Towards High Performance Processing of Streaming Data

May-27-2016

Supun Kamburugamuve, Saliya Ekanayake, Milinda Pathirage and Geoffrey C. Fox

Indiana University

Bloomington, USA

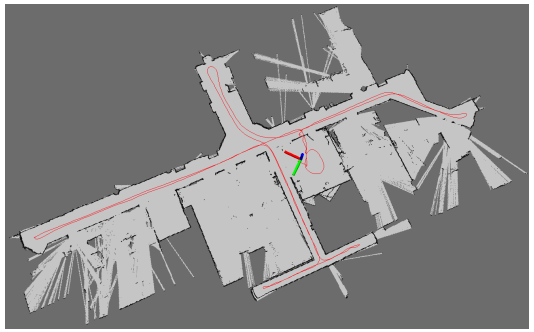
Outline

- Research Motivation
- Distributed stream processing systems (DSPFs)
- Communication improvements
- Results
- Future work

Background & Motivation

Robotics Applications

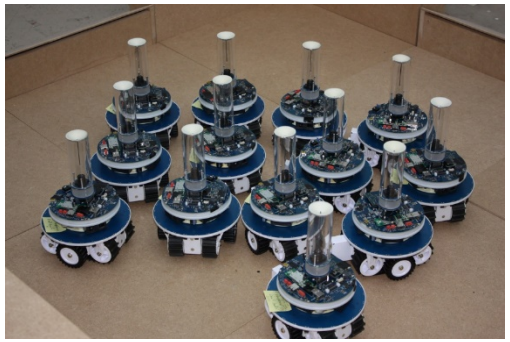
Simultaneous Localization and Mapping



Map Built from Robot data

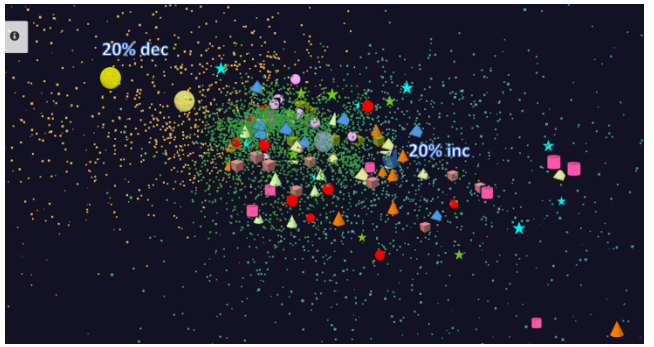
Robot with a Laser Range Finder

N-Body Collision Avoidance



