CSC0056: Data Communication

Real-Time Data Communications

Instructor: Chao Wang 王超

Department of Computer Science and Information Engineering

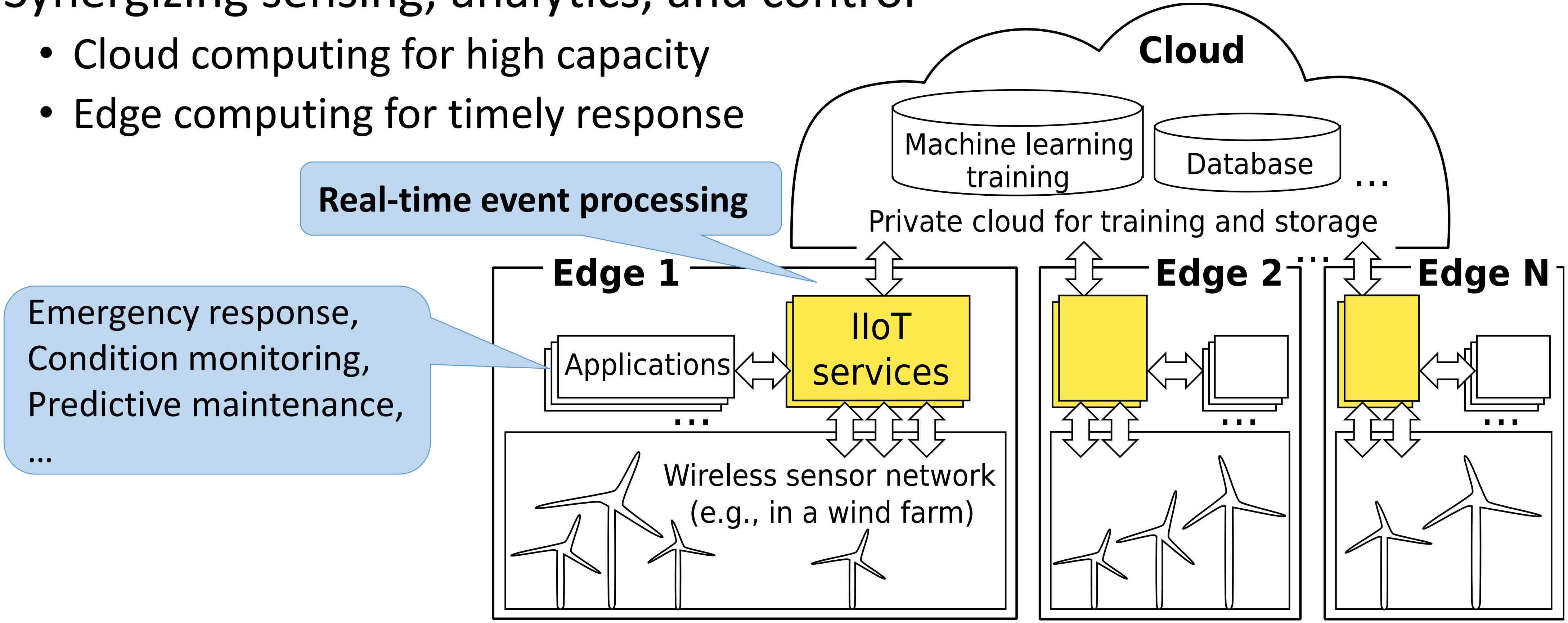
NATIONAL TAIWAN NORMAL UNIVERSITY

References

- Industrial Internet Reference Architecture v1.9 (https://www.iiconsortium.org/IIRA.htm)
- Chao Wang, Christopher Gill, and Chenyang Lu. *Real-Time Middleware for Cyber-Physical Event Processing*. ACM Transactions on Cyber-Physical Systems 3, 3, Article 29 (August 2019) (https://wangc86.github.io/pdf/tcps-cpep.pdf)
- Gomaa, Hassan. *Real-Time Software Design for Embedded Systems.* Cambridge University Press, 2016.

