IEEE/ACM DS-RT 2016
September 2016

Parallel Discrete-Event Simulation on Data Processing Engines

Kazuyuki Shudo, Yuya Kato,
Takahiro Sugino, Masatoshi Hanai
Tokyo Institute of Technology

首藤 一幸, 加藤 裕也
杉野 好宏, 華井 雅俊
東京工業大学

Tokyo Tech

Proposal: Parallel simulator on data processing engine

- Development of a decent parallel simulator is challenging work.
  - with BSD socket API, message passing or shared memory
  - 47.46 sec with PeerSim, but 1 hour 6 min with dPeerSim. 80x ~ slower.

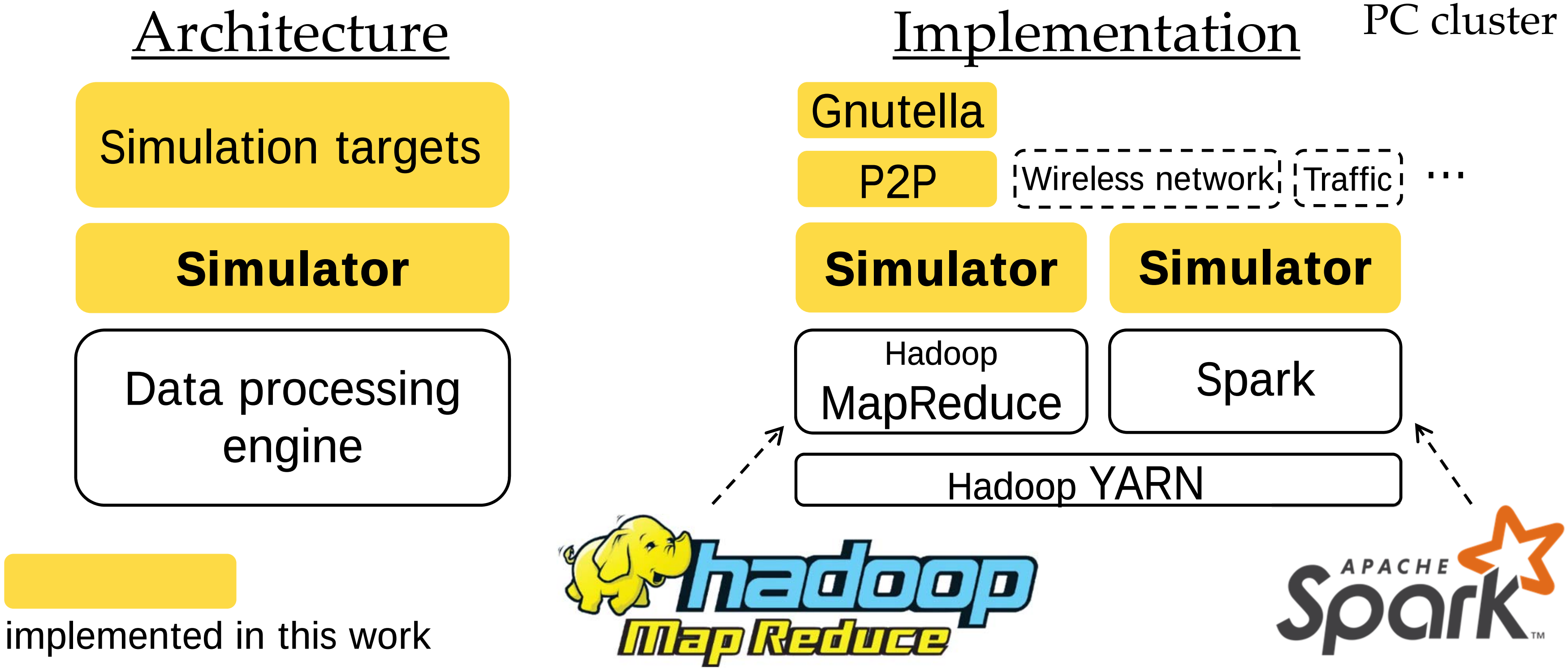
Overlay Weaver
2005 ~

- Data processing engines help it much.
  - **Performance** Moderate
    - » Comparable with a serial simulator
  - **Scalability** ~ Thousands of servers
    - » Hadoop runs on 4500 servers and Spark runs on 4000 cores
  - **Fault tolerance** Automatic reexecution

This work

- Parallel simulators on data processing engines are demonstrated.
  - Gnutella, a distributed system, is simulated on it.
  - It shows good scalability and a moderate performance.