Robots need to avoid collisions when they move

Time series data visualization in real time

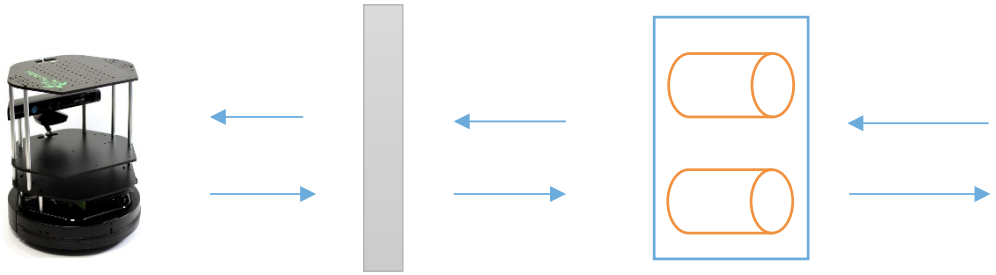


Work in progress

Cloud Applications

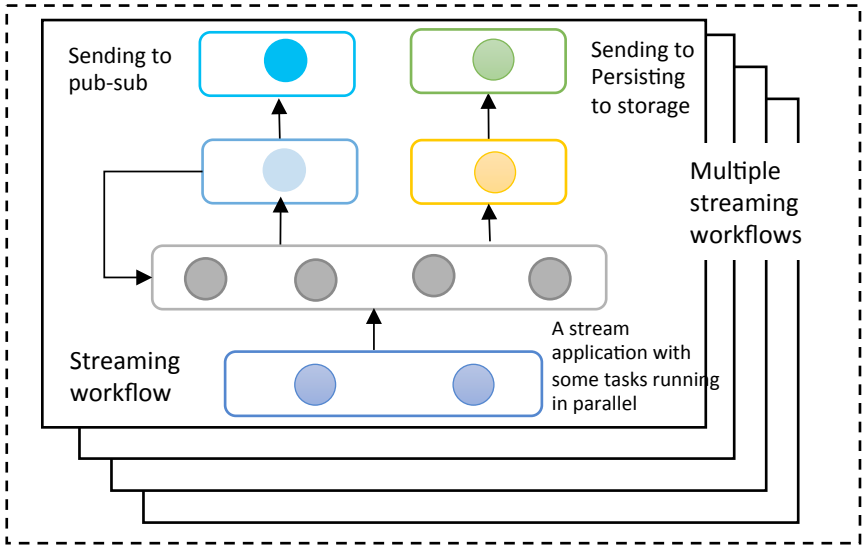
HPC Applications

Data pipeline



Gateway

Message Brokers
RabbitMQ, Kafka



Streaming Workflows
Apache Storm

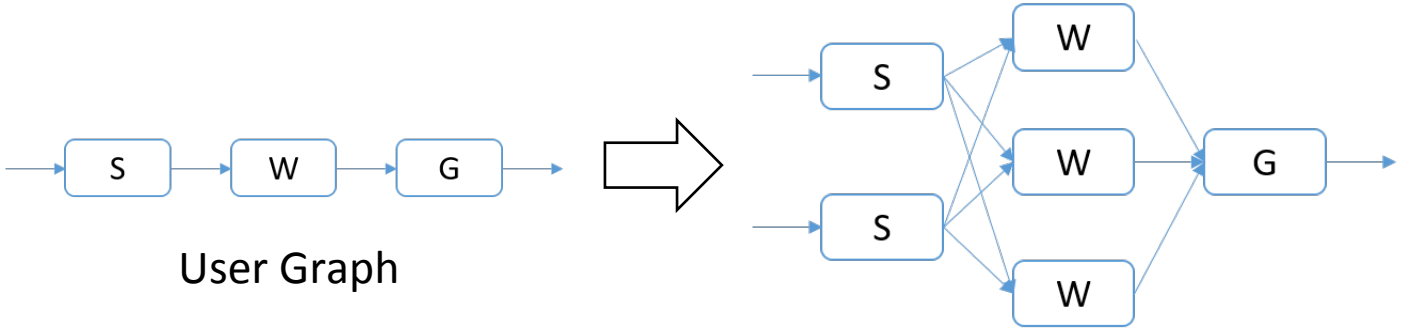
End to end delays without any processing is less than 10ms

Hosted in FutureSystems OpenStack cloud which is accessible through IU network

Improving communications

- Broadcast communication
- Shared memory communication between workers in a node