Industrial Internet of Things (IIoT)

- Synergizing sensing, analytics, and control
  - Cloud computing for high capacity
  - Edge computing for timely response



Low-latency data communications

- Low latency is an essential feature in many networked applications
- Example: emergency notification
  - Fire
  - Flood
  - _______________

- Example: acute weather prediction and notification
  - Earthquake and tsunami
  - Volcanic eruption
  - Tornado

Low-latency $\rightarrow$ Real-time

- Conceptually, in data communication, fast enough is good enough
- *Deadline:* a way to specify what we meant by fast enough
- Soft deadlines vs. hard deadlines
  - Missing a soft deadline is not desirable but may be acceptable
  - Missing a hard deadline will lead to disastrous consequences
- Soft/hard real-time system: a system that meets soft/hard deadlines

Real-time data communications

- For networked applications, people often specify *end-to-end* deadline for data communications
  - From “data created by a sensor” (one end)
    to “data received by an application” (the other end)

- From one end of the system to the other end:
  - Sender
  - Link(s) between sender and intermediary
  - Intermediary (messaging broker/event service/edge computing)
  - Link(s) between intermediary and receiver
  - Receiver

Real-time data communications

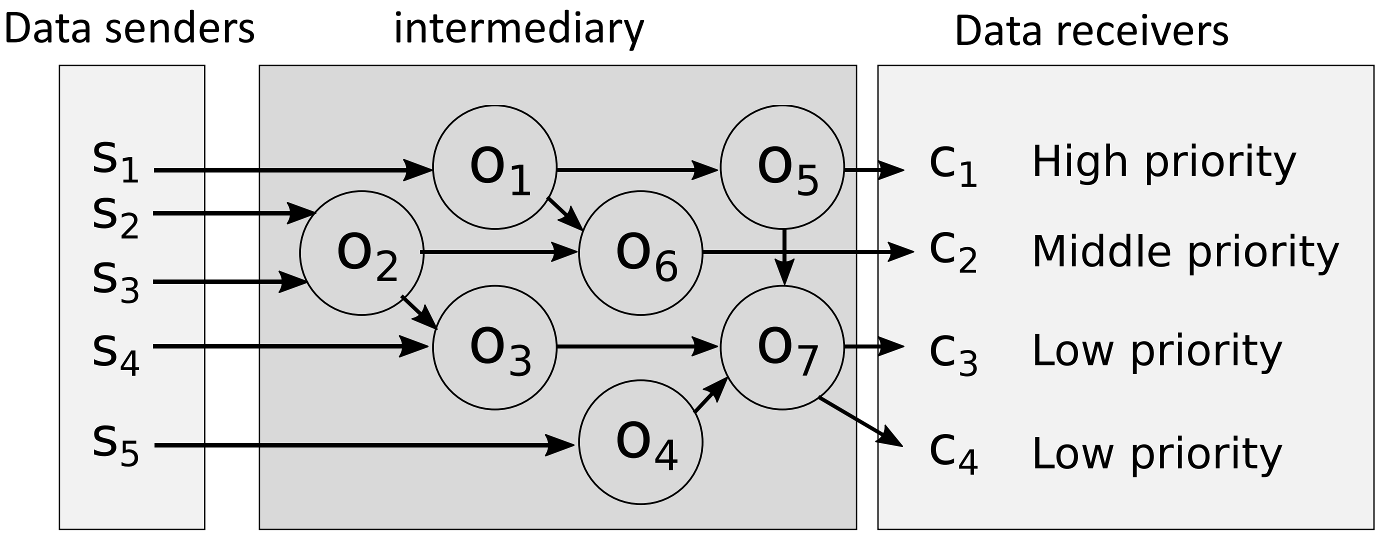
- For networked applications, people often specify *end-to-end* deadline for data communications
  - From "data created by a sensor" (one end)
    to "data received by an application" (the other end)

- From one end of the system to the other end:
  - ~~Sender~~
  - ~~Link(s) between sender and intermedi~~
  - **Intermediary**
  - ~~Link(s) between intermediary and receiv~~
  - ~~Receiver~~

We may assume that links are reliable and have bounded latency (using what we have learned in this course, for example).

