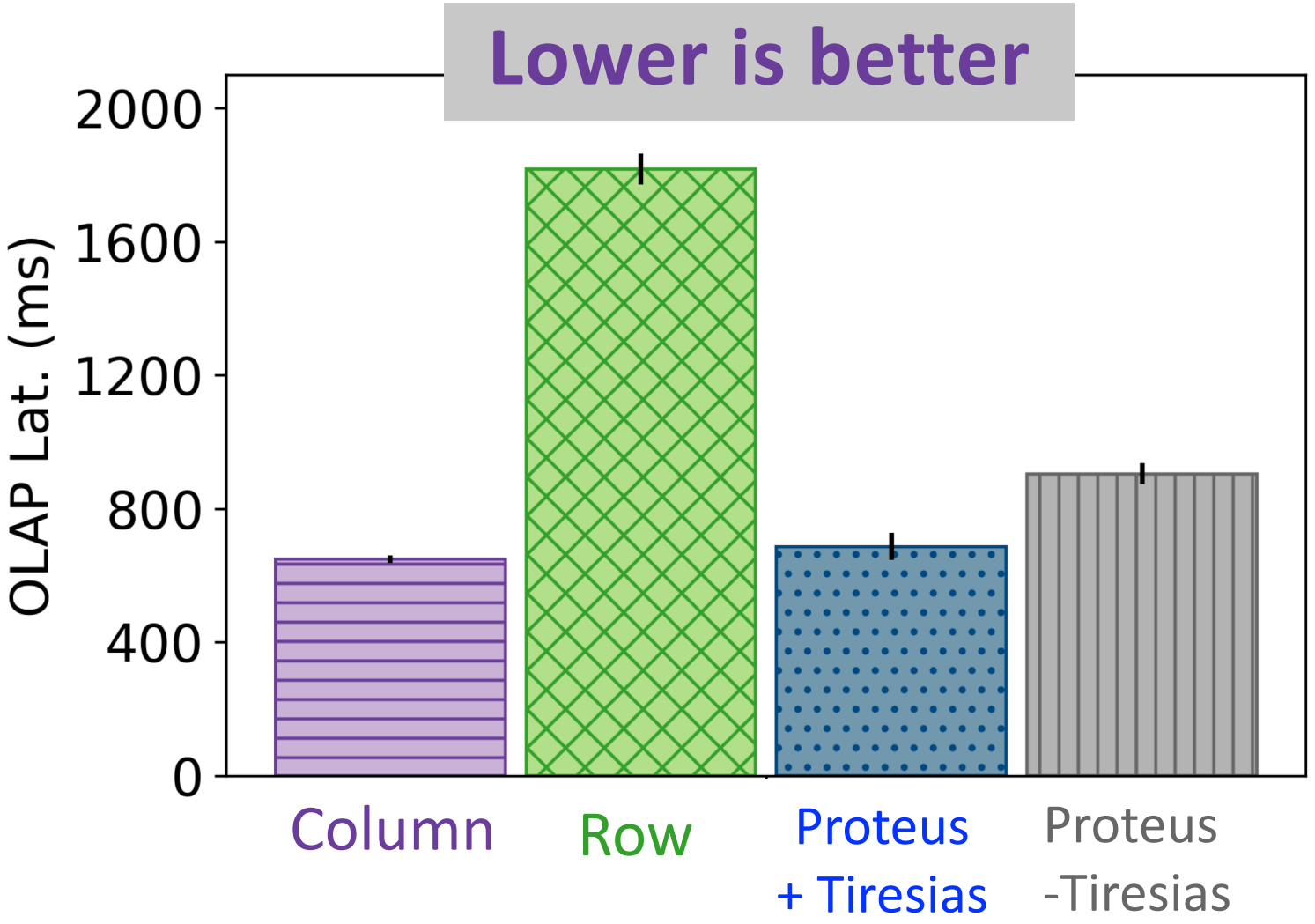
End-to-End Proteus



End-to-End Proteus



End-to-End Proteus



Tiresias Takeaways tiny.cc/tiresias

Automatic adaptation of data storage and indexing

Generalizable API used in different DBMSs

Predict data accesses based on workload patterns

Learn access costs under different storage choices

UNIVERSITY OF WATERLOO