Pre-production and Debugging Tools for Timely dataflow

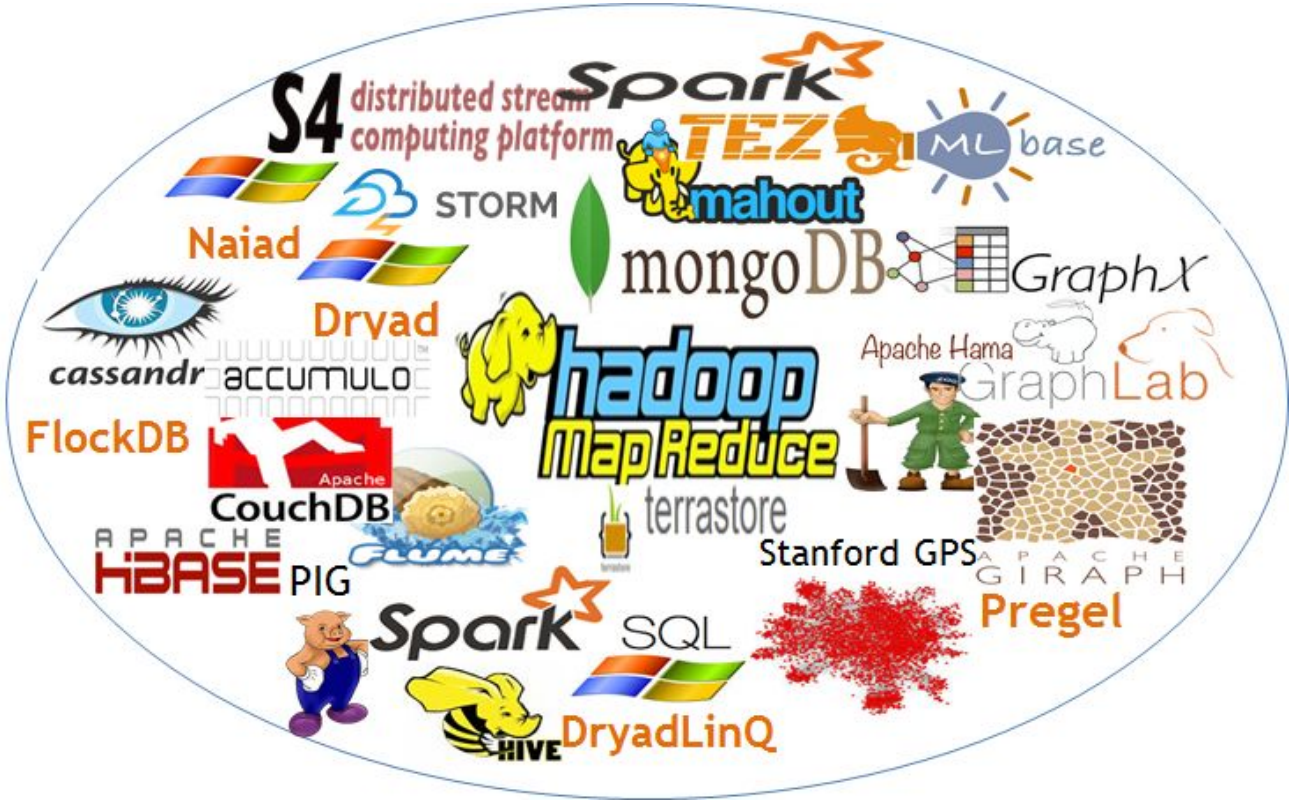
CS 848: Models and Applications of Distributed Data Systems