PC cluster



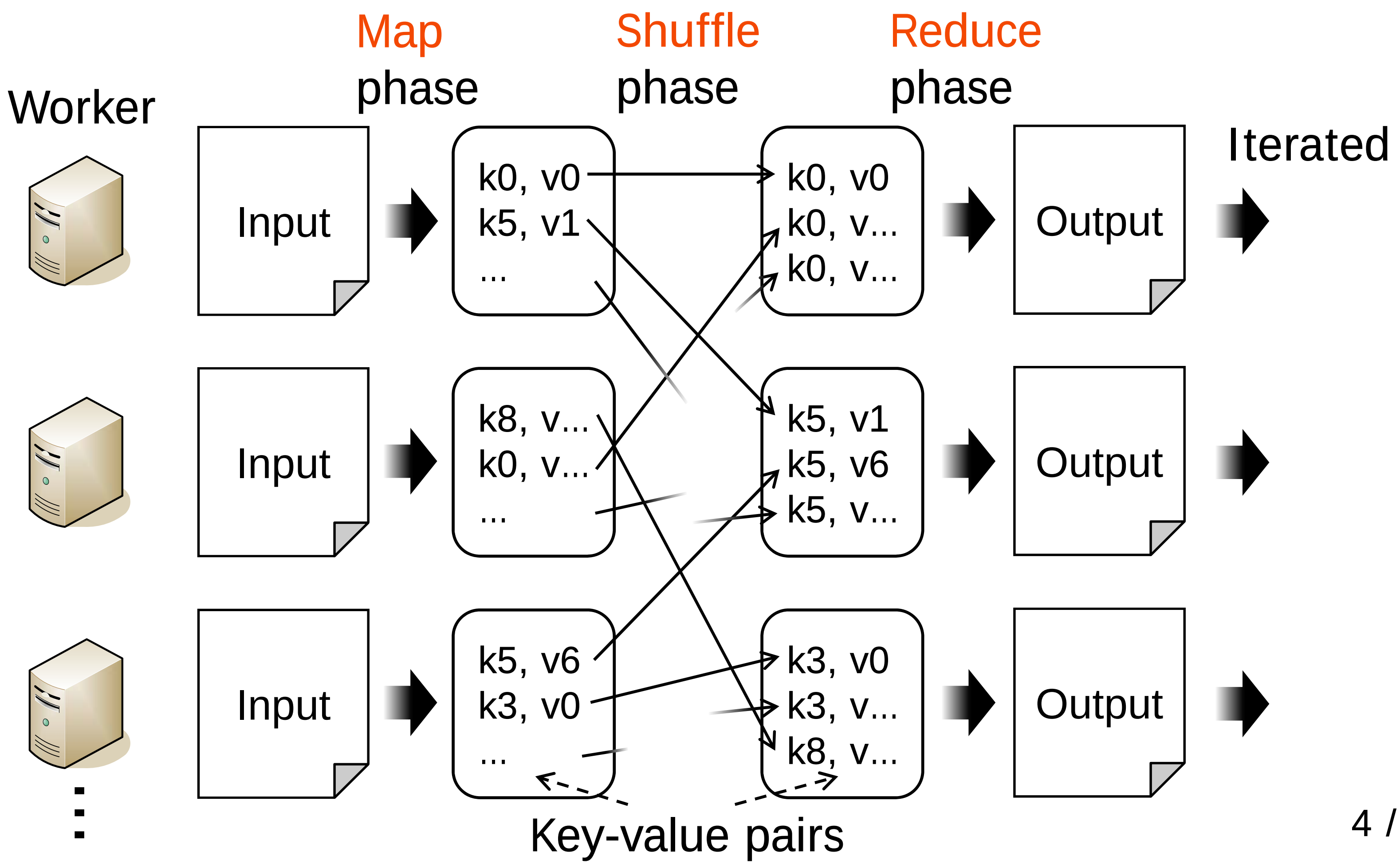
Contribution

We demonstrate that

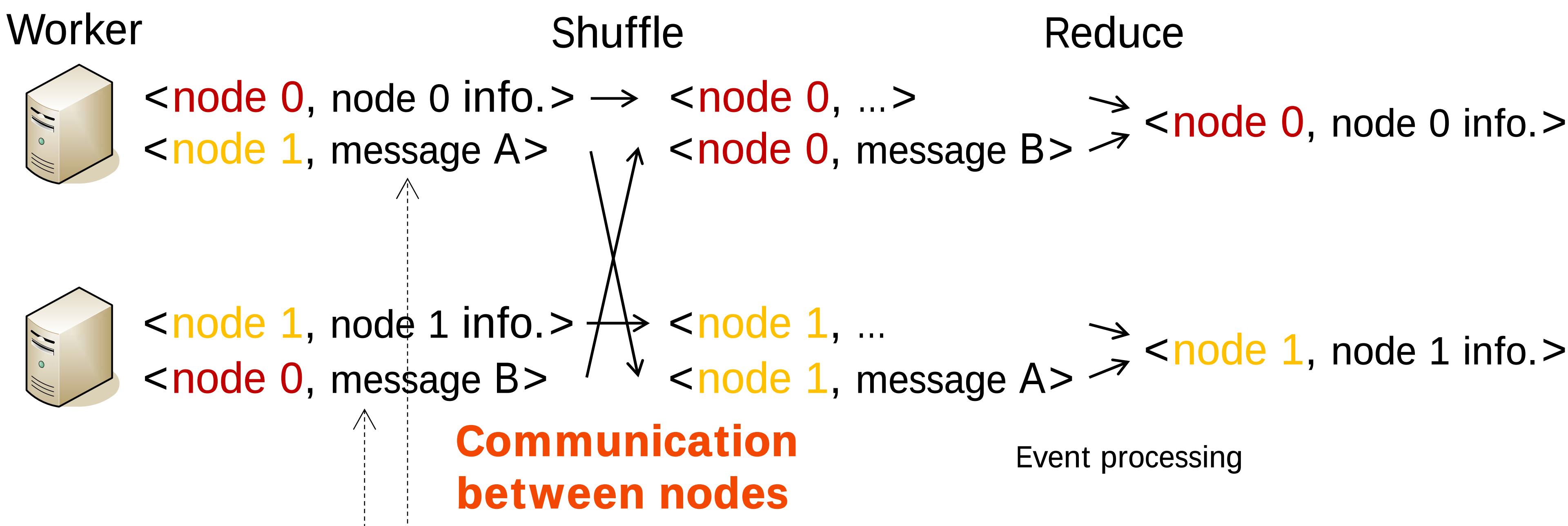
- Parallel Discrete-Event Simulation (PDES) works on data processing engines.
  - Cf. Existing work [20-23] adopted time-step-based synchronization with MapReduce processing model.
- Optimistic parallel simulation with Time Warp shows a moderate performance.
