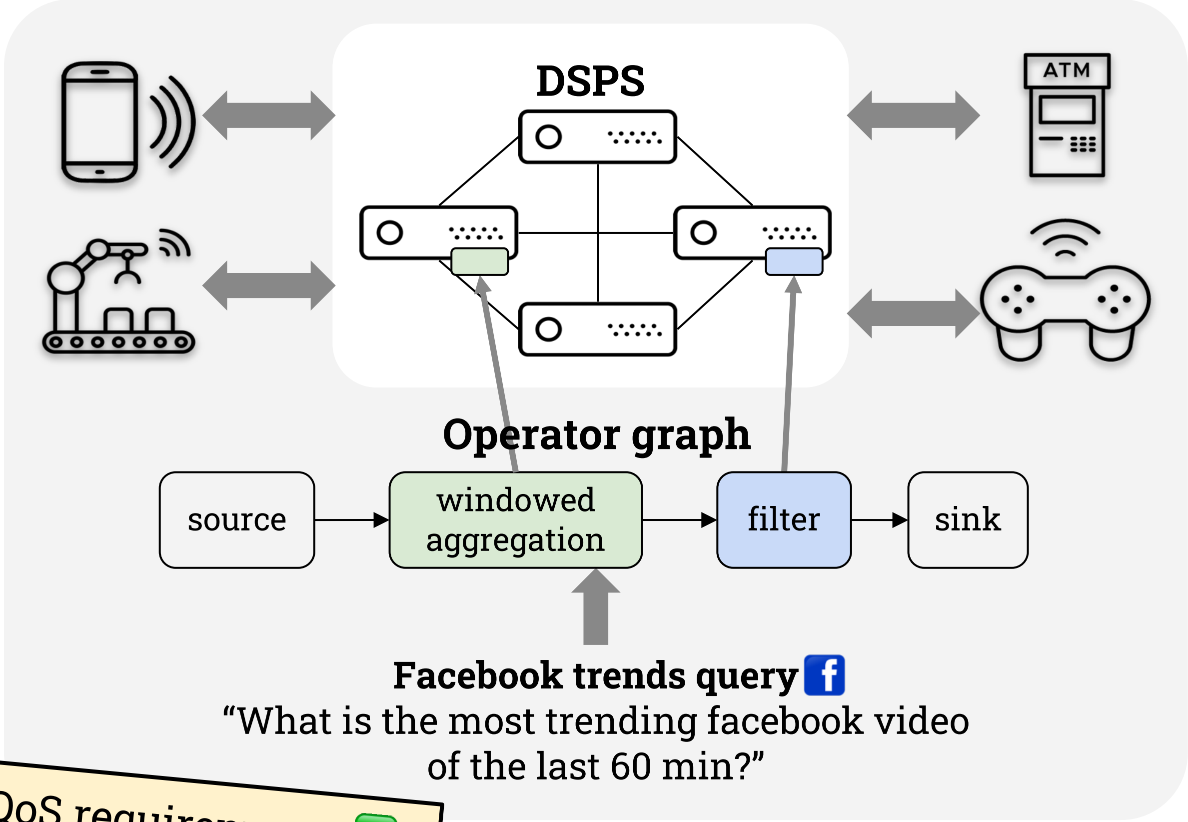
Zero-Shot Cost Models for Distributed Stream Processing

16th ACM International Conference on
Distributed and Event-based Systems (DEBS)
Copenhagen 2022

Roman Heinrich[1], Manisha Luthra[2], Harald Kornmayer[1], Carsten Binnig[2]
[1] DHBW Mannheim, [2] TU Darmstadt