Mon, Dec 5th 2016

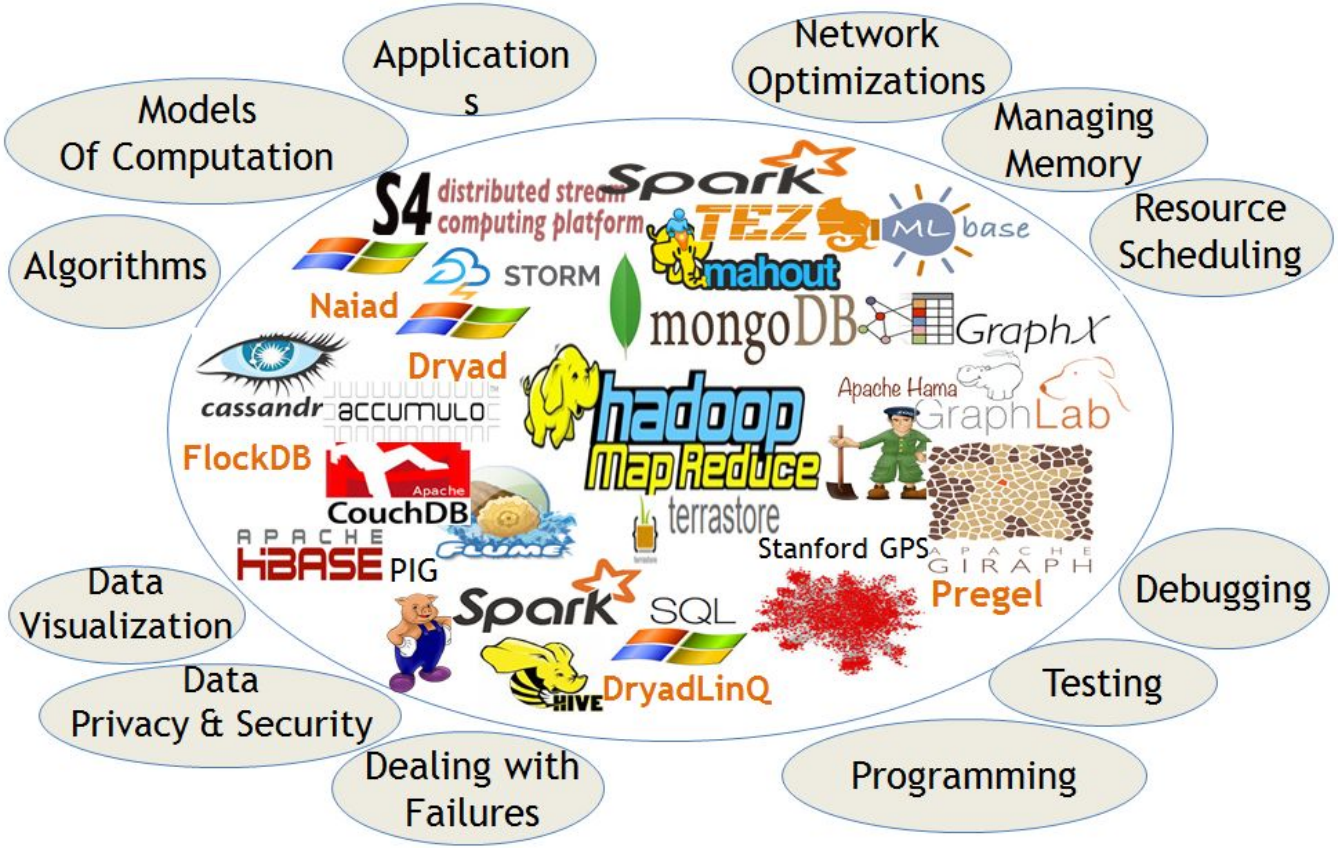
Amine Mhedhbi & Saifuddin Hitawala

Distributed Data Processing Systems in 2006

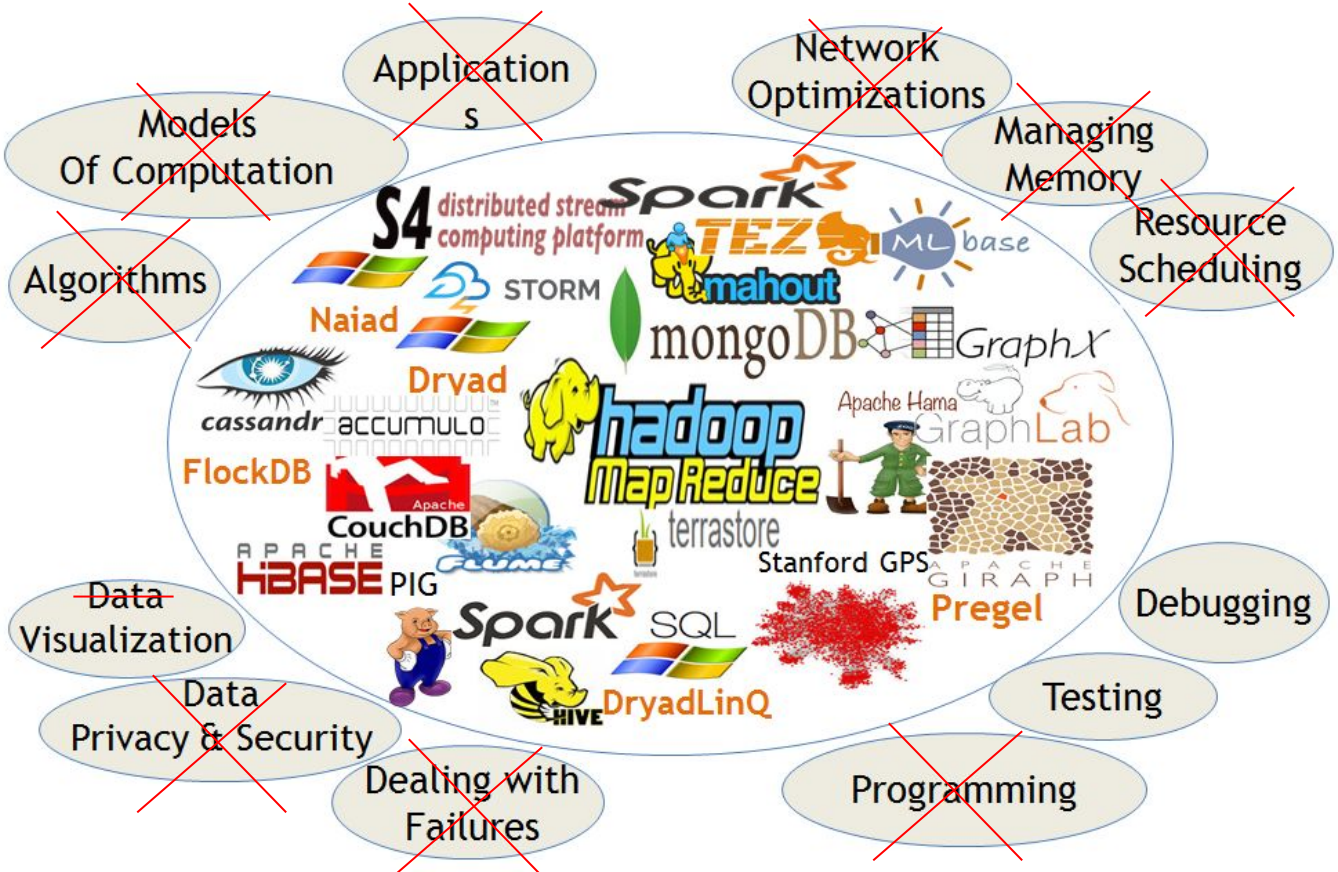
Distributed Data Processing Systems in 2016



Many topics of Interest Within These Systems



We Picked ....


Applications
Network Optimizations
Models Of Computation
Managing Memory
Resource Scheduling
Algorithms
S4
distributed stream computing platform
Spark
TEZ
ML base
mahout
STORM
Naiad
mongoDB
GraphX
Dryad
cassandra
accumulo
Apache Hama
GraphLab
hadoop
MapReduce
FlockDB
Apache
CouchDB
terrastore
APACHE
HBASE
PIG
Stanford GPS
APACHE
GIRAPH
Pregel
Data Visualization
Spark
SQL
Debugging
Data Privacy & Security
HIVE
DryadLinQ
Testing
Dealing with Failures
Programming