  - The performance is about **20x** of an existing parallel simulator. It is comparable with a serial simulator while enabling large-scale simulation.
- Distributed systems are modeled on MapReduce processing model.
  - **Peer-to-peer systems** (our target), wireless networks, ...

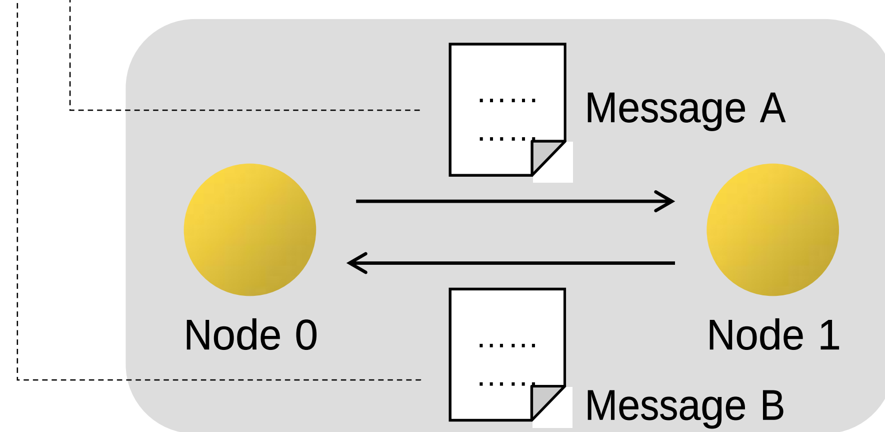
Background: MapReduce programming / processing model

- Most data processing engines support it.



