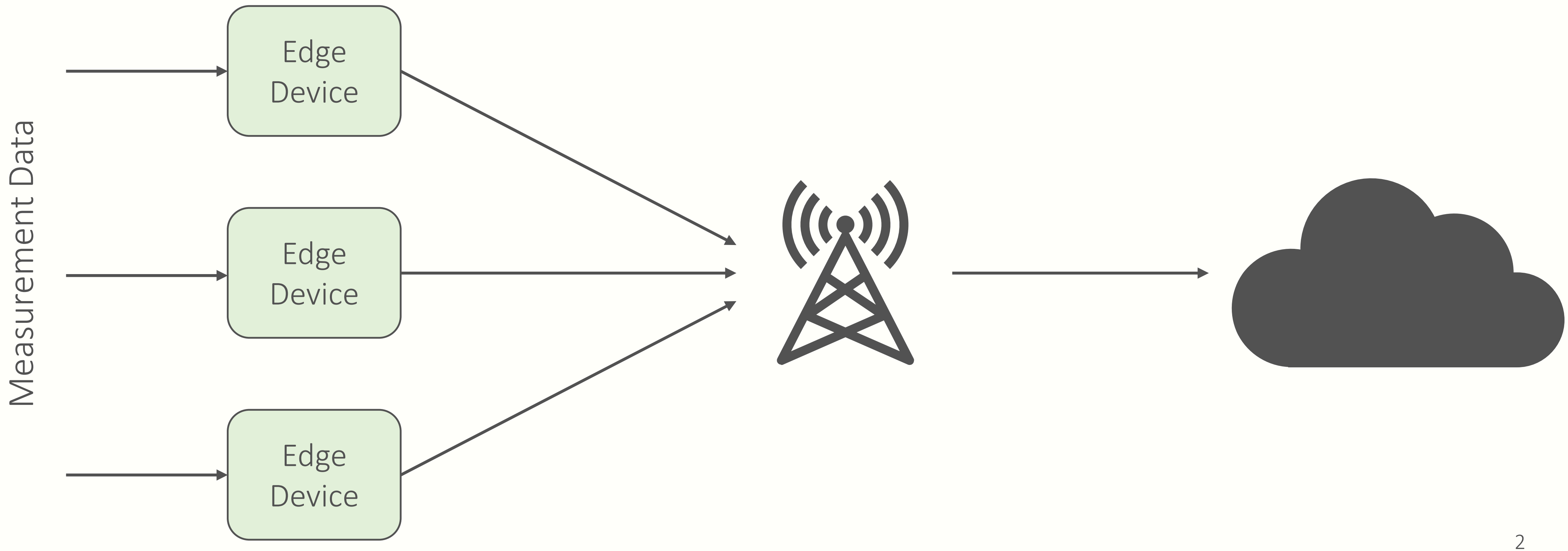
Frontier: Resilient Edge Processing for the Internet of Things

Dan O'Keeffe, Theodoros Salonidis, and Peter Pietzuch

Presented by Tejas Kannan, 31/10/2018

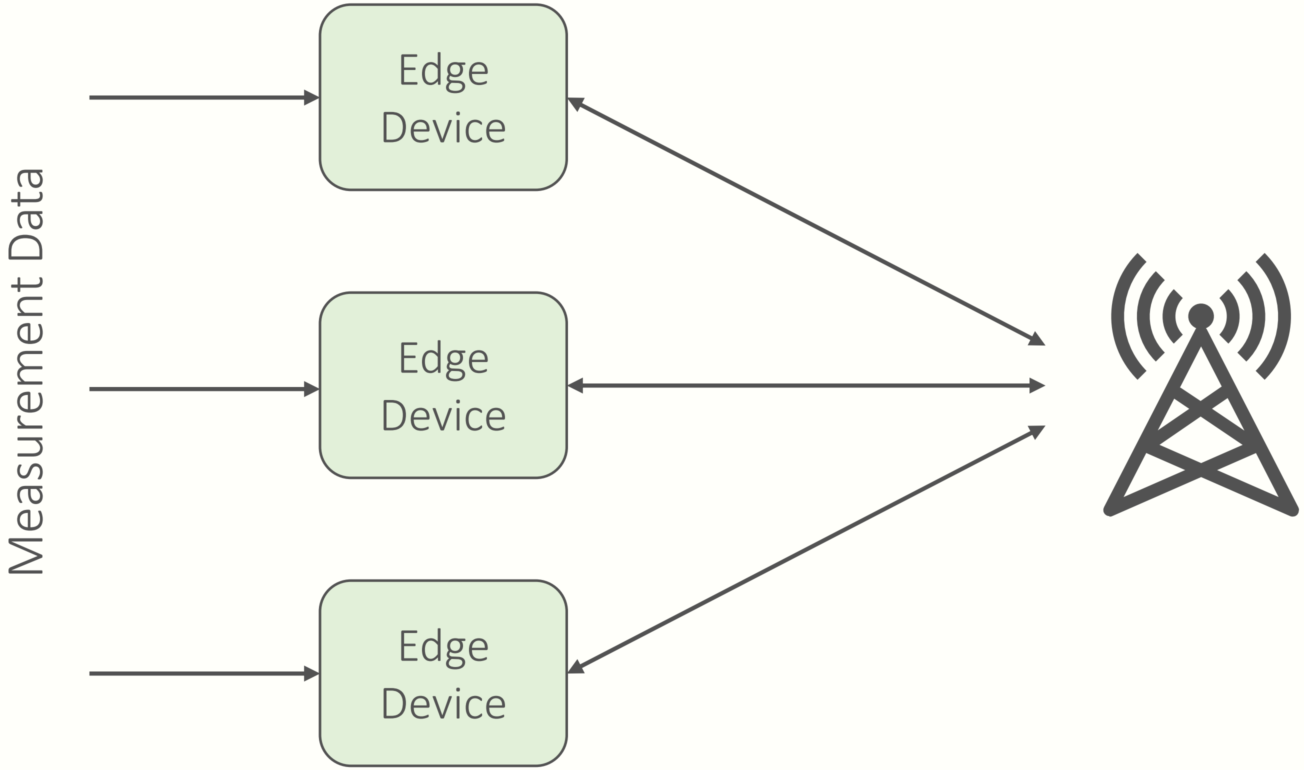
Motivation

IoT systems often offload stream computation to the cloud



Motivation

Increased CPU and memory capabilities of modern devices enables processing to occur without offloading to the cloud