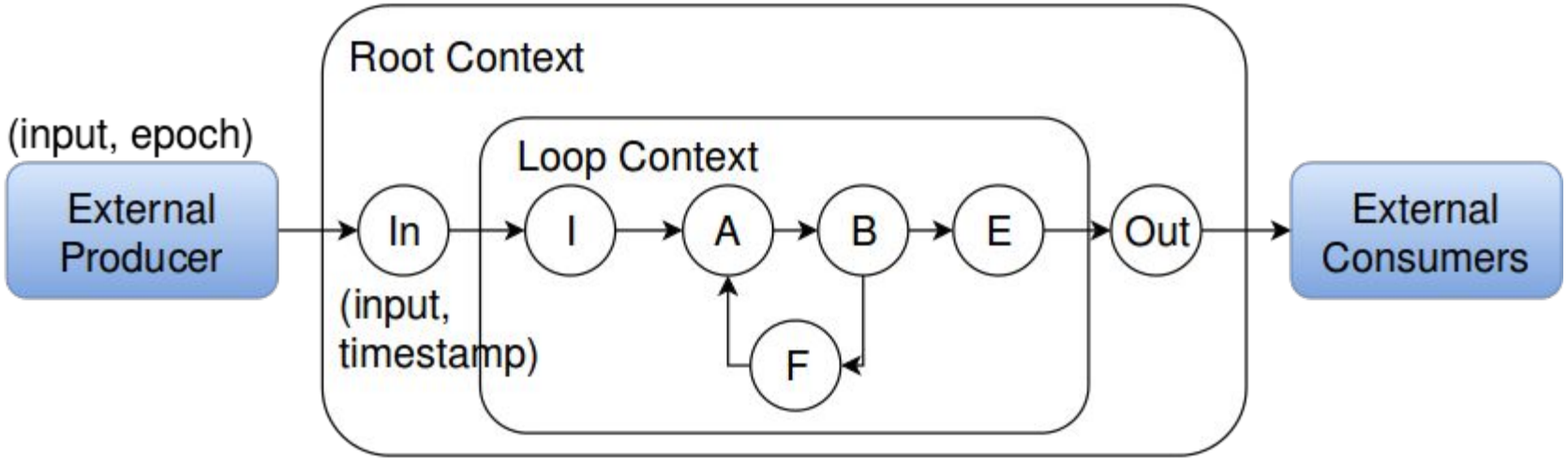
Project Statement

- **“Timely Dataflow”** is a rewrite of Naiad System in Rust under the MIT License. * **Prototype** *

- Goal: DEVELOPER-TOOLS

Flash Back of the Past

Background


Root Context
(input, epoch)
External Producer
Loop Context
In
I
A
B
E
Out
External Consumers
(input, timestamp)
F

Background

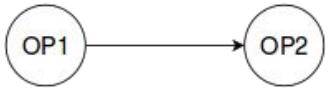

```
"OperatesEvent":          // Type of the logged obj
{
    "id": int,            // unique id.
    "addr": [int, int, int],   // address in terms of scope & id.
    "name": String,       // operators name in timely dataflow
}
```

Background


```
"OperatesEvent":
{
    …
    "name": “OP1”
}
```

```
"OperatesEvent":
{
    …
    "name": “OP2”
}
```

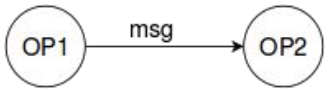
Background




```
"ChannelsEvent":
{
        "id": int,                      // unique id

        "scope_addr": [int, int], // scope & worker id
        "source": [int, int],          // [op_id, scope_id]
        "target": [int, int],          // [op_id, scope_id]
}
```