Modeling of peer-to-peer systems on MapReduce





- Communication-intensive.
  - Different from, for example, traffic simulation

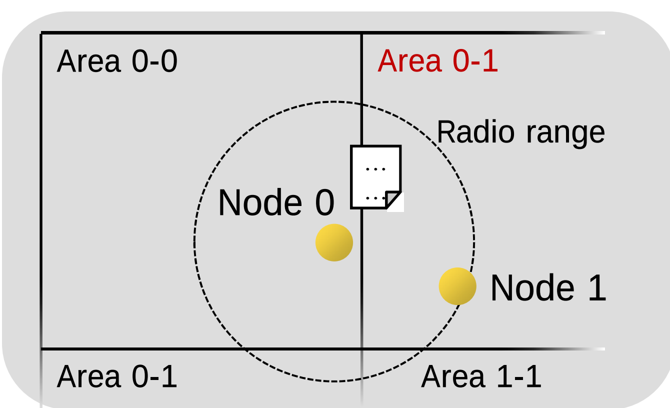
Simulated system

Modeling of wireless networks on MapReduce

Worker		Shuffle		Reduce
	<area 0-0, node 0 info.> →	<area 0-0, node 0 ...>	→	<area 0-0, node 0 ...>
	<area 0-0, message A> →	<area 0-0, message A>	→	
	<area 0-1, message A>			
	<area 1-0, message A>			
	<area 1-1, message A>			
	<area 0-1, node 1 info.> →	<area 0-1, node 1 ...>	→	<area 0-1, node 1 ...>
		<area 0-1, message A>	→	

Communication between nodes

Event processing



Simulated system

- Note: Designed but **not** implemented

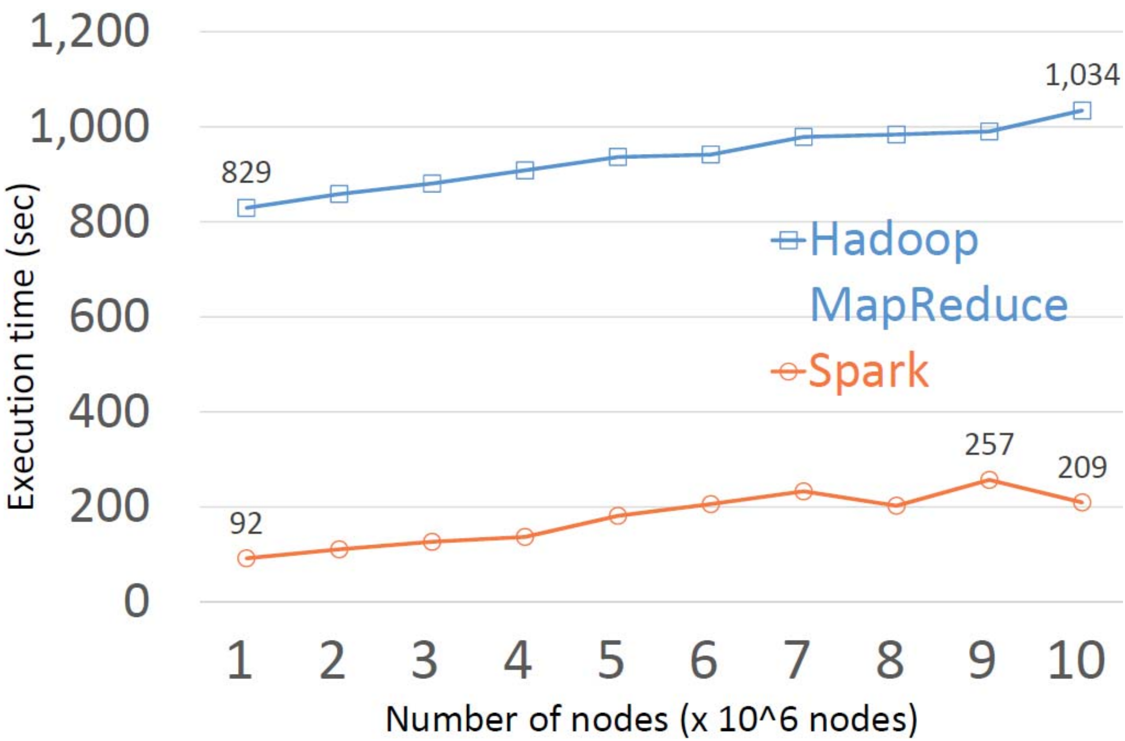
Details about design and impl.

