ELF: Efficient Lightweight Fast Stream Processing at Scale

Liting Hu, Hrishikesh Amur, Karsten Schwan and Xin Chen
College of Computing, Georgia Tech
{foxting, amur, karsten, xchen384}@gatech.edu

How to process large-scale distributed streams with low latency and high throughput?

WHY

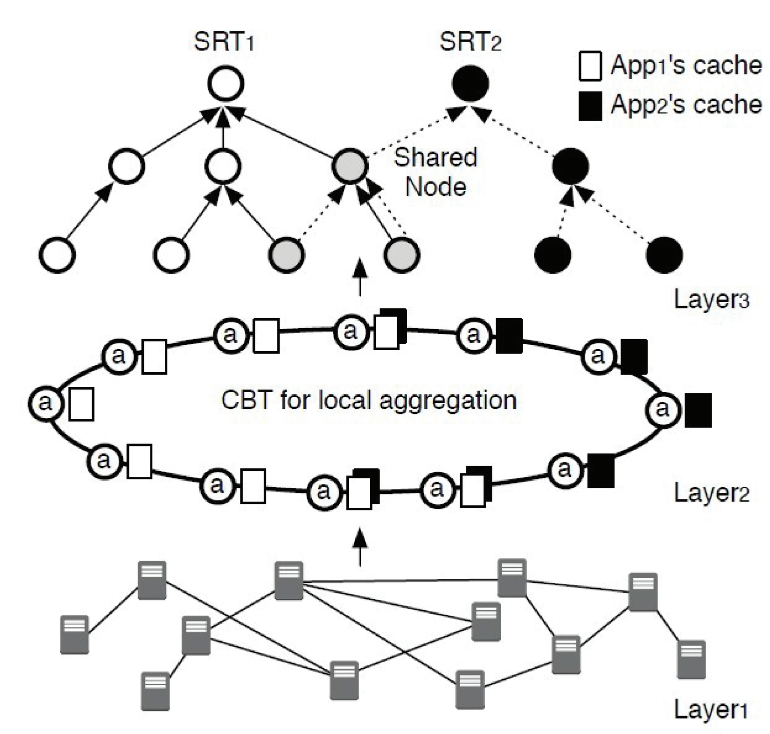
- Large amount of live event logs, click streams, or other various data feeds.
- MapReduce is not for stream applications.
- Solutions need to be flexible and scalable.

OUR PROPOSAL - ELF

- Compressed Buffer Tree (CBT) like "Map".
- Shared Reducer Tree (SRT) like "Reduce".

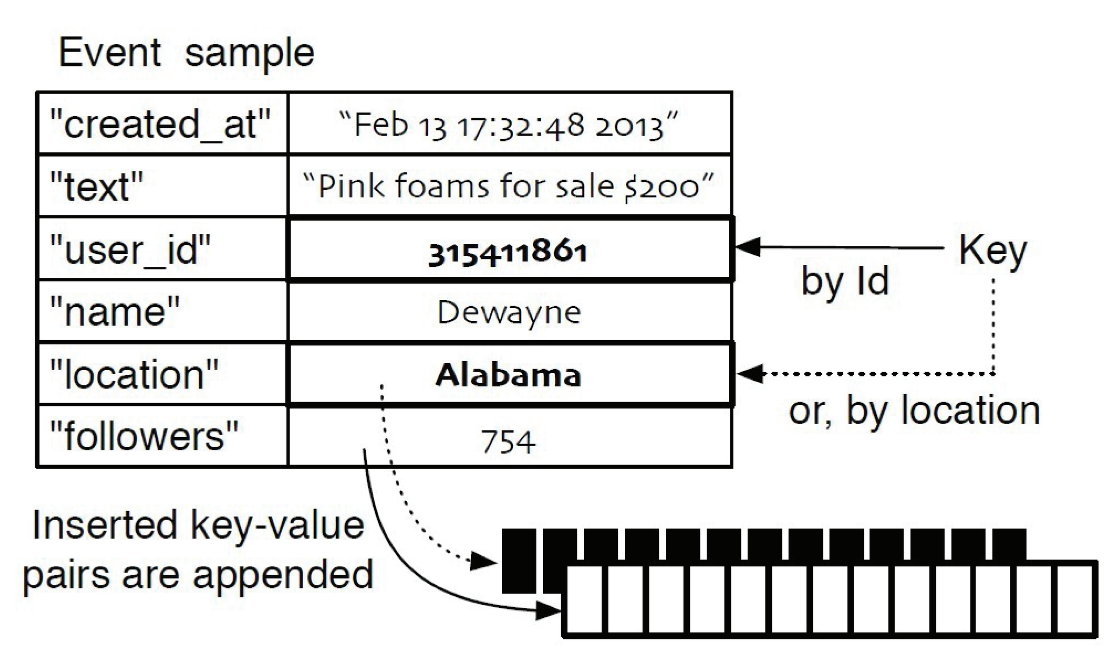
Exploit P2P overlay for scalability and functionalities