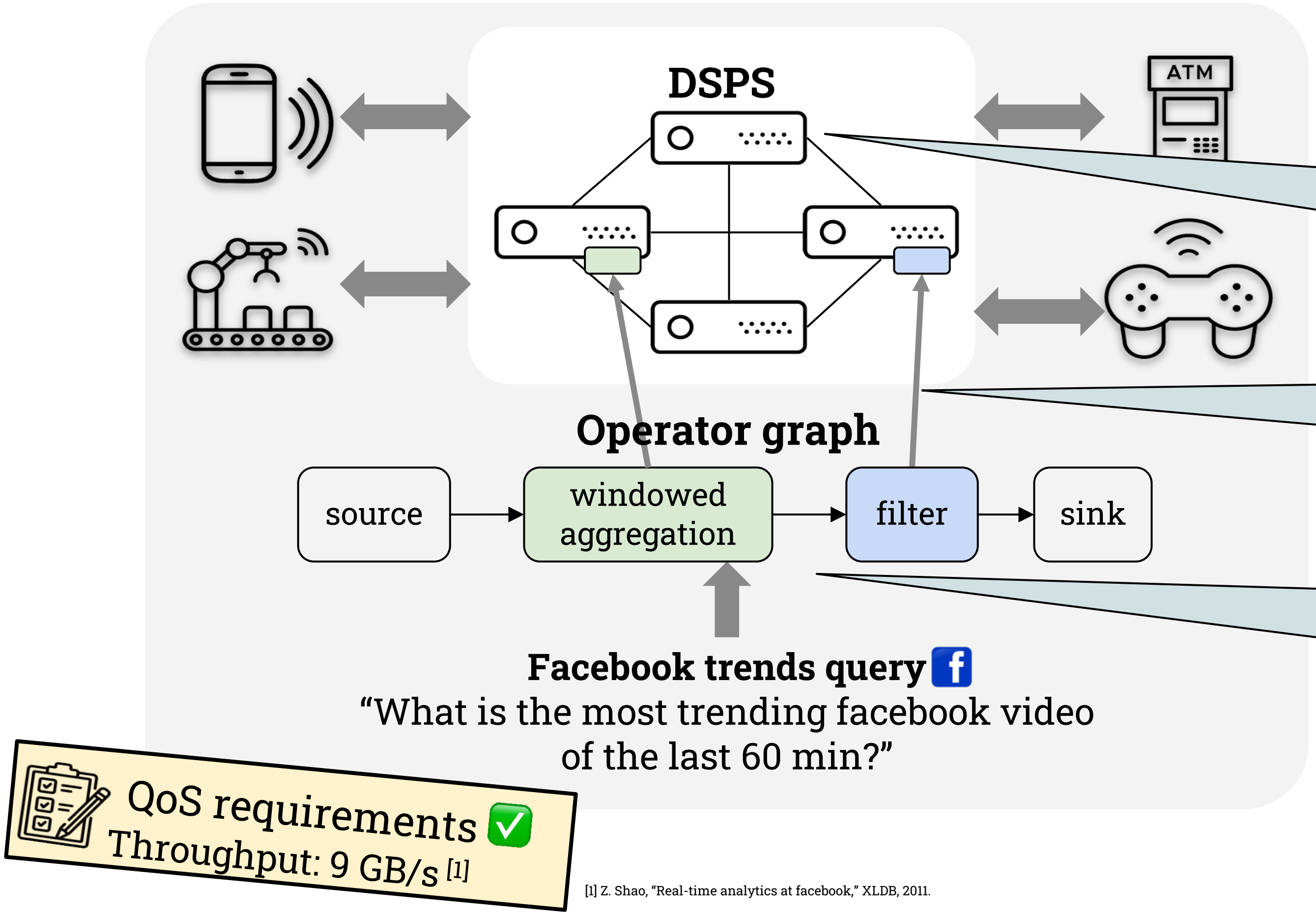
Distributed Stream Processing Systems (DSPS)



QoS requirements ✅
Throughput: 9 GB/s [1]

[1] Z. Shao, "Real-time analytics at facebook," XLDB, 2011.

Optimization tasks for DSPS



Optimization tasks

Elasticity
Add & remove hardware resources

Operator placement
Find the optimal placement

Query optimization
Select the optimal operator graph configuration

[1] Z. Shao, “Real-time analytics at facebook,” XLDB, 2011.