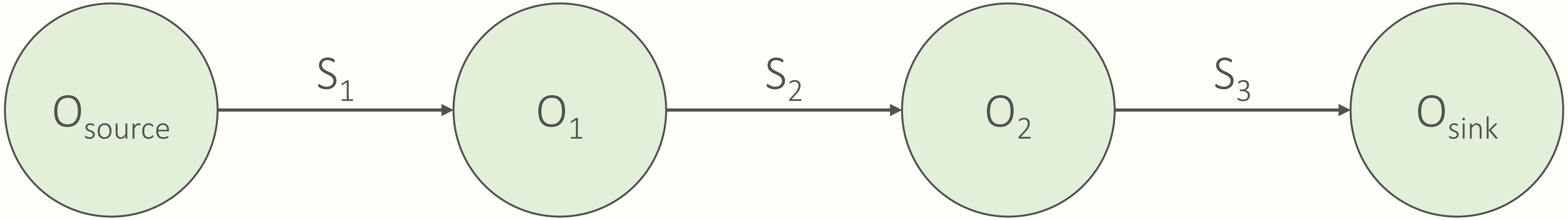


Requirements of an Edge Deployment Model

Requirement	Frontier's Solution
Continuously process streaming data	Move computation to edge devices
Data-parallel processing	Replicate data operators
Adapt to changing network conditions	Backpressure Stream Routing (BSR)
Recover from transient network failures	Selective Network Aware rePlay (SNAP)

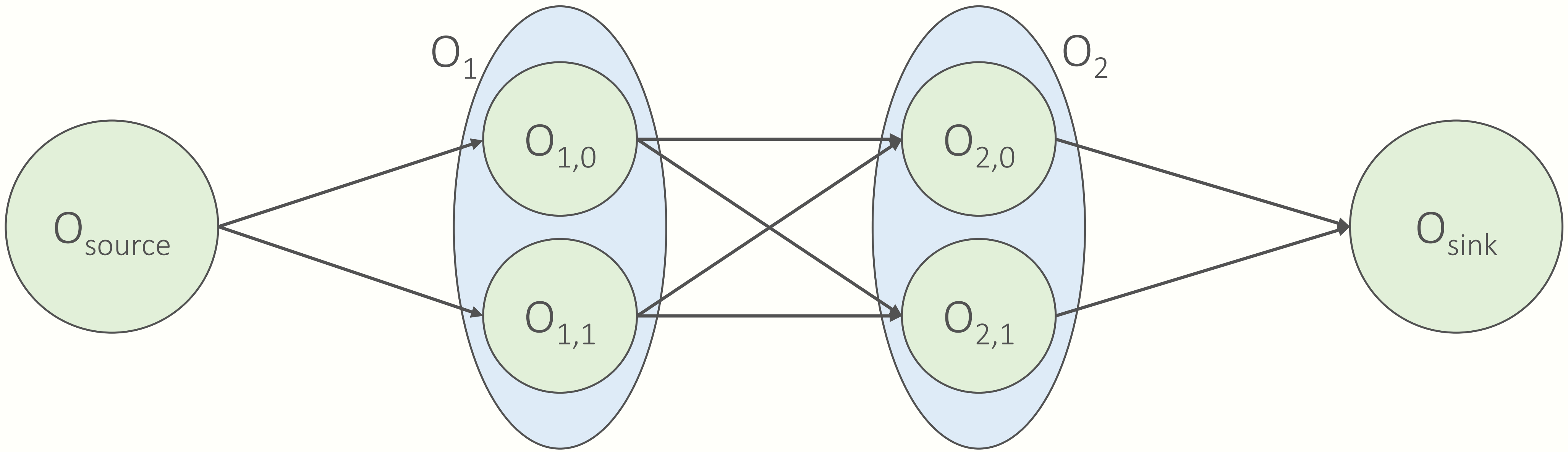
Stream Query Computational Model

Queries can be represented by a directed graph where vertices are operations and edges are streams



Frontier's Replicated Dataflow Graph

Each replica can be placed on a different edge device to enable better data parallelism



Properties of Stream Query Routing

1. Window-Based: Operators may act on windows of streams

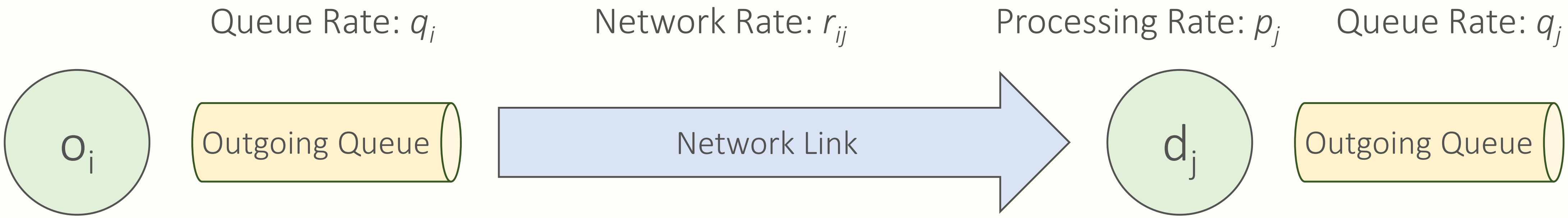
2. Out-of-order: Processing must be able to cope with out-of-order delivery to maintain high throughput