LAYERED STRUCTURE

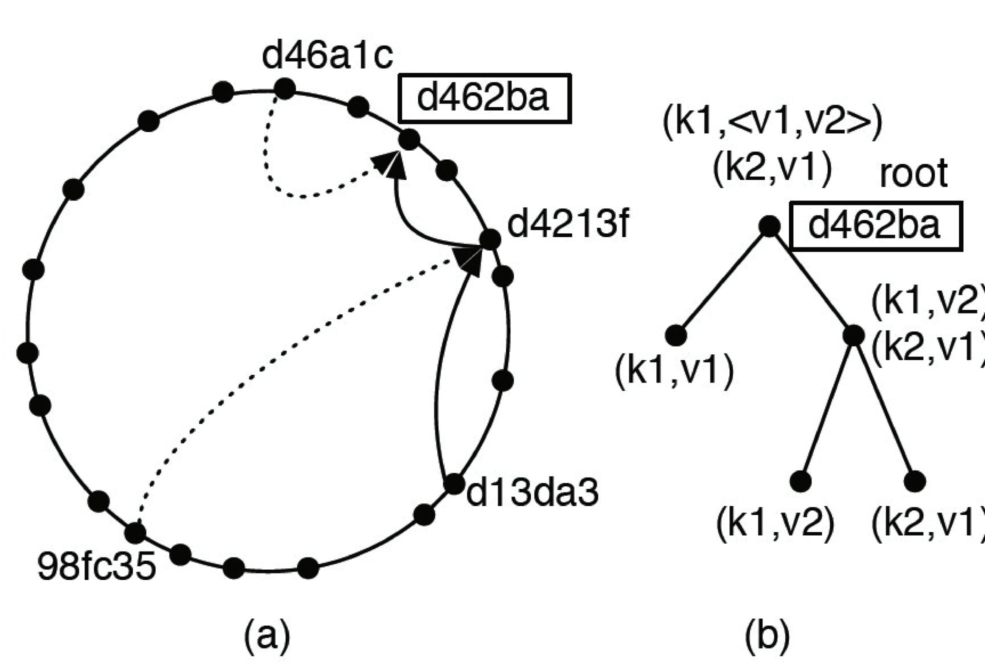


Hiding all messy details of parallelization, load-balancing and fault-tolerance

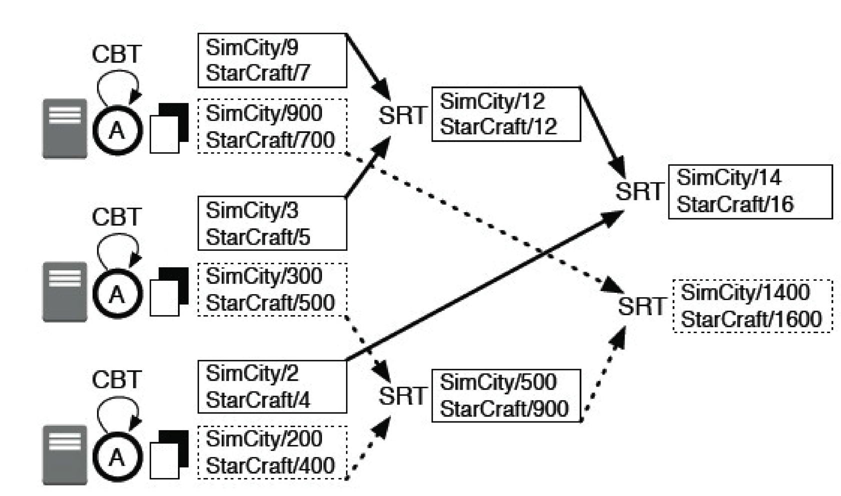
WORKFLOW OF A STREAM APP USING ELF



Parse events into key-value pairs and send to CBT for local aggregation



Distributed datasets (Caches) are progressively reduced by SRT



Example of micro-sale application

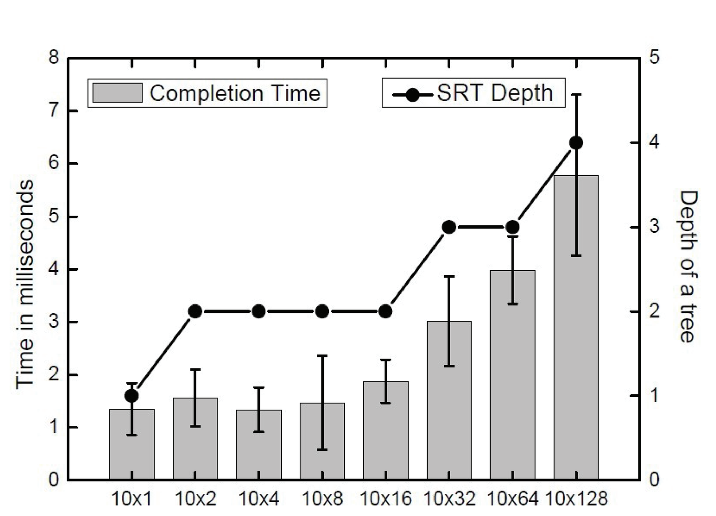
BETTER PERFORMANCE

- Little overheads – no storage nodes, memory efficiency.
- Low latency – 100 times less than MapReduce and its variations.
- High throughput – long historical records.