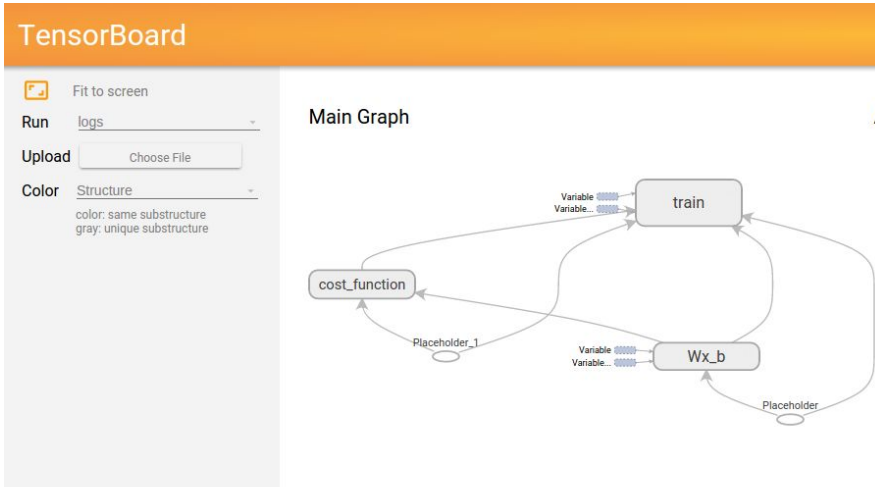
Background

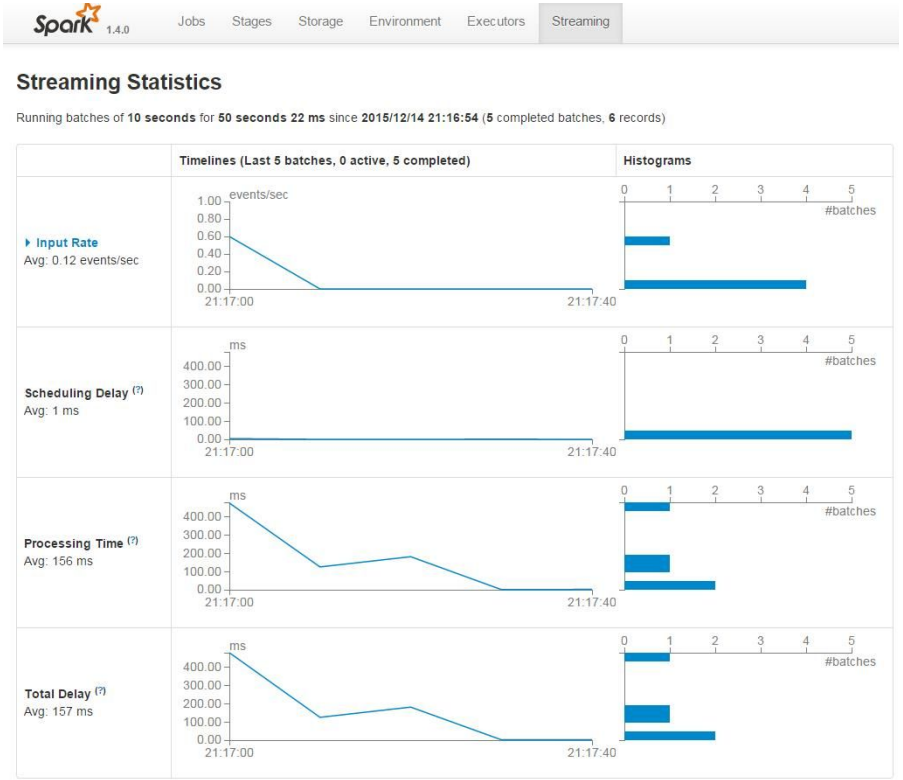



```
"MessageEvent":
{
    "is_send": bool,        // push or pull
    "channel": int,         // unique id
    "source": int,          // worker id
    "target": int,          // worker id
    "length": int,          // number of typed records
}
```

Related Work

Related Work : Tensorflow Dashboard & Apache Stats

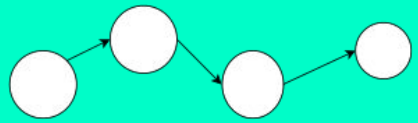




Features

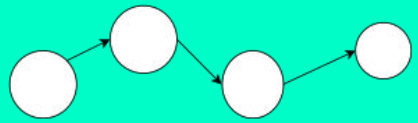