3. Multi-Input: Operators may accept multiple input streams, so windows from input streams must be sent to same replica

4. Batched Windows: Stream windows may be batched to reduce network communication

Backpressure Stream Routing

Backpressure routing enables adaptability to network conditions

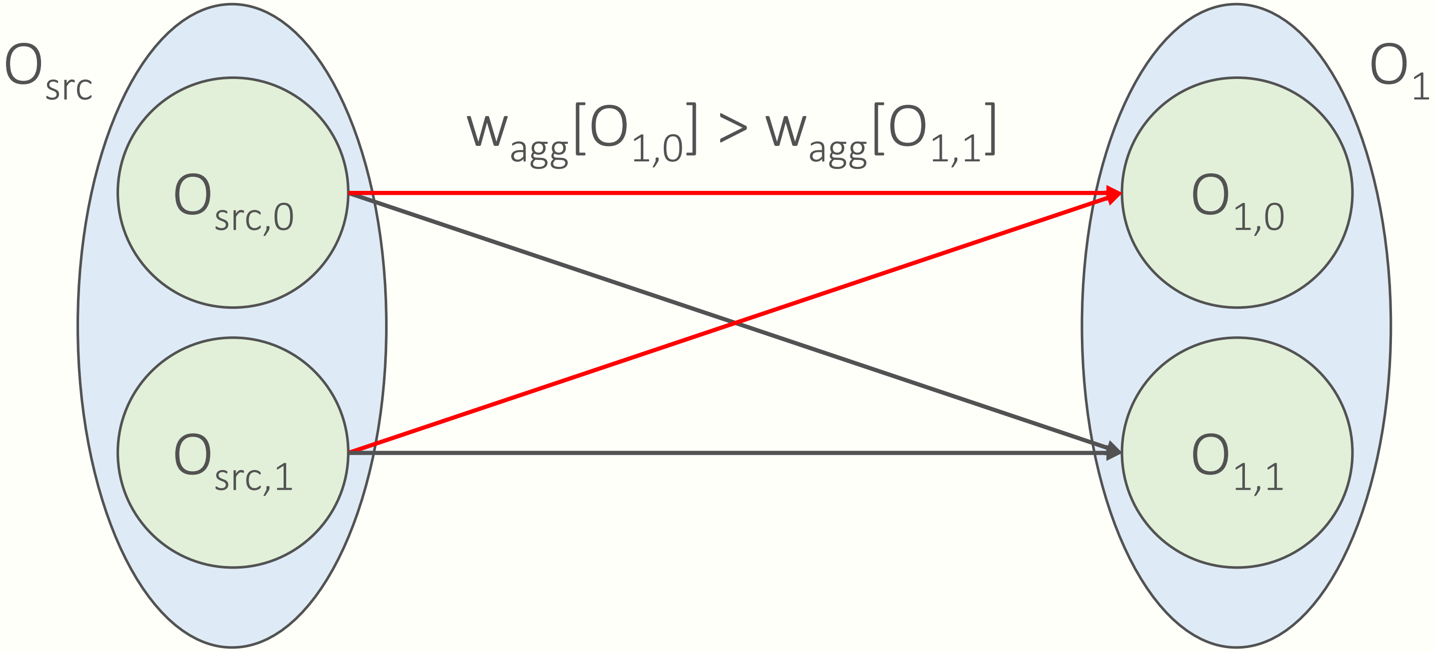


Weight of link is $w_{ij} = \max(0, (q_i - q_j) \times r_{ij} \times p_j)$

Downstream replica with highest weight is chosen

Backpressure Stream Routing

To coordinate windows for multi-input operators, routing is based on aggregate weights over all destination replicas



Routing constraints are used to handle out of order arrival

Selective Network-Aware Replay (SNAP)

Batches are buffered by senders and re-sent upon discovery of a single failed node