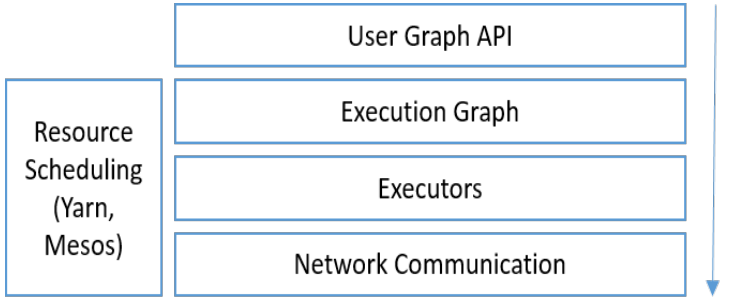
Streaming Application



Stream of events

Event processing logic

User graph is converted to an execution graph

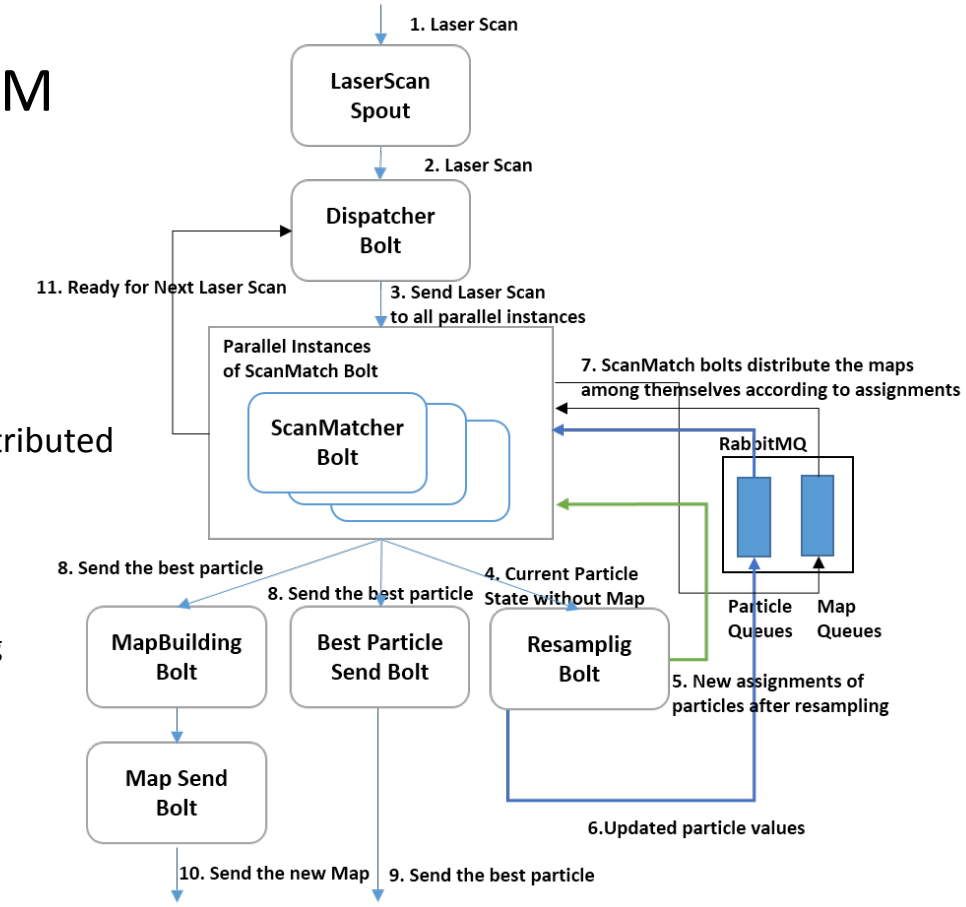


Communication methods

Broadcast, Round Robin, Direct, Stream Partitioning

Example applications

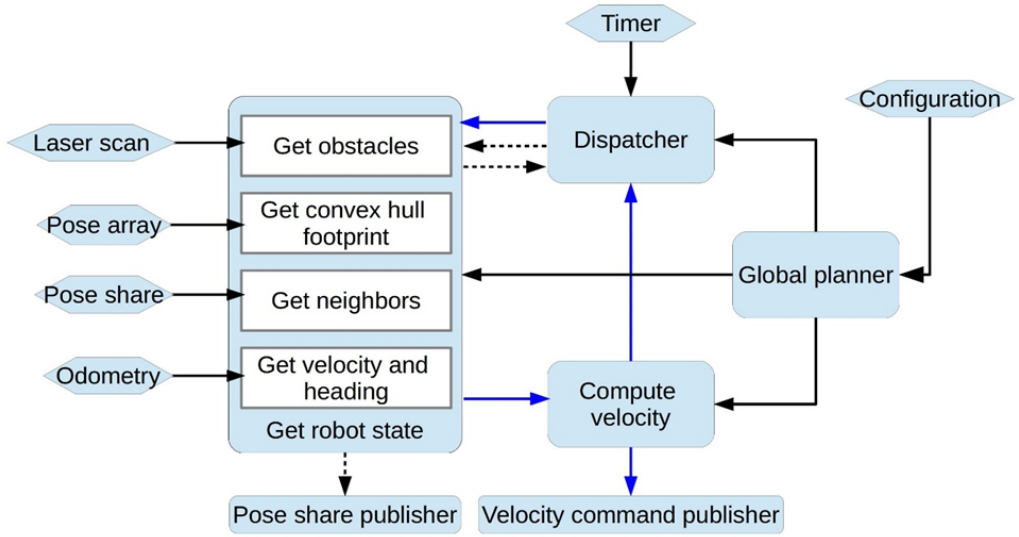
SLAM



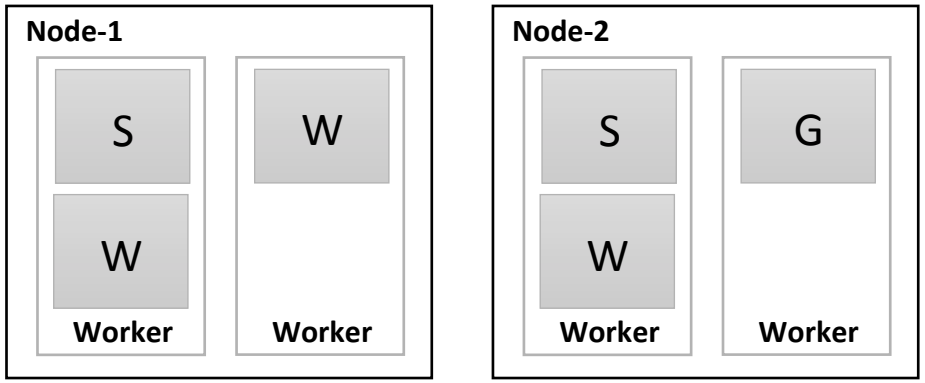
Particles are distributed in parallel tasks