Features

- Visualize The Computation Topology



Features

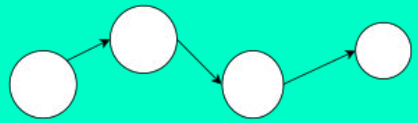
- Visualize The Computation Topology



- Report skew between workers

Features

- Visualize The Computation Topology

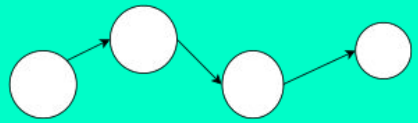


- Report skew between workers

- Replay computation step-by-step visually

Features

- Visualize The Computation Topology



- Report skew between workers

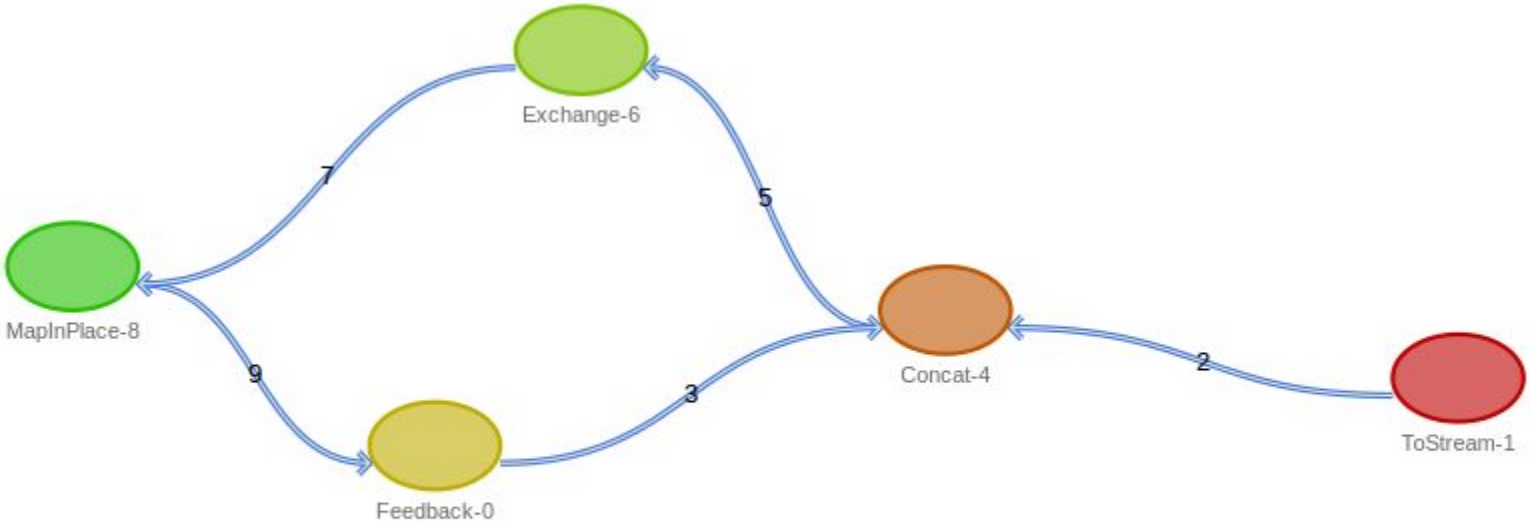
- Replay computation step-by-step visually