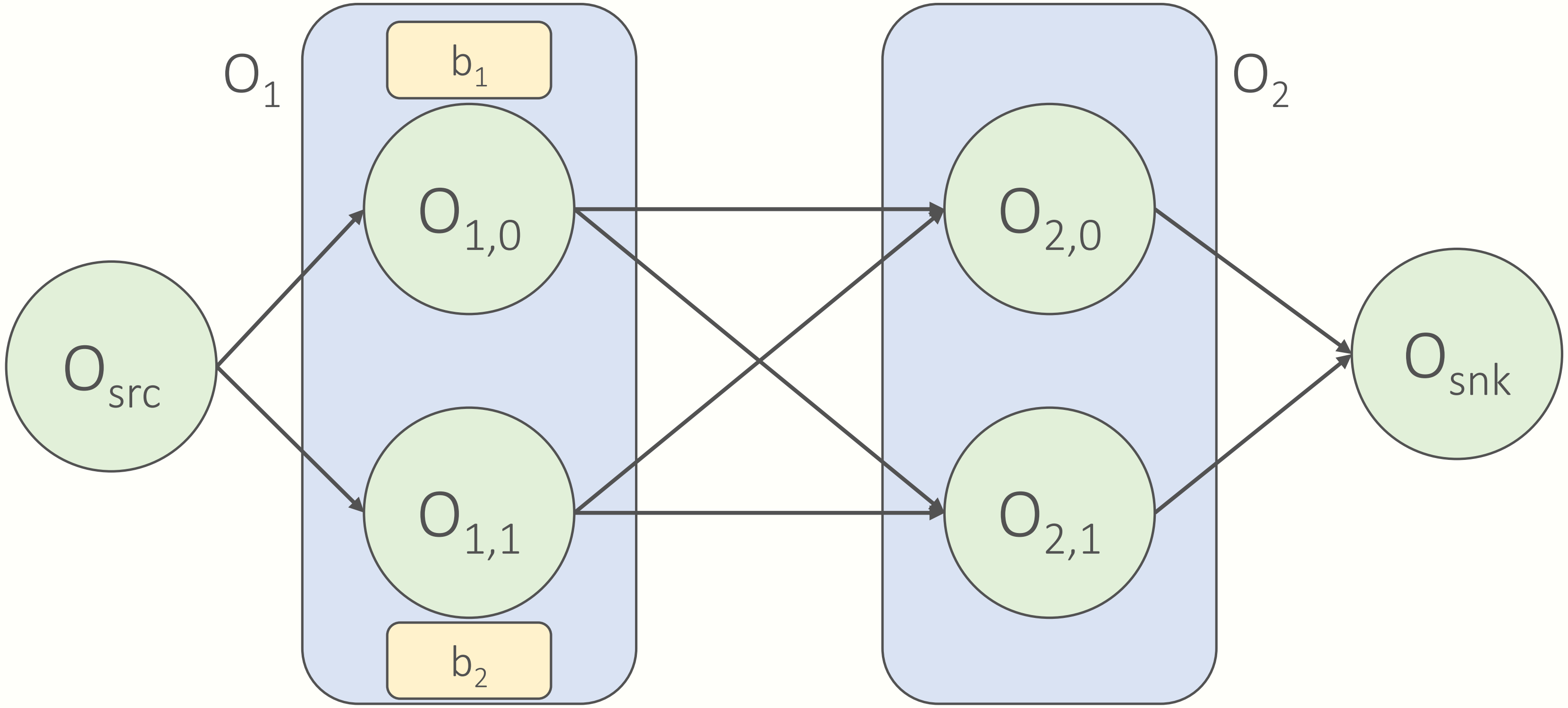


Diagram from [4]

Selective Network-Aware Replay (SNAP)

Batches are buffered by senders and re-sent upon discovery of a single failed node

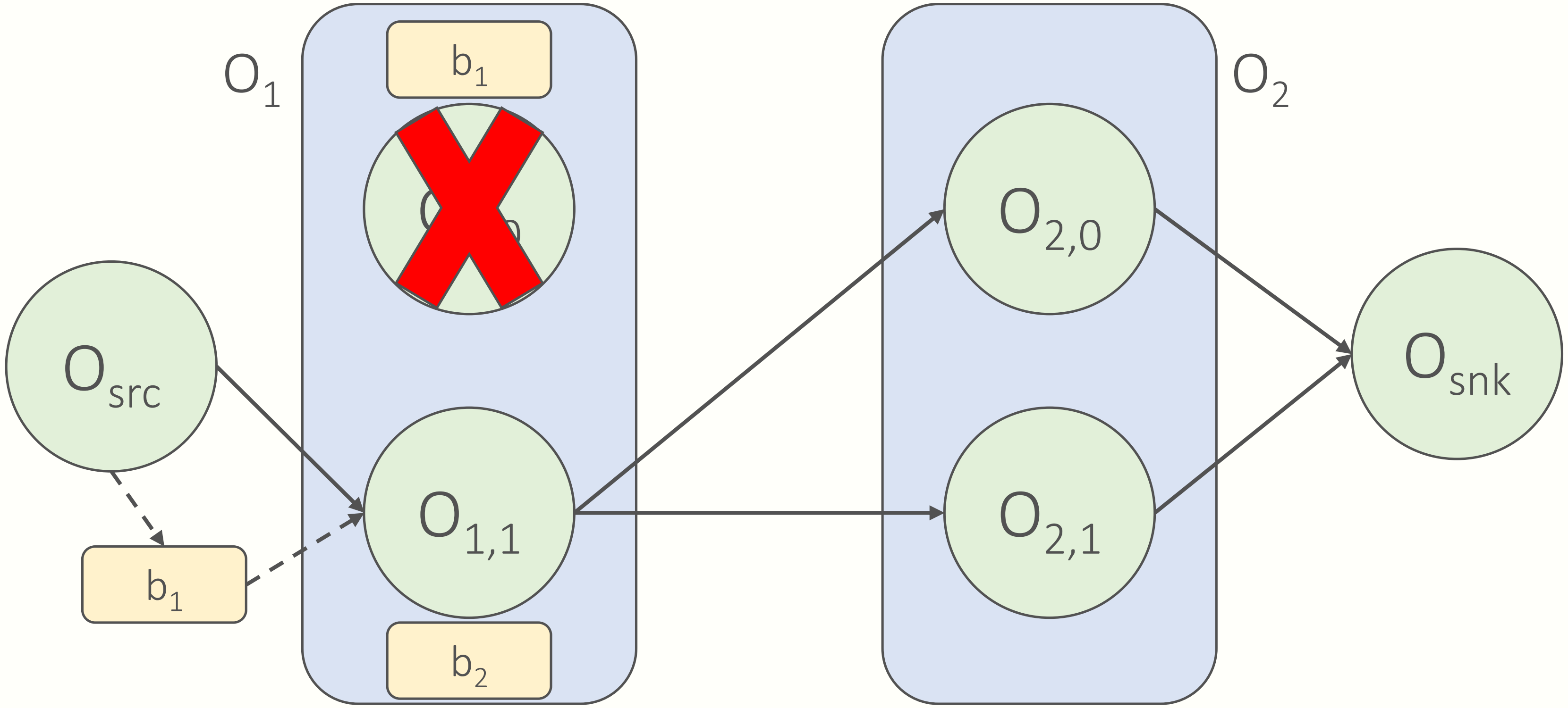


Diagram from [4]

Selective Network-Aware Replay (SNAP)

Heartbeat messages indicate which batches have been sent downstream, batches not replayed if heartbeats continue to be sent