Data communication intermediaries

- Purposes:
  - Decoupling senders and receivers
  - Simplifying senders and receivers
- Example intermediary: TAO, MQTT, NSQ, …



O_i: operations (filtering, transformation, encryption, …)

Outline of lecture15

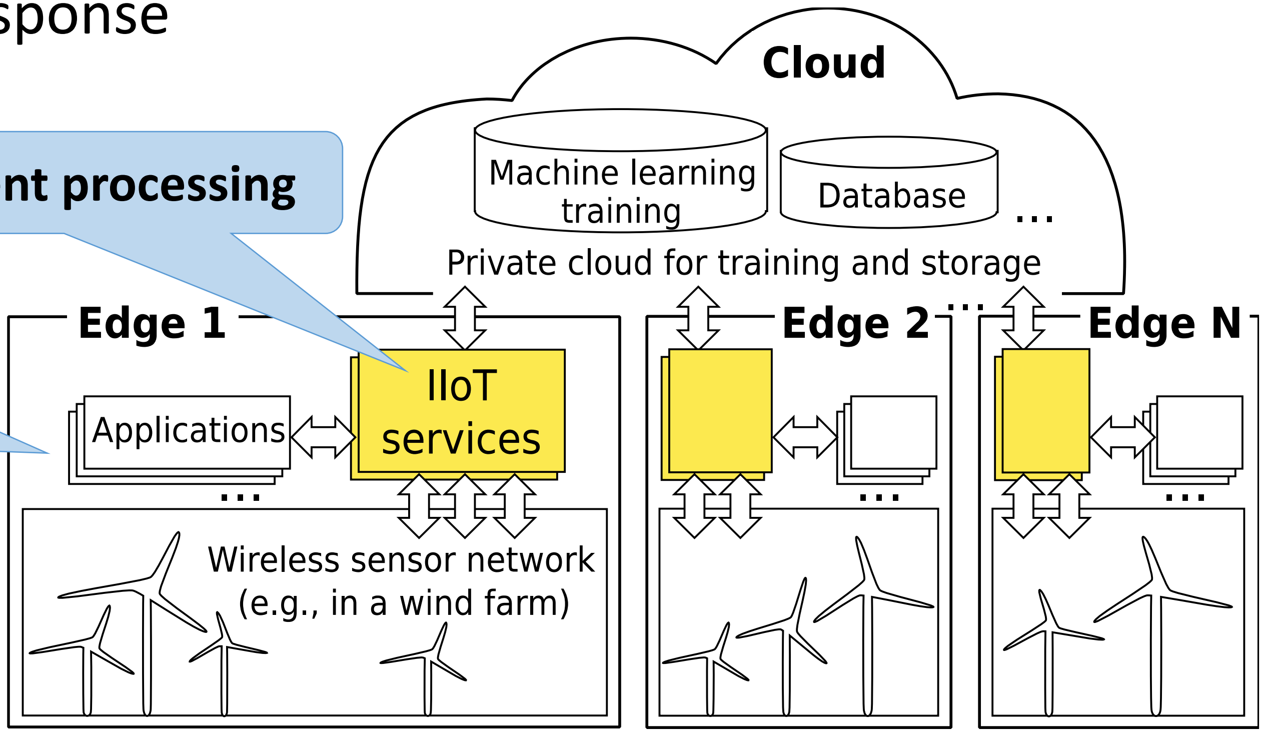
- A review of flow control (Section 6.5.1 in particular)
- Timing aspects of data communications
- Real-time data communications
- **Case study: Real-time event processing**