- Models provide API to simulation targets.
  - Gnutella uses peer-to-peer (message passing) API.
- Simulation scenarios and simulated environment are also supplied.
  - From Hadoop Distributed File System
  - E.g. Network topology, bandwidth, latency and jitter
- Non-optimistic and optimistic synchronization protocols are implemented.
  - Null Message algorithm [Chandy 1979] and Time Warp [Jefferson 1985]
  - Optimization techniques for Time Warp: Lazy cancellation, Moving Time Window (MTW) and Adaptive Time Warp (ATW)

Evaluation and results

1. Comparison among data processing engines
   - Spark was faster than Hadoop MapReduce.
2. Scalability
   - Our simulators could simulate 10^8 nodes with 10 commodity computers.
3. Optimistic parallel simulation
   - It worked.
   - Lazy cancellation was always effective.
   - Moving Time Window (MTW) and Adaptive Time Warp (ATW) reduced memory consumption at the cost of execution time.
4. Performance evaluation
   - 20 times of dPeerSim (parallel) and 1/4 of PeerSim (serial)

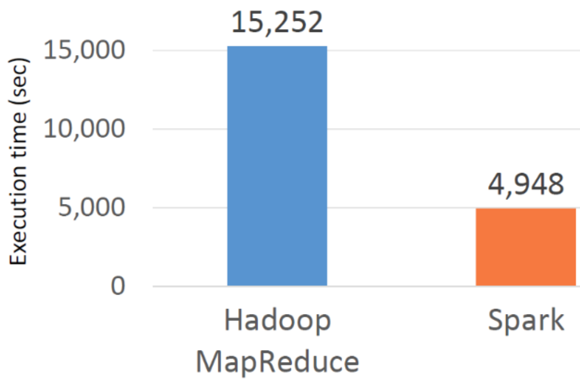
Hadoop MapReduce v.s. Spark



- 10 worker computers with 32 GB of memory running YARN's NodeManager.
  - In all the experiments.
- Gnutella with a complex network generated by Barabasi-Albert (BA) model (m = 1)
  - 100 queries
- Non-optimistic synchronization
  - Although the simulator processes a large number of events because timings of message reception are aligned.

- Spark is faster than Hadoop MapReduce.
  - It eliminates various overheads of Hadoop MapReduce and utilizes memory well.
- Faster engines will show further better results. E.g. Spark4TM