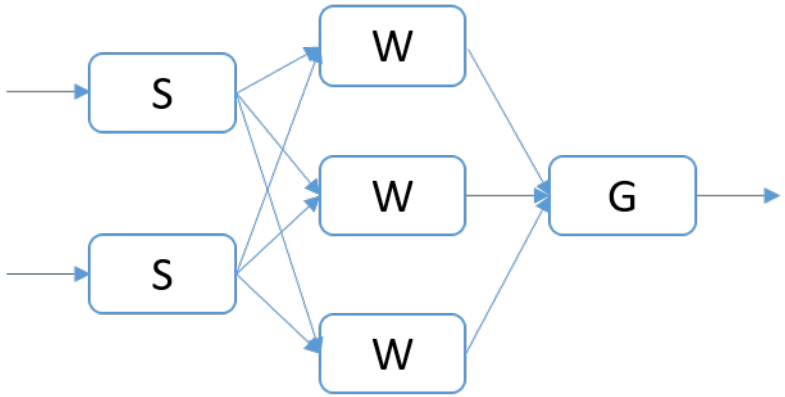
Map building happens periodically

N-Body collision avoidance



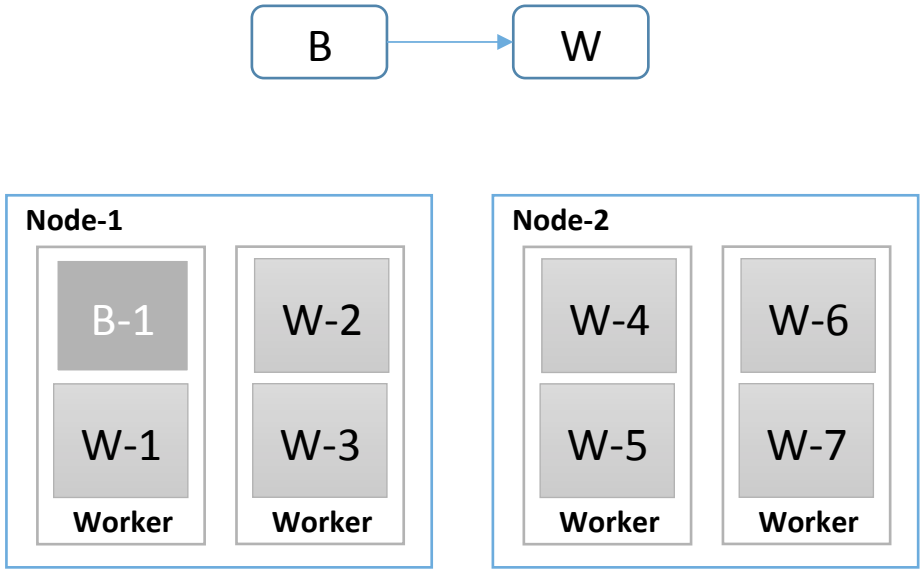
Execution Graph Distribution in the Storm Cluster



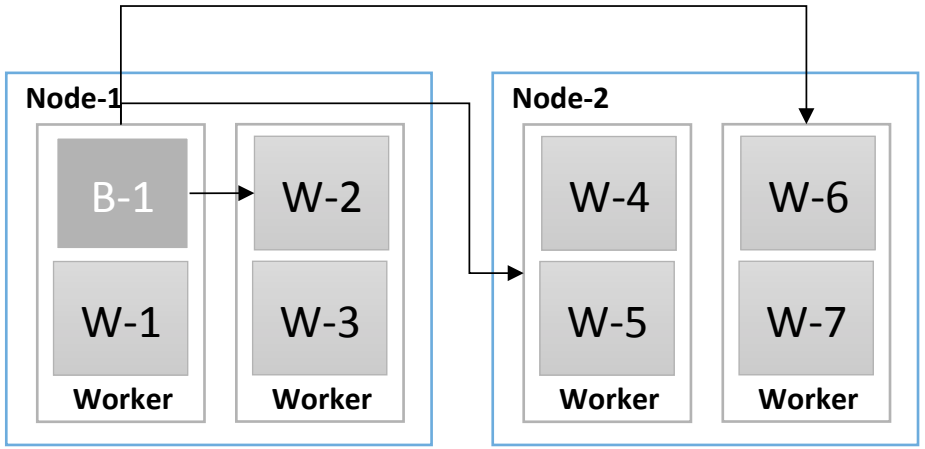


Two node cluster each running two workers. The tasks of the Topology is assigned to the workers