Industrial Internet of Things (IIoT)

- Synergizing sensing, analytics, and control
  - Cloud computing for high capacity
  - Edge computing for timely response

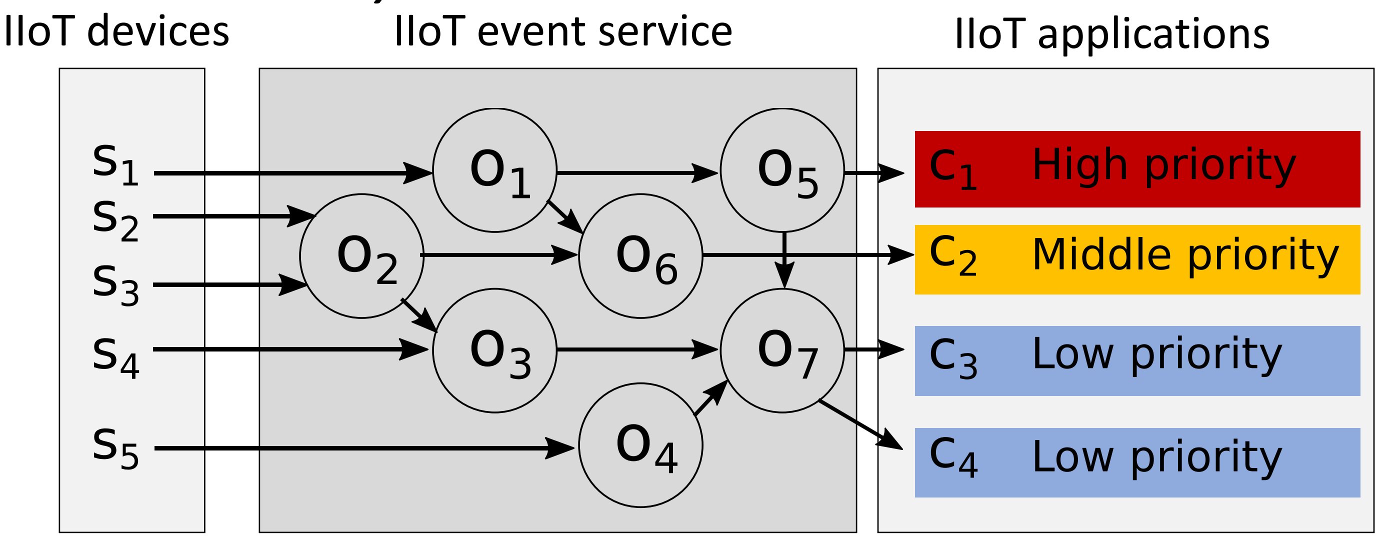
Real-time event processing

Emergency response,
Condition monitoring,
Predictive maintenance,
…



A model for event processing

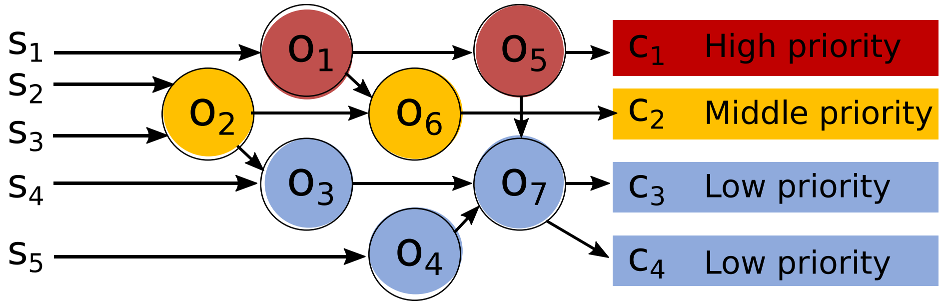
- Latency requirements
- Temporal semantics:
  - *Absolute time consistency* on an event's elapse time since creation
  - *Relative time consistency* on the difference between event's creation time