Scalability

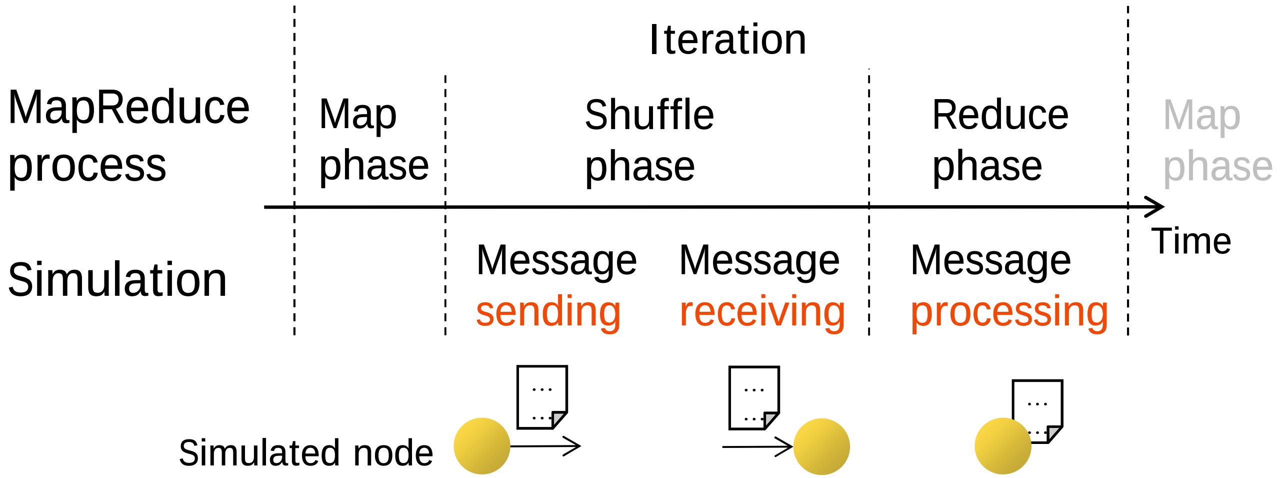


- Gnutella with a complex network generated by BA model (m = 2)
  - 100 queries
- Non-optimistic synchronization

- Our simulators could handle 10^8 nodes with 10 commodity computers with 32 GB of memory.
  - We just confirmed. It will not be the limit.
  - dPeerSim could simulate 5.75 x 10^6 nodes on a single computer with 1.5 GB of memory and 84 x 10^6 nodes on 16 computers. Chord is simulated, not Gnutella.
- They can simulate
  - BitTorrent DHT, one of the largest distributed system (〜 10^7) on a single computer
  - All the things connected to Internet (10^10 〜 in 2020 estimated by Gartner) with 1000 computers ☺

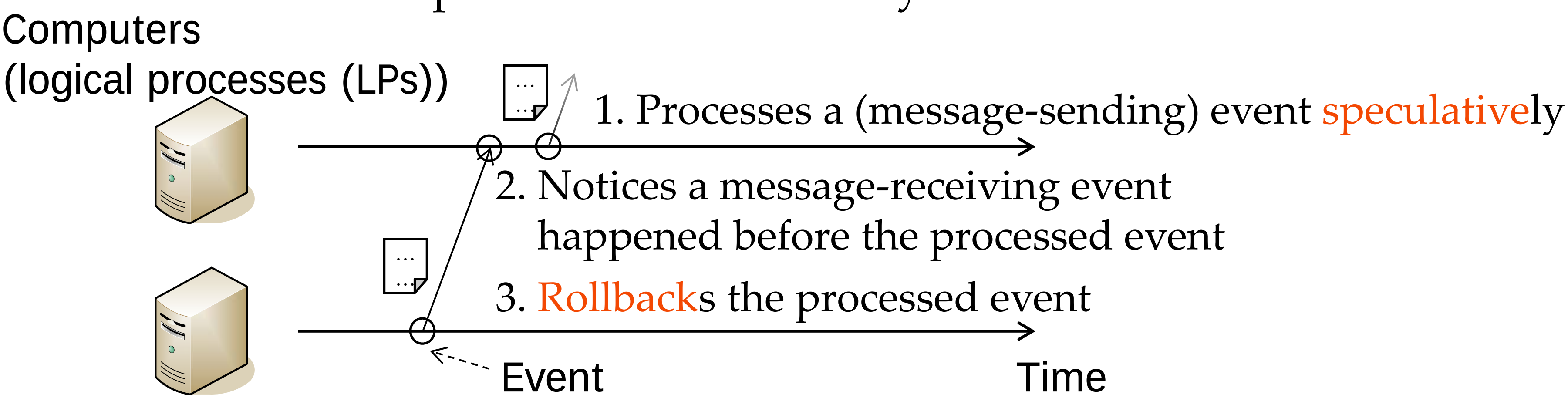
Optimistic parallel simulation

- Our simulator can process very limited number of message-sending events in a MapReduce iteration without an **optimistic** synchronization protocol. ☹
  - At worst, a single message. Because …
  - In MapReduce, communication between nodes is simulated by shuffle phase. Because of it, in an iteration, each node sends messages and then receives messages.
  - A discrete-event simulator processes only the earliest events.