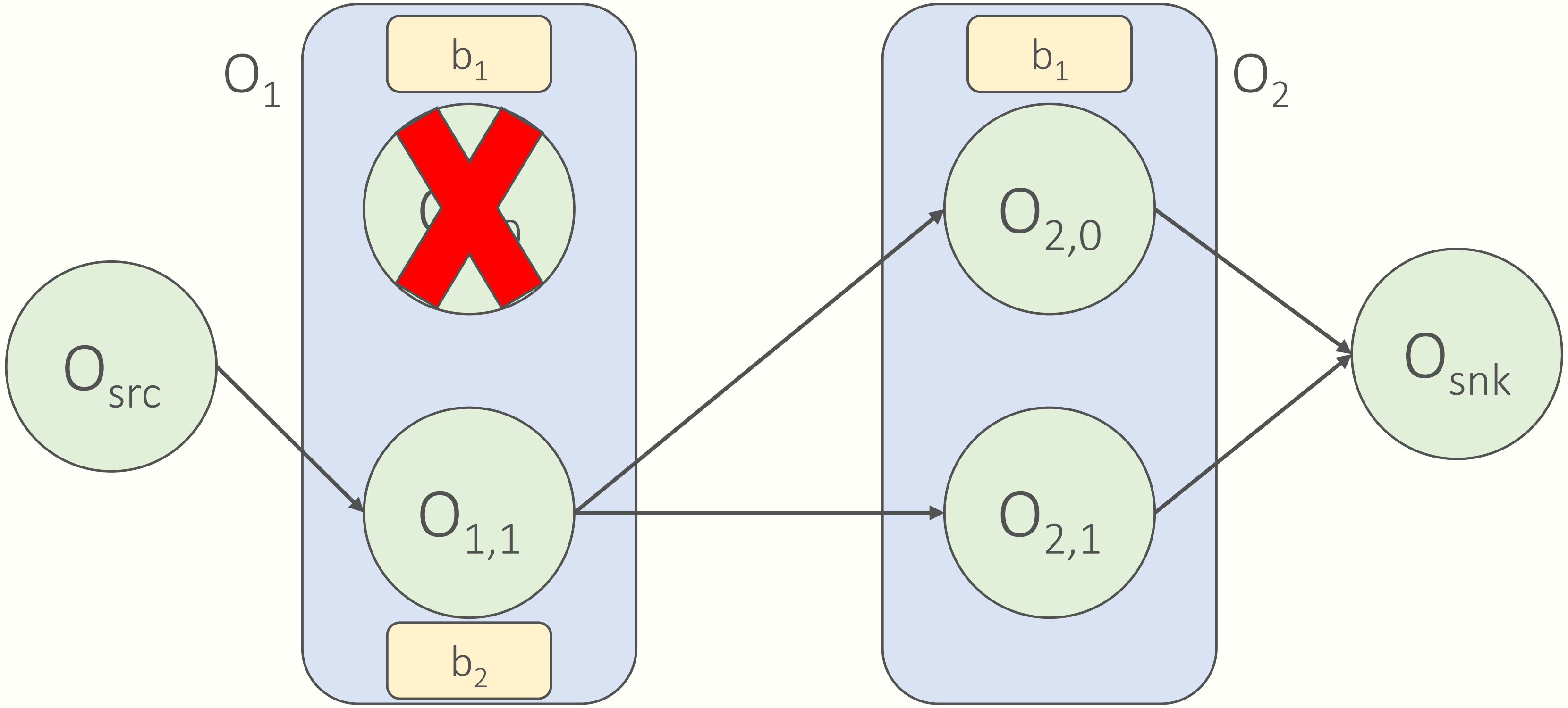
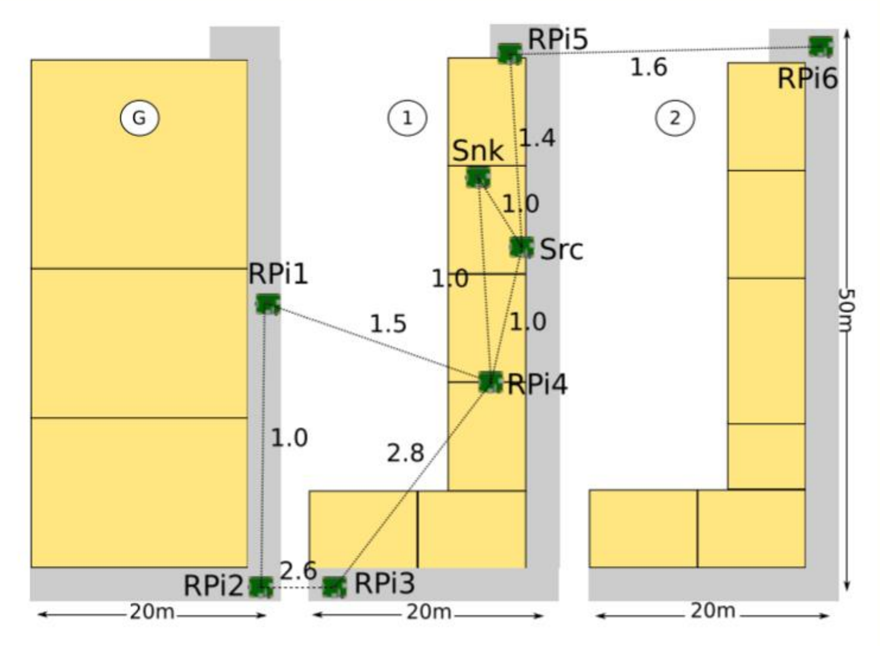


Diagram from [4]