Communications in Storm

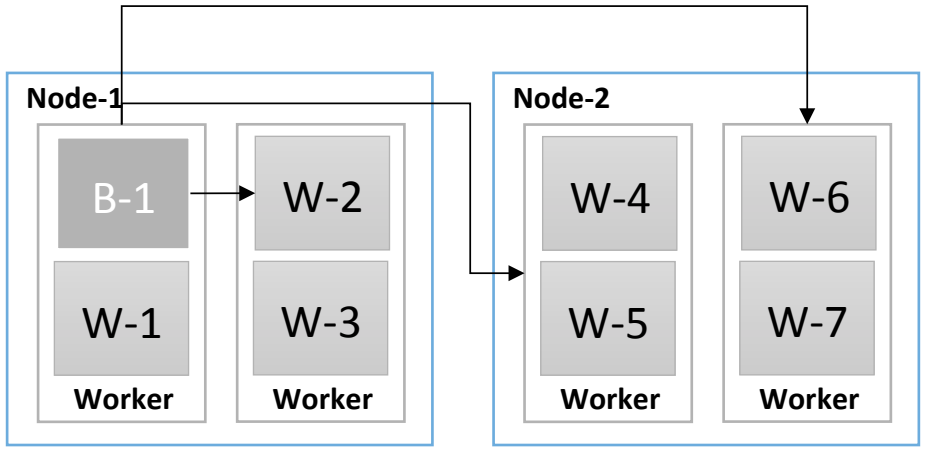


Worker and Task distribution of Storm
A worker hosts multiple tasks. B-1 is a task of component B and W-1 is a task of W



Communication links are between workers
These are multiplexed among the tasks

Default Broadcast

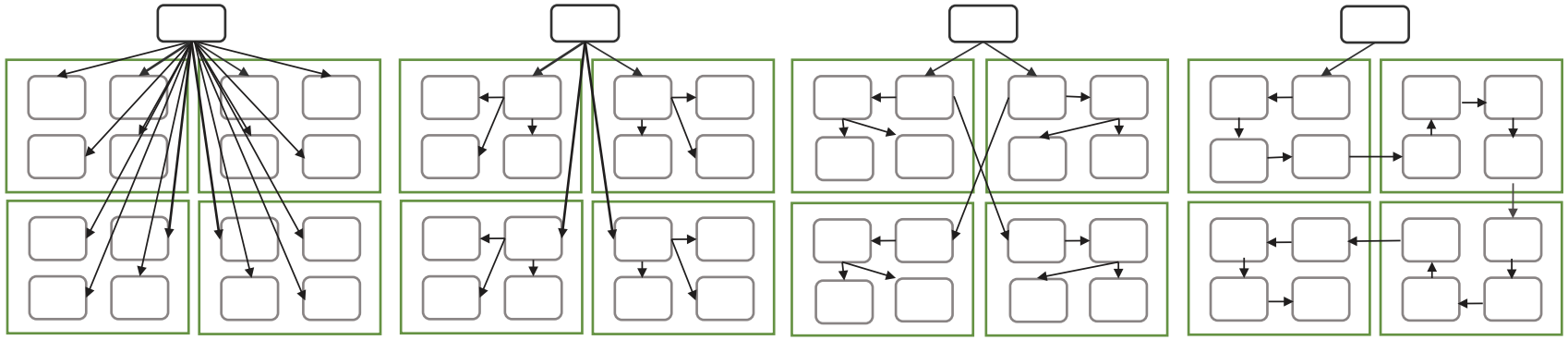


B-1 wants to broadcast a message to W, it sends 6 messages through 3 TCP communication channels and send 1 message to W-1 via memory

Optimized Broadcast

- Binary tree
  - Workers arranged in a binary tree
- Flat tree
  - Broadcast from the origin to 1 worker in each node sequentially. This worker broadcast to other workers in the node sequentially
- Bidirectional Rings
  - Workers arranged in a line
  - Starts two broadcasts from the origin and these traverse half of the line