Cost models for DSPS

join

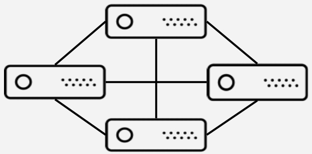
filter

windowed aggregation

Query operators

Workload

Data stream characteristics



Hardware properties

Cost estimations

Provided by a cost model

Latency: 52 ms
Throughput: 5 GB/s

Optimization tasks

Elasticity
Add & remove hardware resources

Operator placement
Find the optimal placement

Query optimization
Select the optimal operator graph configuration

Limitations of existing approaches

join

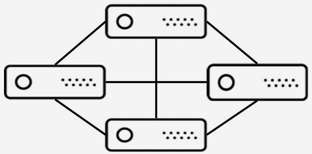
filter

windowed aggregation

Query operators

Workload

Data stream characteristics



Hardware properties

Monitoring

❌ query execution(s) required
❌ gives no idea how to improve

Heuristics & analytical approaches[2]

❌ inaccurate estimates
❌ often limited to specific optimization

Learned models[3]

✅ show accuracte estimates
❌ no generalization → workload-driven

```
costs = f(query, streams, hardware)
```

Optimization tasks

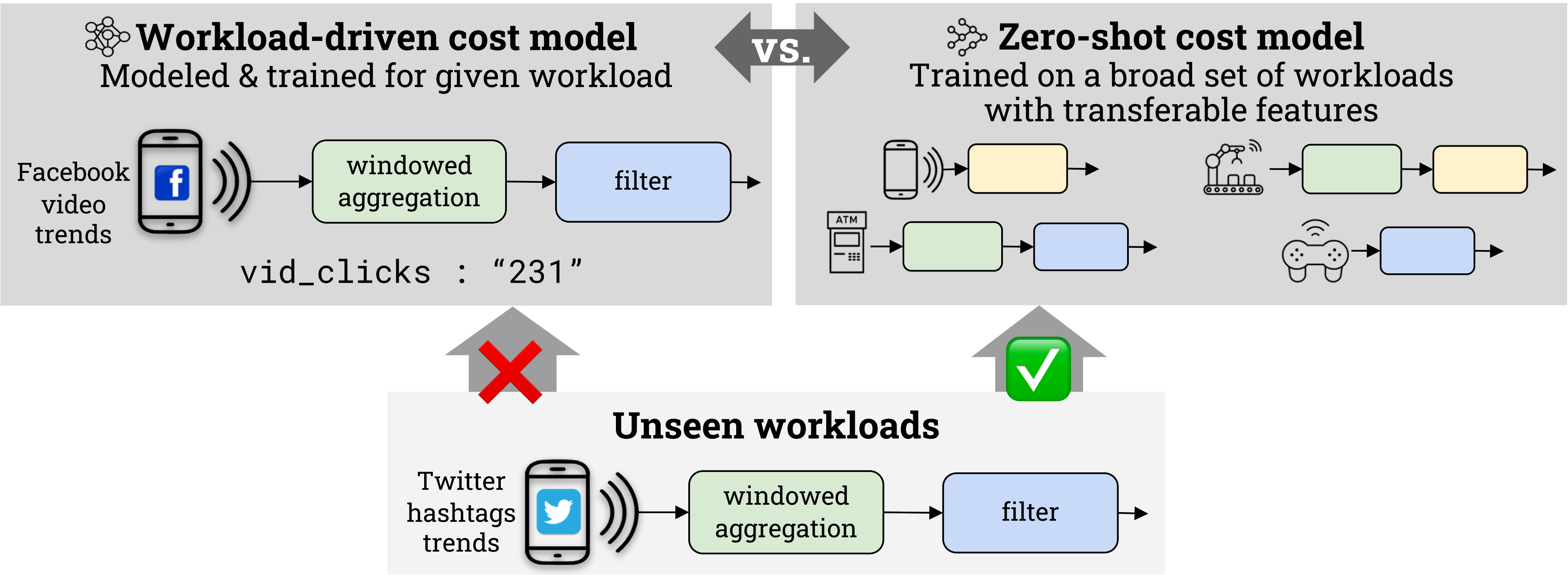
Elasticity
Add & remove hardware resources