Experimental Setup

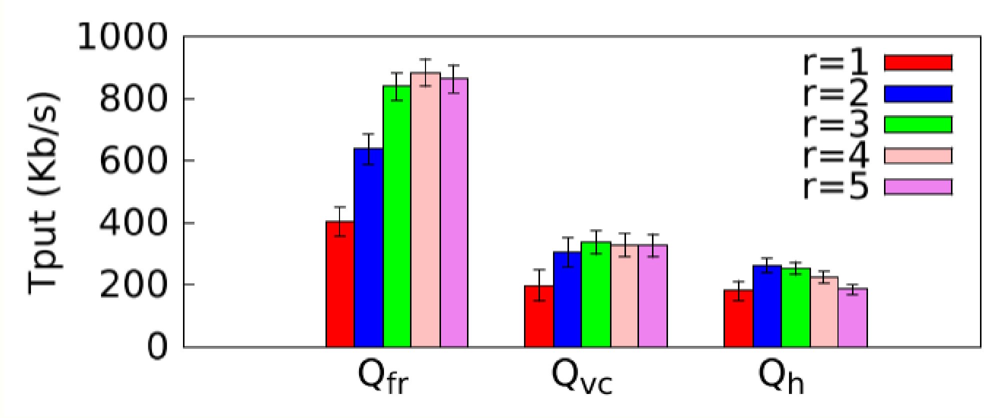
- Experiments run on a wireless network of Raspberry Pi's
- Two different networks created: high and low diversity
- CORE/EMANE [3] wireless network emulator also used
- Three different queries: Distributed Face Recognition, Video Correlation, Heatmap of Users in Area