Three Algorithms for broadcast

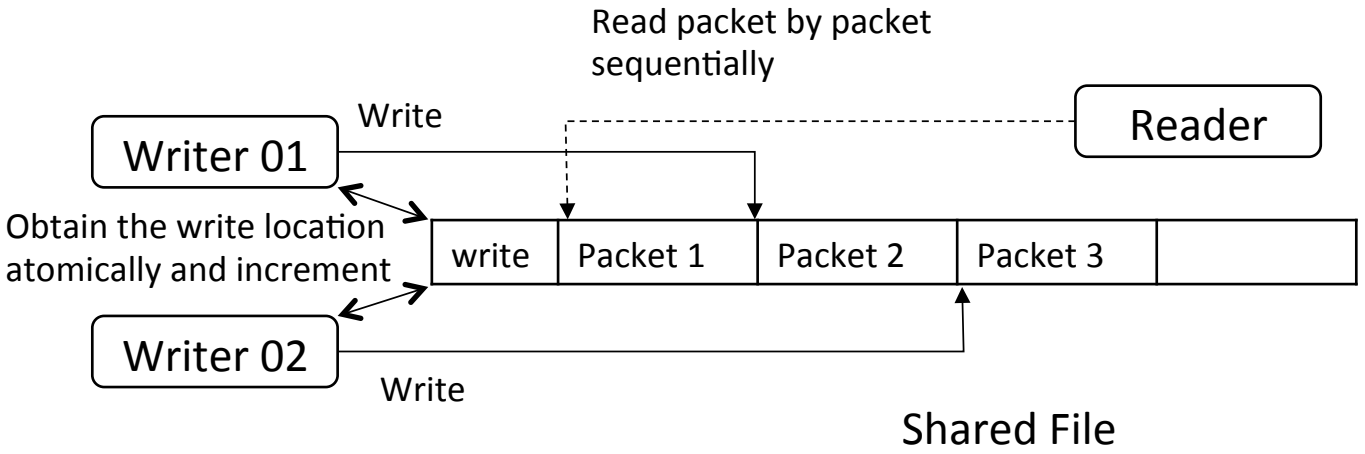


Example broadcasting communications for each algorithm in a 4 node cluster with each machine having 4 workers. The outer green boxes show cluster machines and inner small boxes show workers. The top box displays the broadcasting worker and arrows illustrate the communication among the workers

Shared memory based Communications

- Inter process communications using shared memory for a single node
- Multiple writer single reader design
- Writer breaks the message in to packets and puts them to memory
- Reader reads the packets and assemble the message

Shared Memory Communications



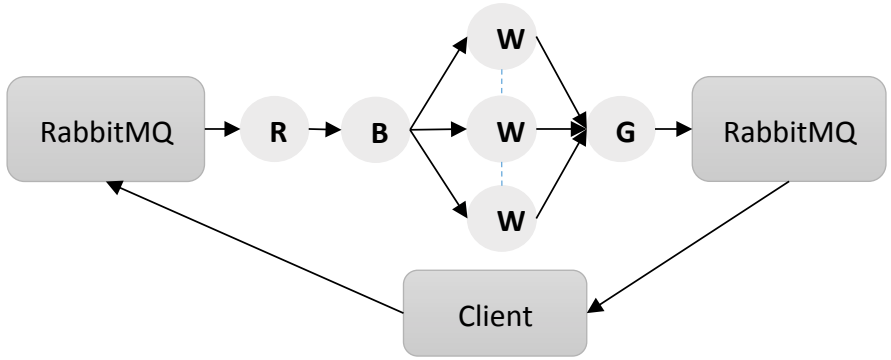
Use a new file when the file size is reached
Reader deletes the files after it reads them fully

Fields	ID	No of Packets	Packet No	Dest Task	Content Length	Source Task	Stream Length	Stream	Content
Bytes	16	4	4	4	4	4	4	Variable	Variable

Packet Structure

Experiments

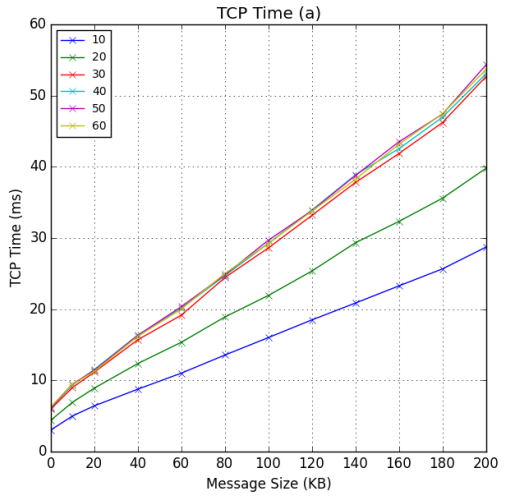
- 11 Node cluster
- 1 Node – Nimbus & ZooKeeper
- 1 Node – RabbitMQ
- 1 Node – Client
- 8 Nodes – Supervisors with 4 workers each
- Client sends messages with the current timestamp, the topology returns a response with the same time stamp. Latency = current time - timestamp