Operator placement
Find the optimal placement

Query optimization
Select the optimal operator graph configuration

[2] L. Eskandari, J. Mair, Z. Huang, and D. Eyers, “I-scheduler: Iterative scheduling for distributed stream processing systems,” Future Generation Computing ystems, vol. 117, pp. 219–233, 2021.
[3] T. Li, Z. Xu, J. Tang, and Y. Wang, “Model-free control for distributed stream data processing using deep reinforcement learning,” PVLDB, vol. 11, no. 6, p. 705–718, 2018.

Workload-driven vs. zero-shot models



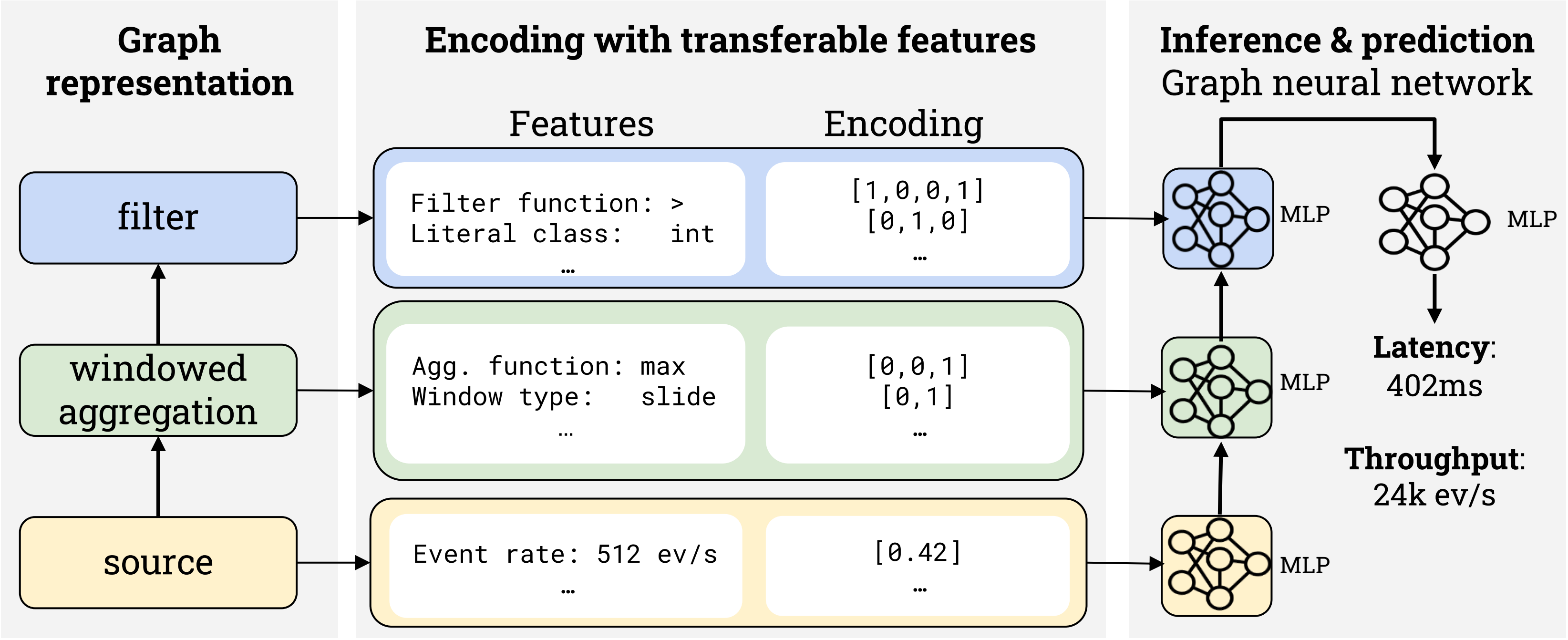
- ❌ No generalization possible
- ❌ Costly re-training for a new workload is required

- ✅ Generalization to unseen streams, queries and hardware
- ✅ No retraining required

Zero-shot model architecture

User query

"What is the most trending facebook video of the last 60 min?"