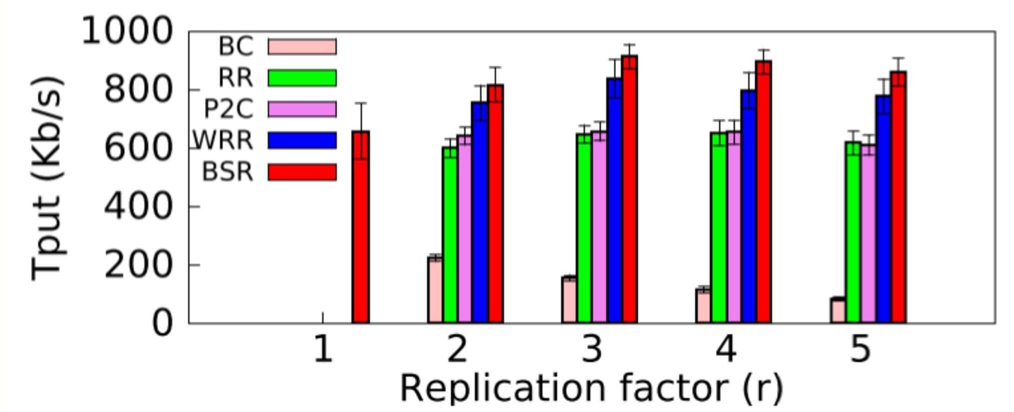
High-diversity Mesh Network

Experiments: Throughput

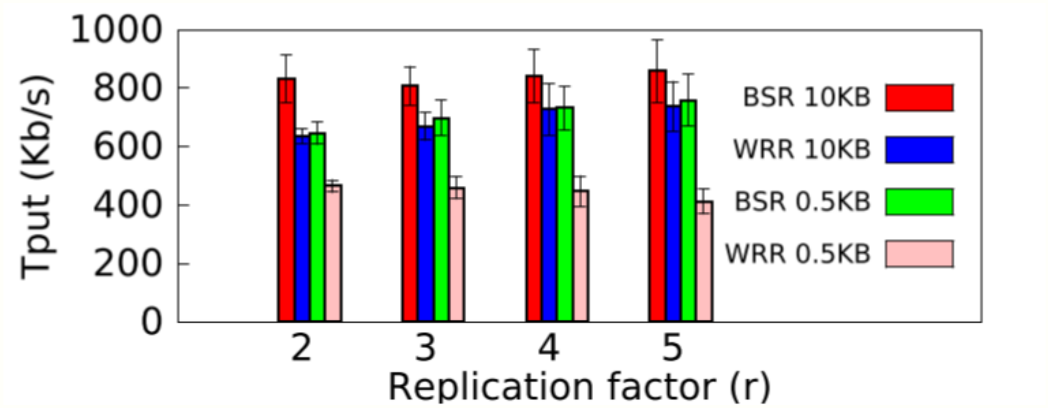
Varying Replication Factor



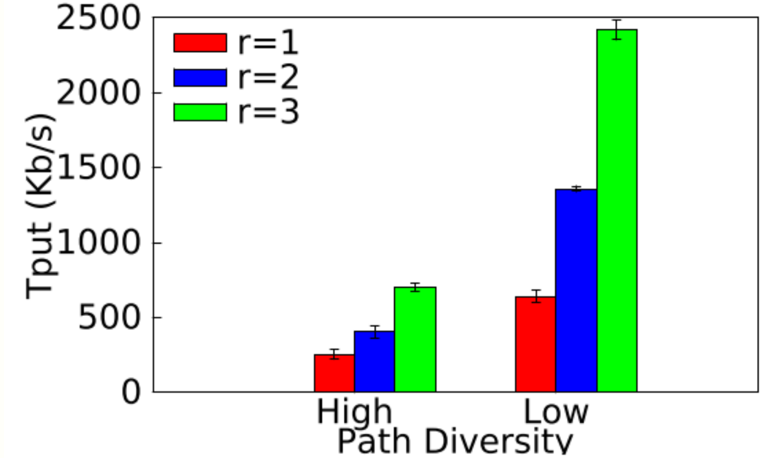
BSR Compared to Baselines



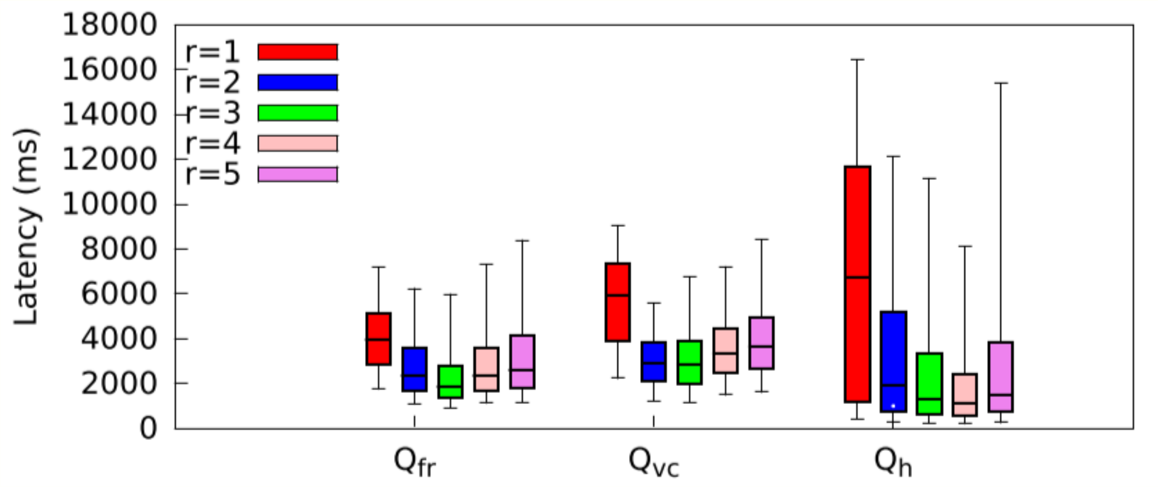
Varying Replication Factor and Batch Size



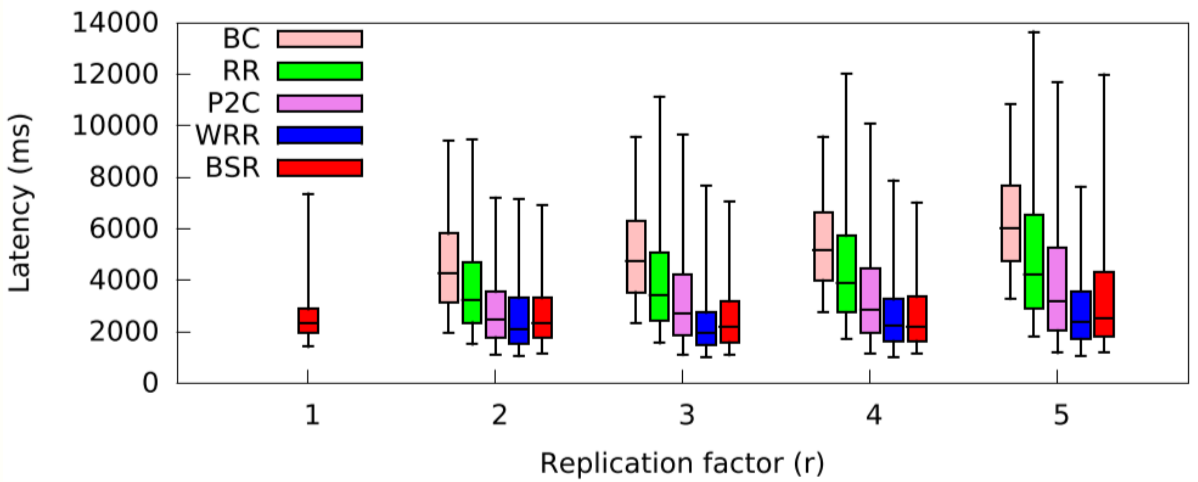
BSR Path Diversity with Different Replication



Experiments: Latency



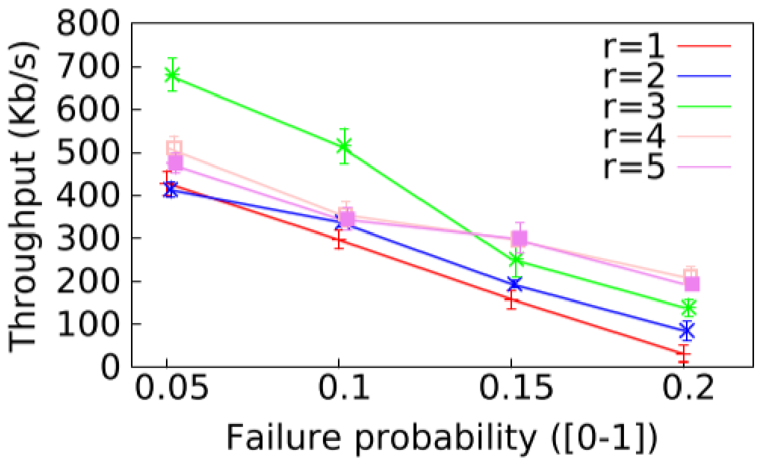
Latency with Varying Replication Factor
(Error bars show 5/25/50/75/95 percentiles)