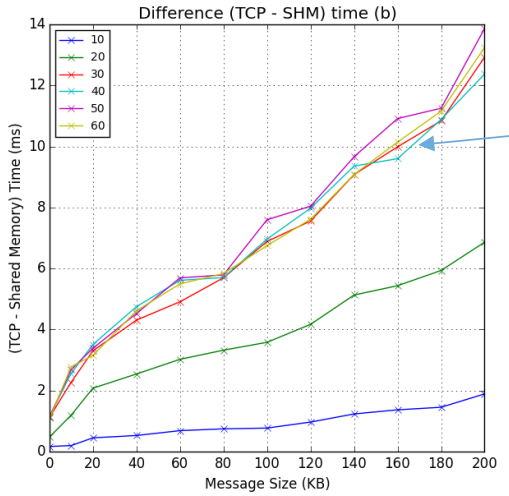
Memory Mapped Communications

Relative Importance of Shared Memory Communication to TCP

A topology with communications going through all the workers arranged in a line



Default TCP implementation Latency

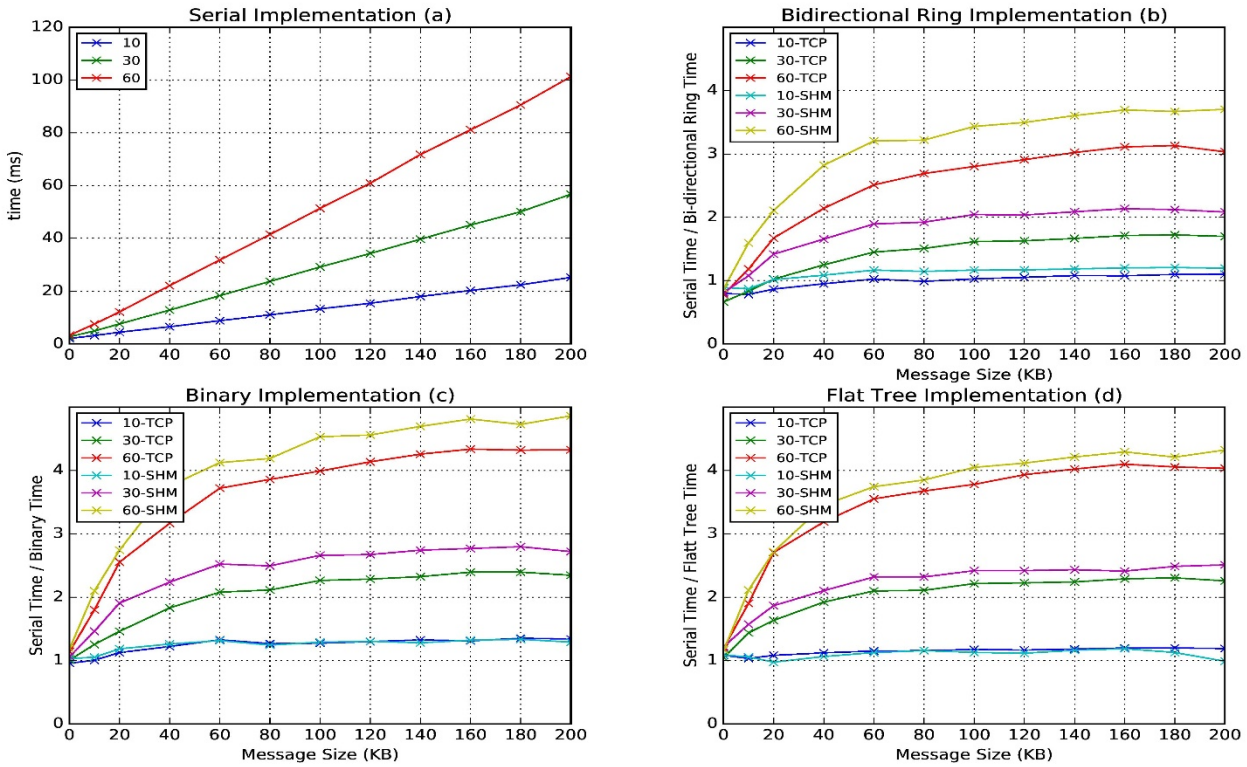


Y axis shows the difference in latency of default TCP implementation and shared memory based implementation (TCP - SHM)

No significant difference because we are using all the workers in the cluster beyond 30 workers