Transferable features

Non-transferable features

```
region:  “europe”
vid_clicks : “231”
```

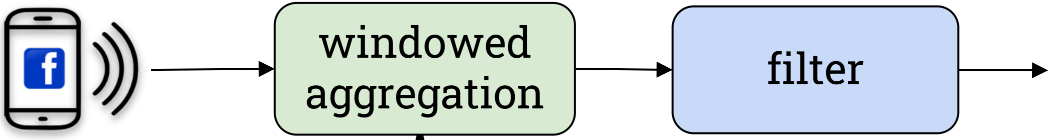
vs.

Transferable features

```
event_rate:  312ev/s
selectivity: 0.9
```

❌ rely on given workload

✅ workload-agnostic



Operator-related features

```
window type:   sliding
window policy: time
window length: 60min
```

…

Data-related features

```
selectivity:      0.66
tuple width in:   5
tuple width out:  5
```

…

Selectivity definitions → paper

Hardware-related features

```
instance size: large
```

Large: 8CPU, 8GB RAM, 80GB disk

Training & inference methodology

Generate training data

Create a broad data set by enumeration

Execute queries and collect costs (labels)

Enumerate operator properties
For filter, aggregation, join, window

Example – window:
window_type: sliding, tumbling
window_policy: count-based, time-based
window_length: [3... 100] tuples, [0.25...3] sec

Enumerate query structures
Linear query, 2-way-join, 3-way-join

Enumerate workload properties
Different event rates & tuple widths

Enumerate hardware
Different CPU & RAM sizes

Inference

Unknown workload → **Train GNN** → Cost estimation

✅ Graph representation for GNNs allow predictions for flexible & unseen queries

Evaluation of the zero-shot model

- **Set-up:** 10 clusters (each 10 nodes) with Apache Storm v2.2.0
- **Cost metrics**:
  end-to-end latency & throughput
- **Metric**: q-error
  $q(c,\hat{c})=\max(c/\hat{c},\ \hat{c}/c)$
  - reporting median and 95-percentile
  - $q \leftarrow 1$: perfect estimate