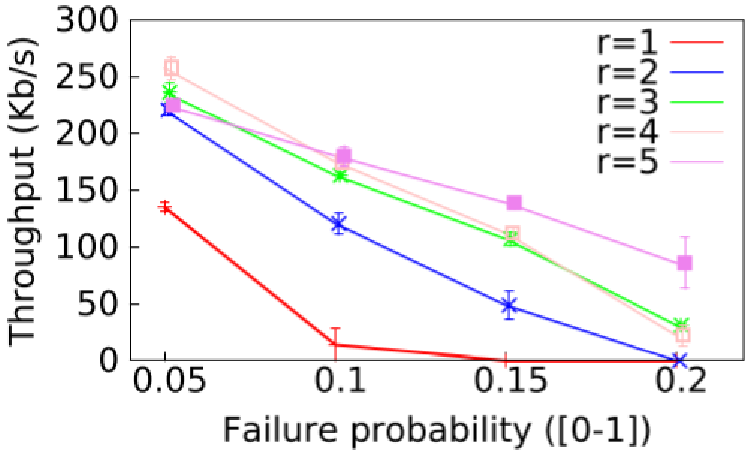
Latency of BSR Compared to Baselines

Experiments: Recovering from Failure

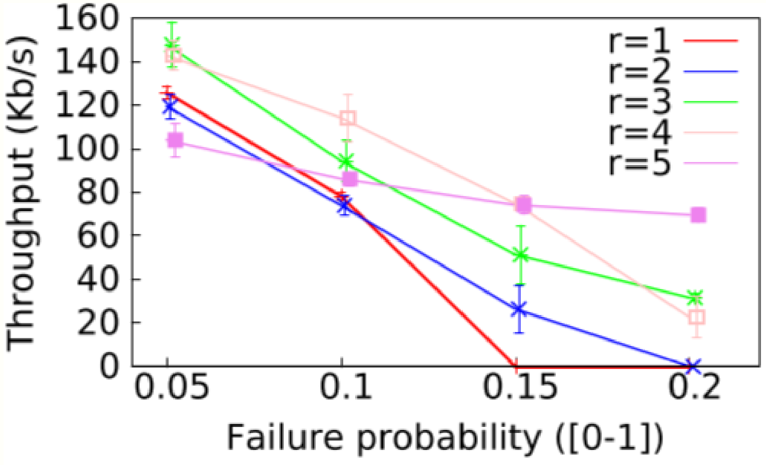
Larger replication factors generally lead to higher throughput in the face of failure



Distributed Face Recognition



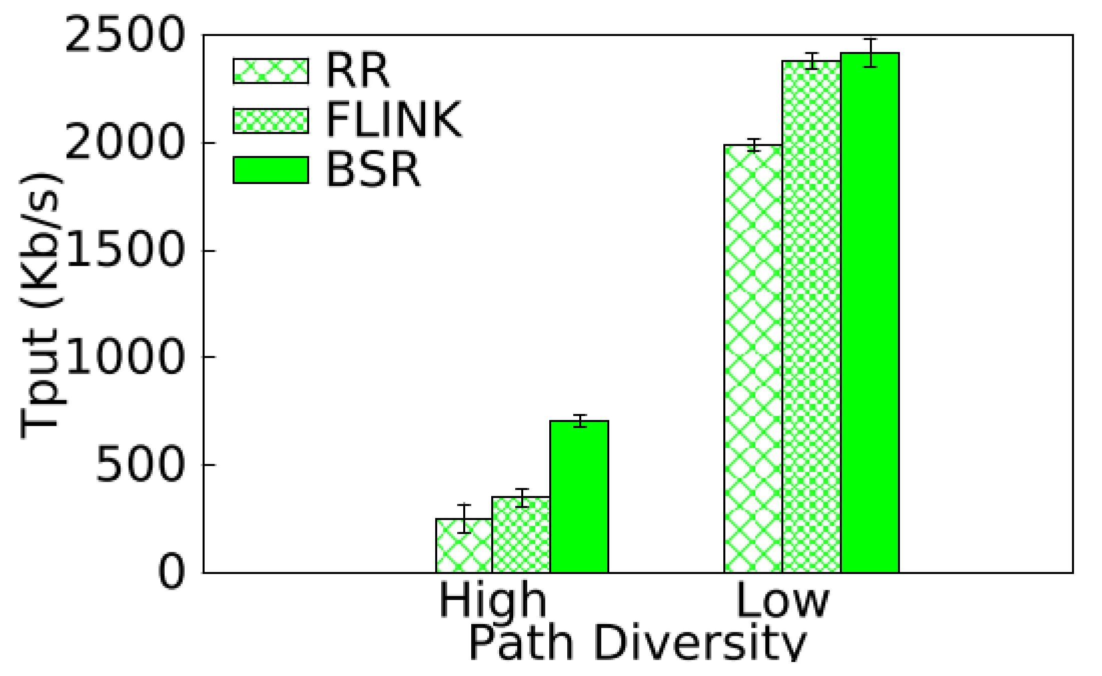
Video Correlation



Heatmap

Experiments: Frontier vs. Apache Flink

Frontier generally exhibits a higher throughput than that of other stream processing systems



Frontier vs Flink [1] and Round Robin on Face Recognition Query

Critique

- Method of computing aggregate BSR weights is never explained
- Experimentation is robust but minimal comparison of Frontier to other platforms
- Possible area of future work: making BSR predictive of network conditions could ease tension between using larger batches and updating network parameters

Related Work

Platform Name	Description	How Frontier is Different
Spark Streaming [7]	Cluster-based, structures computation as stateless batch computations	Spark Streaming assumes wired connections between nodes
SBON [5]	Manages operator placement to efficiently use network resources	SBON does not replicate operators and use backpressure to load balance on these replicas
CSA [6]	Stream processing for IoT systems which relies on single nodes on network edge	CSA does not distribute computation across devices

Conclusion

- Frontier is a stream evaluation platform which performs computation on edge devices
- Achieves data-level parallelism by replicating operators and distributing execution across devices
- Uses network-aware routing to efficiently use resources in wireless settings
- Recovers from transient errors without causing network congestion

References

[1] Apache flink. https://flink.apache.org/.

[2] Apache storm. https://storm.apache.org/.

[3] Jeff Ahrenholz. Comparison of core network emulation platforms. In *Military Communications Conference, 2010-MILCOM 2010*, pages 166–171. IEEE, 2010.

[4] Dan O'Keeffe, Theodoros Salonidis, and Peter Pietzuch. Frontier: resilient edge processing for the internet of things. *Proceedings of the VLDB Endowment*, 11(10):1178–1191, 2018.

[5] Peter Pietzuch, Jonathan Ledlie, Jeffrey Shneidman, Mema Roussopoulos, Matt Welsh, and Margo Seltzer. Network-aware operator placement for stream-processing systems. In *Data Engineering*, 2006. ICDE'06. IEEE, 2006.

[6] Zhitao Shen, Vikram Kumaran, Michael J Franklin, Sailesh Krishnamurthy, Amit Bhat, Madhu Kumar,Robert Lerche, and Kim Macpherson. Csa: Streaming engine for internet of things. *IEEE Data Eng.Bull.*, 38(4):39–50, 2015.

[7] Matei Zaharia, Tathagata Das, Haoyuan Li, Timothy Hunter, Scott Shenker, and Ion Stoica. Discretized streams: Fault-tolerant streaming computation at scale. *In Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, pages 423–438. ACM, 2013.