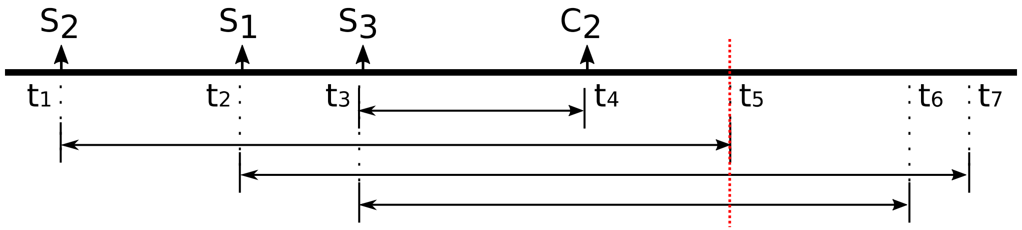
O_i: operations (filtering, transformation, encryption, …)

Real-time cyber-physical event processing

- Processing in the order of priorities propagated from application:

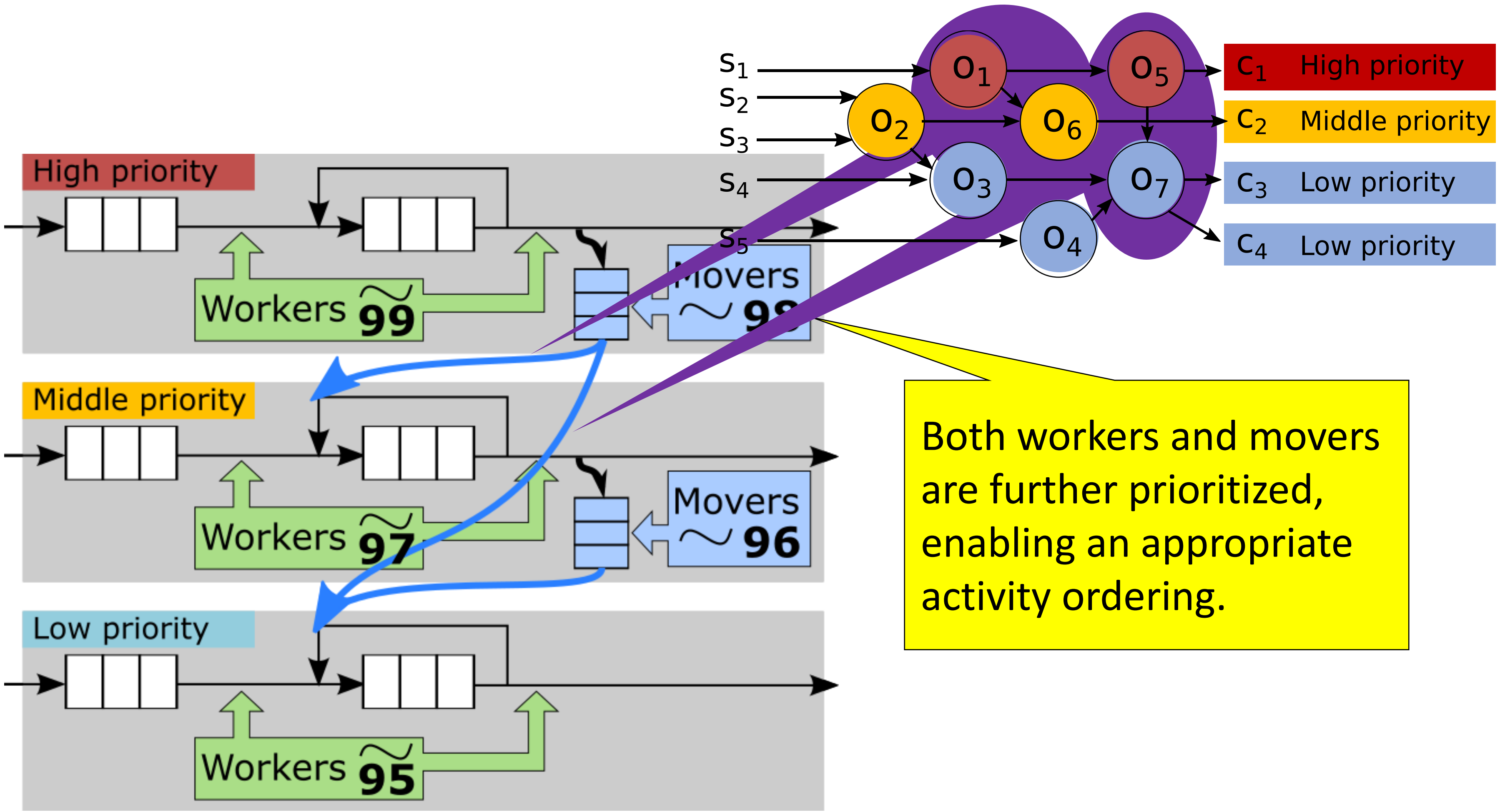


- Temporal semantics enforcement
  - Absolute time consistency



  - Relative time consistency: track both the earliest and the latest event creations per operator

Processing architecture



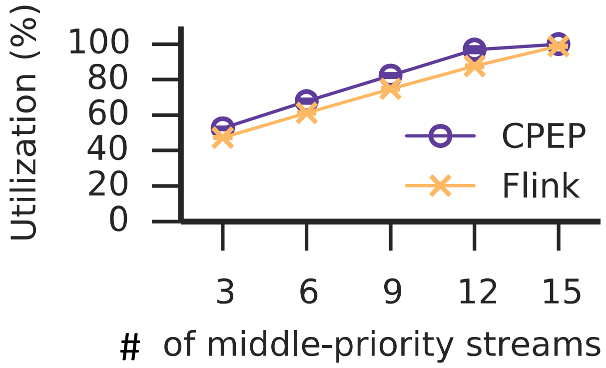
Both workers and movers are further prioritized, enabling an appropriate activity ordering.

Latency performance

99th percentile latency (unit: ms)

Priority	Service	Number of middle-priority streams				
		3	6	9	12	15
High	Flink	3.8 ± 0.1	5.9 ± 0.2	12.6 ± 0.4	52.6 ± 4.1	448.9 ± 171.7
	CPEP	0.8 ± 0.0	0.7 ± 0.0	0.7 ± 0.0	0.7 ± 0.0	0.7 ± 0.0
Middle	Flink	4.5 ± 0.1	6.4 ± 0.2	11.3 ± 0.4	28.9 ± 0.5	107.9 ± 18.1
	CPEP	1.6 ± 0.0	1.8 ± 0.0	2.2 ± 0.0	2.5 ± 0.0	3.0 ± 0.1
Low	Flink	5.2 ± 0.3	7.4 ± 0.2	15.5 ± 0.6	43.3 ± 1.3	679.8 ± 274.0
	CPEP	3.7 ± 0.3	4.8 ± 0.2	6.8 ± 0.6	10.6 ± 1.0	33.4 ± 0.2