About 25% reduction for 32 workers

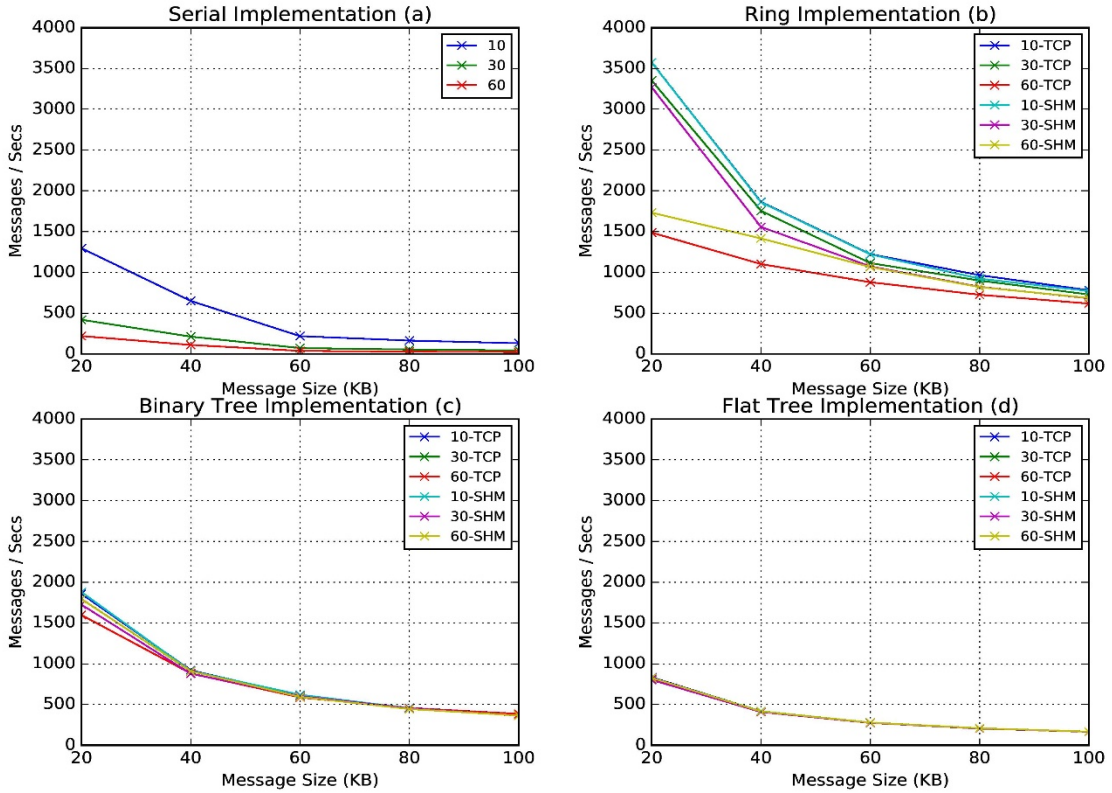
Broadcast Latency Improvement



Speedup of latency with both TCP based and Shared Memory based communications for different algorithms

Latency of binary tree, flat tree and bi-directional ring implementations compared to serial implementation. Different lines show varying parallel tasks with TCP communications and shared memory communications(SHM).

Throughput



Throughput of serial, binary tree, flat tree and ring implementations. Different lines show varying parallel tasks with TCP communications and shared memory communications (SHM)

Future Work

- Implement the Shared memory and communication algorithms for other stream engines (Twitter Heron)
- Experiment on larger clusters with applications
- HPC Scheduler for Streaming applications (Slurm, Torque)
- C++ APIs for data processing
- High performance interconnects for high throughput low latency communications in HPC clusters
- Scheduling to take advantage of the shared memory & collective communications

Summary

- Communication algorithms reduce the network traffic and increases throughput
- Ring gives the best throughput and binary tree best latency
- Shared memory communications reduce the latency further but not throughput because TCP is the bottleneck

This work was partially supported by NSF CIF21 DIBBS 1443054 and AFOSR FA9550-13-1-0225 awards. We would also like to express our gratitude to the staff and system administrators at the Digital Science Center (DSC) at Indiana University for supporting us on this work.

References

- Projects https://github.com/iotcloud
- SLAM http://dl.acm.org/citation.cfm?id=2896433
- Collision Avoidance http://goo.gl/xdB8LZ
- Time series data visualization http://dsc.soic.indiana.edu/publications/tsmap3d.pdf
- Apache Storm http://storm.apache.org/
- Twitter Heron https://github.com/twitter/heron