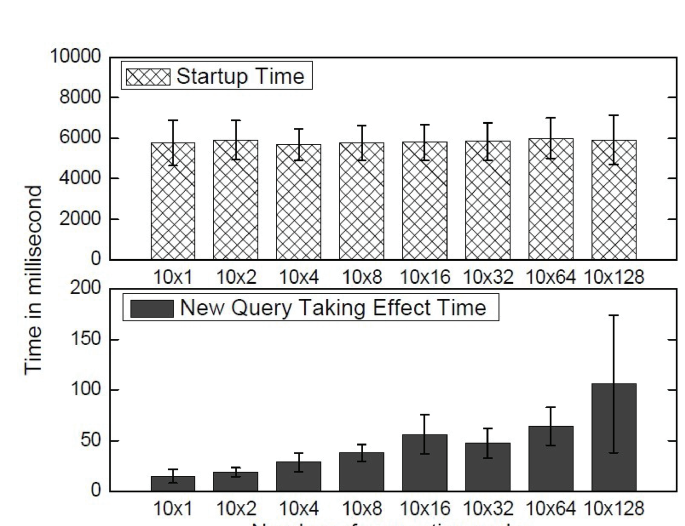
MORE FUNCTIONALITIES

- Support changing the query on the fly.
- Support adding or removing participating nodes.
- ELF is full decentralized without master node.

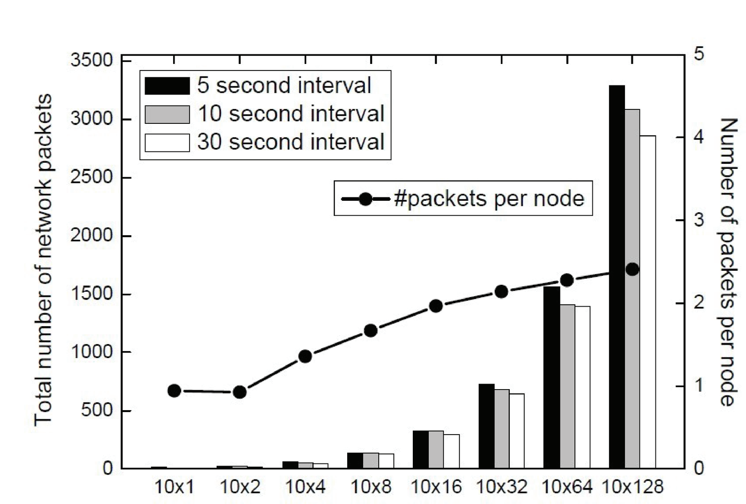
EVALUATION OF ELF



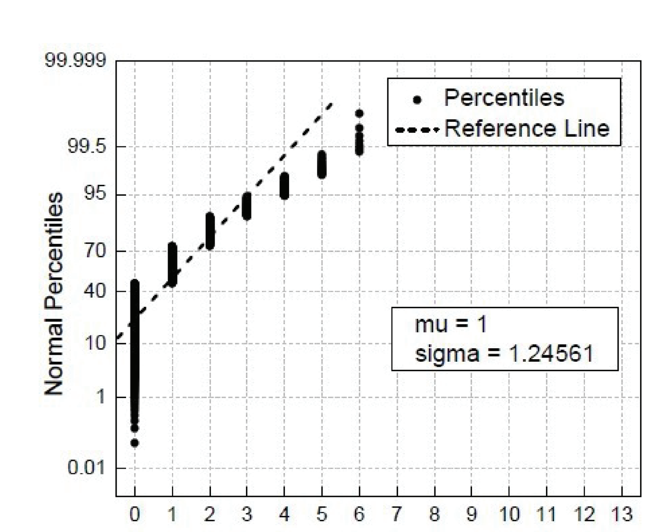
Latency is as low as 10 milliseconds for query completion time; Scales well with number of nodes.



Startup time is around 7 seconds; New query taking effect time is as low as 0.1 second.



The network bandwidth overhead for maintaining the overlay and SRT is low.



When deploying 1000 jobs onto 1000 nodes, the load is balanced without causing bottleneck.

ELF is scalable, flexible, and configuration-free!

Intel Science & Technology Center for Cloud Computing
cercs
Carnegie Mellon University
Georgia Tech
intel
PRINCETON UNIVERSITY
UC Berkeley
UNIVERSITY of WASHINGTON