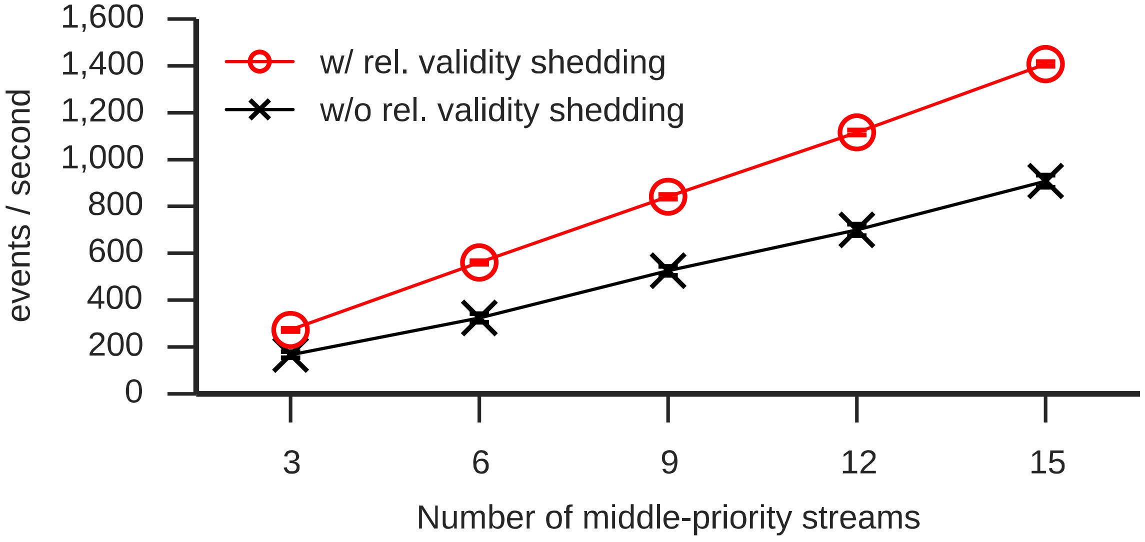
CPEP maintained high-priority latency performance as workload increased.



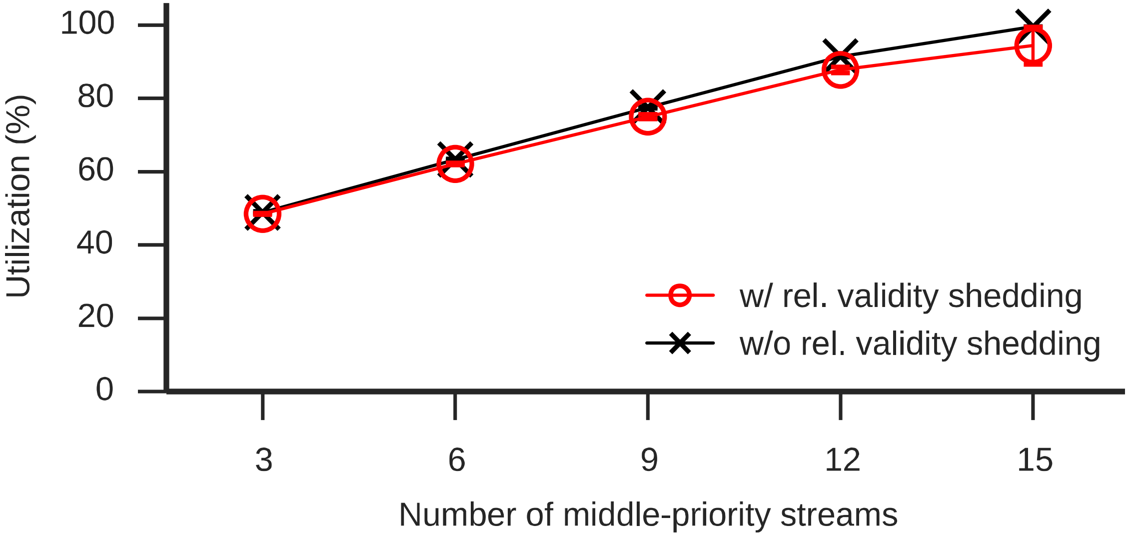
CPEP differentiated latency according to priority level.

Benefits of shedding inconsistent events

Improve the throughput of consistent events.



Save CPU utilization.



Dependability issues in computing systems

- What if some of the network components may fail to work?
- How to keep applications running properly while fixing the underlying network problems?
- Will fault-tolerance affect the performance of a data network? In which ways? How to amend it?