StreamBox: Modern Stream Processing on a Multicore Machine

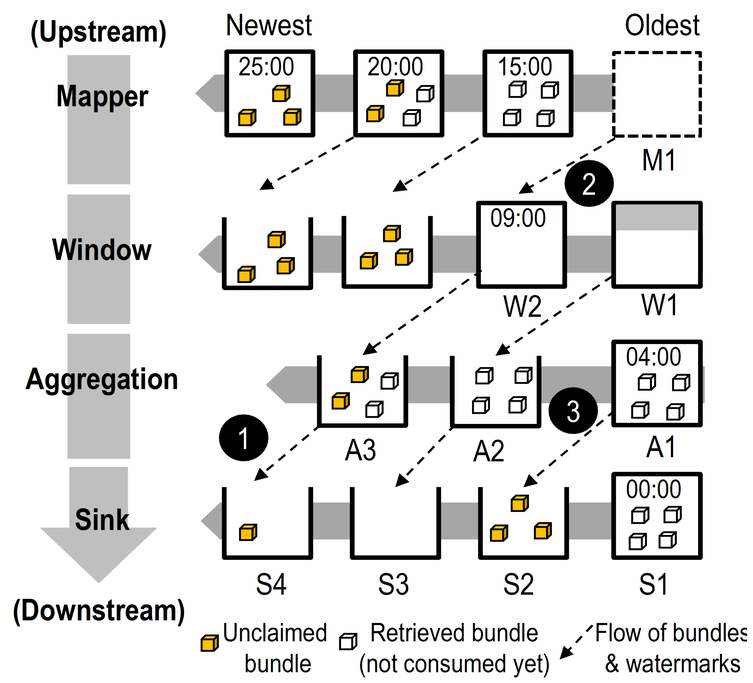
Hongyu Miao*, Heejin Park*, Myeongjae Jeon^, Gennady Pekhimenko^, Kathryn S. McKinley#, and Felix Xiaozhu Lin*
***Purdue ECE** **^Microsoft Research** **#Google**
xzl@purdue.edu

USENIX ATC '17
JULY 12–14, 2017
SANTA CLARA, CA
www.usenix.org/atc17

1. Motivation

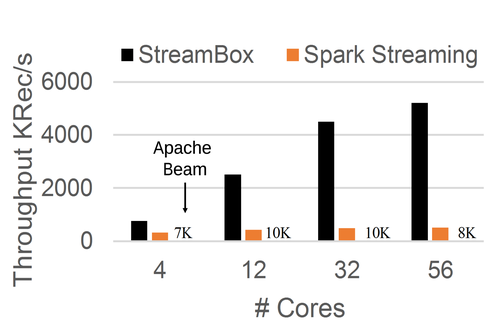
- Single multicore machine for stream processing:
  - Terabyte DRAM, large numbers of cores, and fast I/O
- Challenges of stream processing on a multicore machine:
  - Handling out-of-order input data
  - Exploiting parallelism to harness tens of CPU cores
  - Exploring memory hierarchy to minimize data move
  - Achieving both high throughput and low latency

2. Key Mechanism: Cascading Containers for Processing Stream Epochs in Parallel

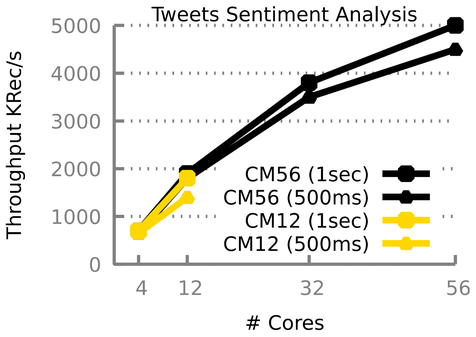


3. Key Results

- Built StreamBox from scratch in 23K SLoC C++
- Designed *Cascading Container* mechanism for processing out-of-order stream in high concurrency
- Achieved both high throughput and low latency -- 20x lower than popular large-scale streaming engines



Superior multicore performance compared to popular streaming engines



Good multicore scalability

Test platforms:
CM56: Dell PowerEdge R930. 4x Intel Xeon E7-4850v4 (14C/28T); 256GB DRAM; 1TB SSD
CM12: Dell PowerEdge R720. 2x Intel Xeon E5-2630v2 (6C); 256GB DRAM; 4x 3TB SAS HDD

4. Ongoing and Future Work

- Optimizing streaming operator performance
- Making StreamBox dataflow NUMA-friendly
- Exploiting heterogeneous memory architecture, e.g. Intel Knights Landing
- Guaranteeing data security, e.g. confidentiality and integrity, during stream processing

PURDUE UNIVERSITY
Microsoft Research
Google
NSF

http://xsel.rocks