- Real-Time Machine Monitoring

DEMO TIME(LY)!

Experiments & Evaluation

PingPong: Topology

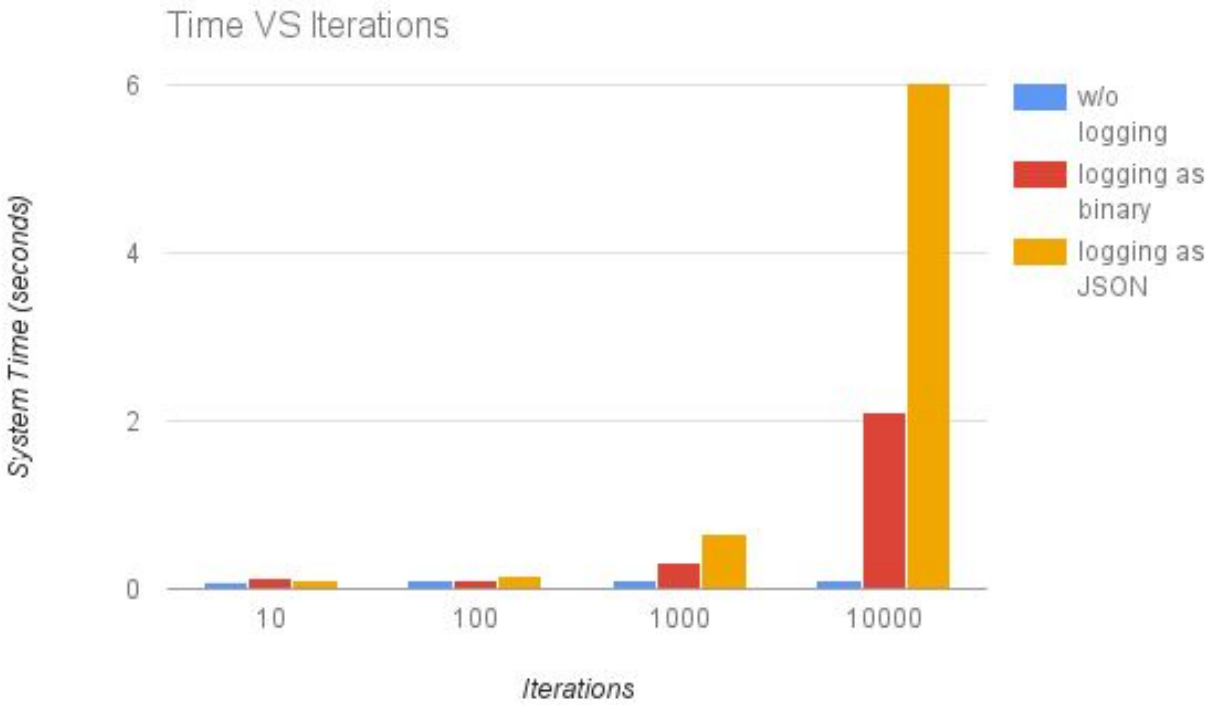


redraw

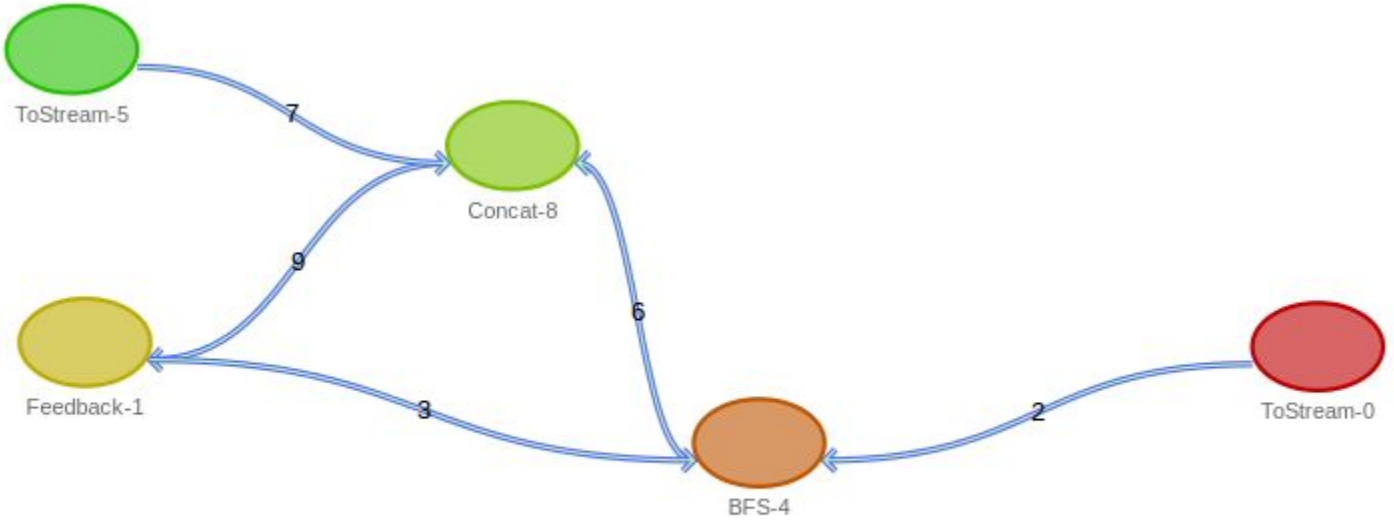
Pingpong: Experimental Runs, num of iterations = 10000

Used Himrod Cluster with machines having 256GB memory

Pingpong: Experimental Runs, num of iterations = [10, 100, 1000, 10000]



BFS: Topology



BFS: Experimental Runs

Web App Back-end Profiling

In Progress:

- Profile server-client response time for the 4 features.

Conclusion

Conclusions

- JSON -> Binary for logging.

Conclusions

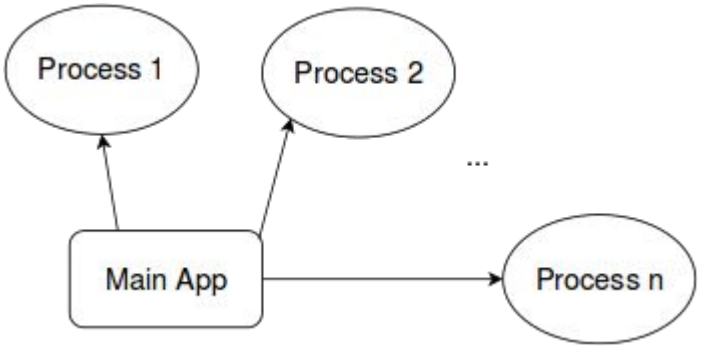
- JSON -> Binary for logging.
- Large scale testing is a must.

Conclusions

- Project is a prototype. A lot of needed improvements:

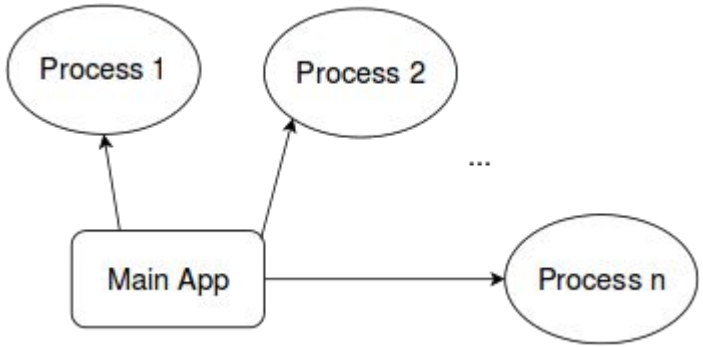
Conclusions

- Project is a prototype. A lot of needed improvements:



Conclusions

- Project is a prototype. A lot of needed improvements:



kafka

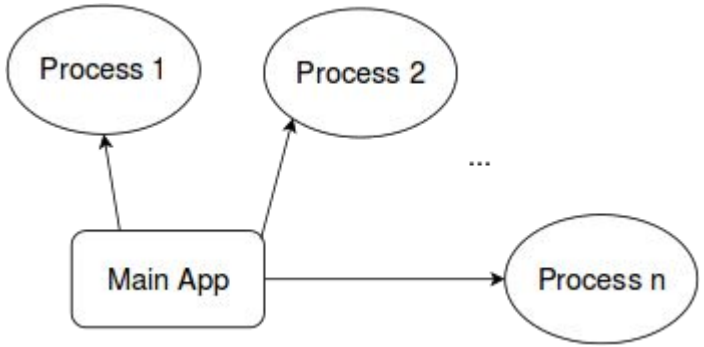
ØMQ

RabbitMQ

Open Source Enterprise Messaging

Conclusions

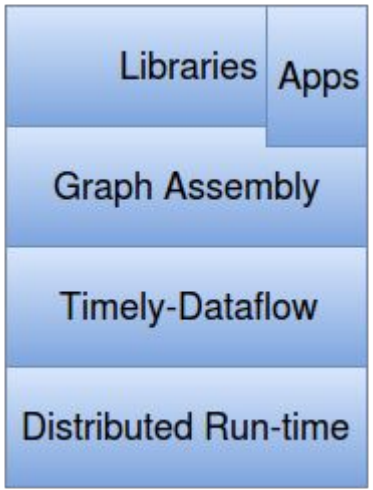
- Project is a prototype. A lot of needed improvements:



kafka

ØMQ

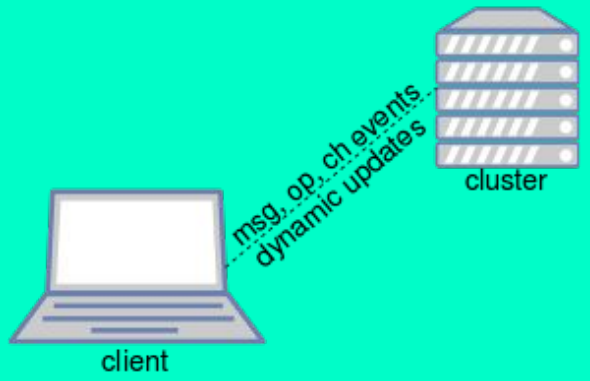
RabbitMQ
Open Source Enterprise Messaging



Future Work

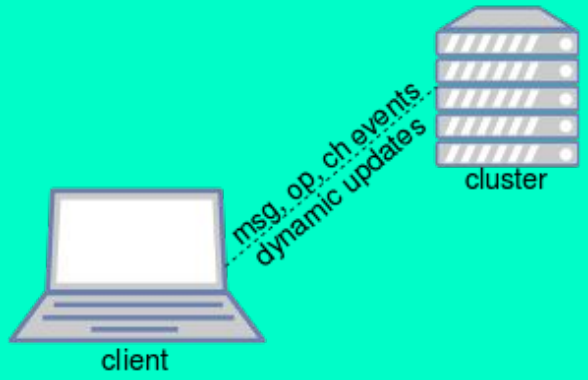
Future Work

- Real-Time Computation Monitoring

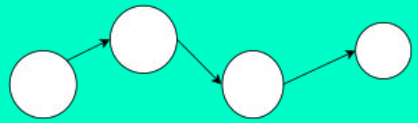


Future Work

- Real-Time Computation Monitoring

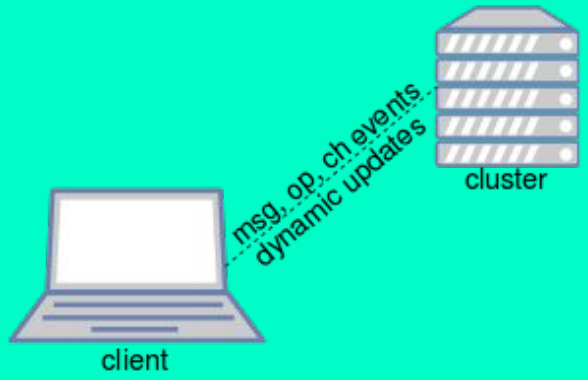


- UI code generation (drag & drop) for small computation

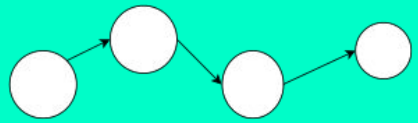


Future Work

- Real-Time Computation Monitoring



- UI code generation (drag & drop) for small computation



- Step-by-step debugging of multiple workers computations?!

Resources

- Timely Dataflow (Rust Implementation)

- Frank blog posts:
  - Timely dataflow
  - Differential dataflow

- Naiad Paper
- For slides [2-5]: Class slides by Prof. Semih Salihoglu

Fin.

Thank you! Q&A?!