Interpolation for workloads & placements

	median	95th
Latency:	**1.13**	3.19
Throughput:	**1.16**	3.50

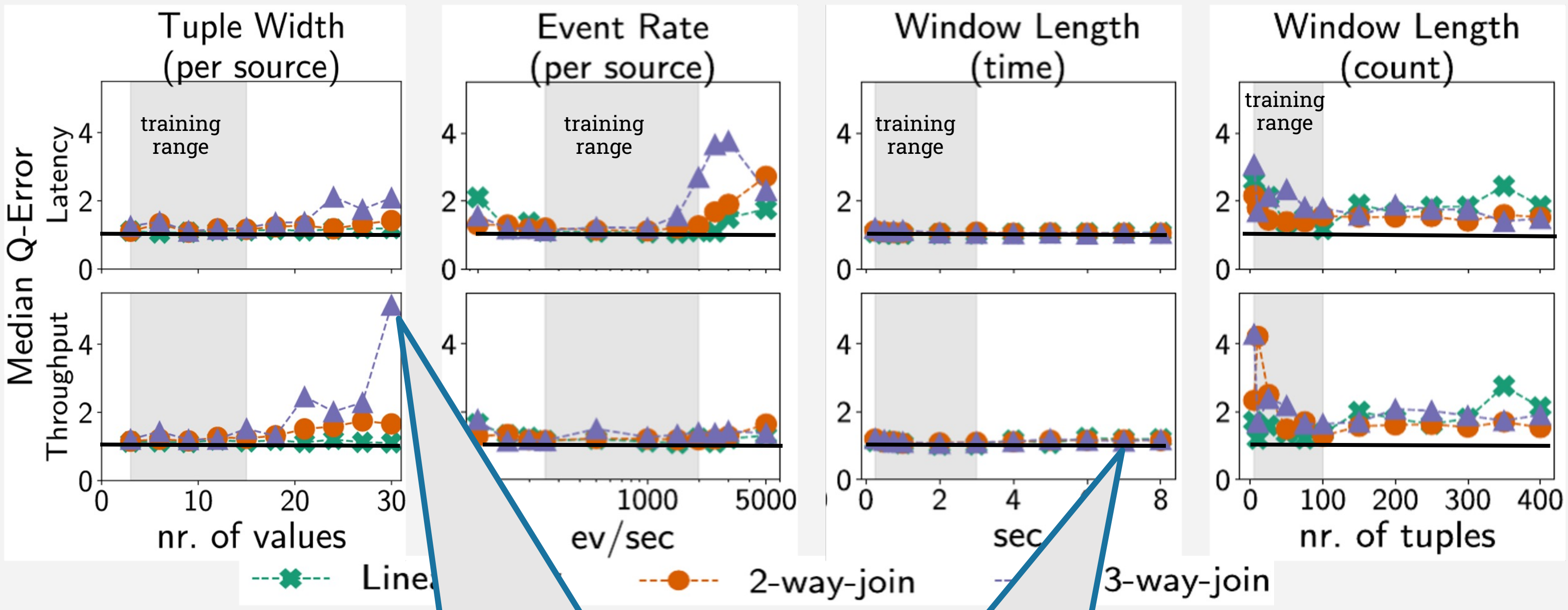
Extrapolation for **unseen benchmarks** (DSPBench[4])

Benchmark	**Latency**		**Throughput**	
	median	95th	median	95th
Advertisement (clicks)	**1.51**	1.53	**1.38**	1.39
Advertisement (imp.)	**1.51**	1.52	**1.38**	1.39
Advertisement (join)	**1.99**	2.06	**1.55**	2.16
Spike Detection	**1.01**	1.04	**1.73**	1.94
Smart Grid (local)	**1.21**	1.23	**1.92**	1.92
Smart Grid (global)	**1.20**	1.66	**1.91**	1.91

[4] M. V. Bordin, D. Griebler, G. Mencagli, C. F. R. Geyer, and L. G. L. Fernandes, "DSPbench: A suite of benchmark applications for distributed data stream processing systems", IEEE Access, vol. 8, pp. 222 900–222 917, 2020.

Zero-shot model for unseen workloads

Extrapolation for **unseen data streams & operator properties**



Increase accuracy by few-shot learning

High accuracy for unseen values

Conclusion & outlook

Our zero-shot cost model…

- …is generalizable and workload independent
- …requires an one-time training effort
- …predicts accurately and robustly for seen & unseen workloads

✅ can be used as a main building block in DSPS optimization tasks

Open questions on zero-shot models:

- How to model hardware properties more precisely?
- How to featurize co-location of operators?
- How to make use of the cost model in specific optimization tasks like providing elasticity?

Thank you for your attention!

16th ACM International Conference on
Distributed and Event-based Systems (DEBS)
Copenhagen 2022

Roman Heinrich[1], Manisha Luthra[2], Harald Kornmayer[1], Carsten Binnig[2]

[1] DHBW Mannheim, [2] TU Darmstadt

Zero-Shot model for unseen workloads

Extrapolation for **unseen query structures**

	Latency		**Throughput**	
	median	95th	median	95th
2-filter chain	**1.14**	2.41	**1.59**	3.65
3-filter chain	**2.67**	46.34	**2.82**	27.78
4-filter chain	**7.33**	54.68	**3.94**	59.73
4-way joins	**1.95**	24.30	**1.33**	20.79
5-way joins	**1.91**	26.67	**1.35**	21.87