Optimistic parallel simulation

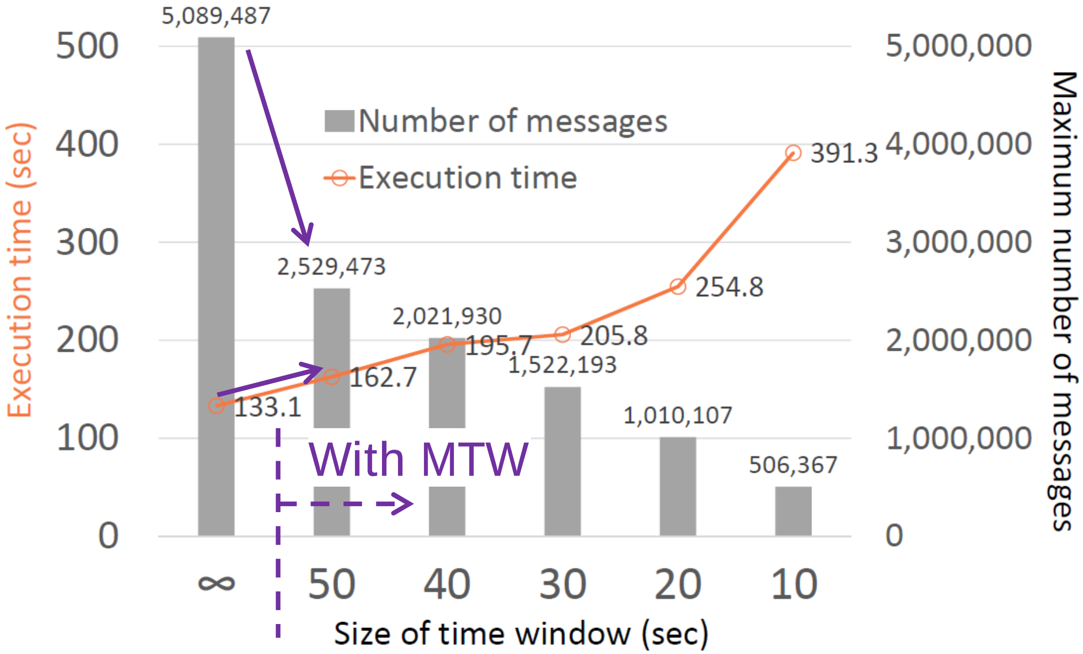
- Time Warp [Jefferson 1985]
  - Each computer processes events speculatively.
  - It rollbacks processed events if they should be cancelled.



- It requires memory / storage to save simulation states and/or events after global virtual time (GVT) = commitment horizon.
  - For rollbacks.
- We try MTW and ATW to control (reduce) memory consumption.
  - It is important because Spark basically places data in memory.

Optimistic parallel simulation

- It works.



- 2D mesh network with 10^6 nodes
  - 10000 queries during 100 sec
- Optimistic synchronization
  - with lazy cancellation

- Moving Time Window (MTW) reduced # of messages in memory at the cost of execution time.
  - MTW limits speculative event processing.
  - The best size of time window depends on a simulation target.
- Adaptive Time Window (ATW) also works as expected. See the paper.

Performance evaluation

- # of events / second
  - Our Spark-based simulator 1.41 × 10^4
    - Optimistic
    - 10 computers

    x 20
  - dPeerSim (parallel) 7.39 × 10^2
    - Non-optimistic - Null message algorithm
    - 16 computers

    x 1/4
  - PeerSim (serial) 6.17 × 10^4
- This result is very preliminary.

	Simulation target	Computers
Our work	Gnutella	2.4 GHz Xeon × 2 × 10, Gigabit Ethernet (2010)
(d)PeerSim	Chord	3.0 GHz Xeon × 2 × 16, Gigabit Ethernet + Myrinet (~2004)

Summary

- Parallel Discrete-Event Simulation (PDES) on data processing engines was demonstrated.
  - On Hadoop MapReduce and Spark
  - Our Spark-based simulator showed x20 performance of dPeerSim thanks to Time Warp, a optimistic synchronization protocol.
  - Optimization techniques for Time Warp worked as expected
    - Lazy cancellation, MTW and ATW.

- Future work
  - Scalability challenge with thousands of computers
  - Confirmation of fault-tolerance features of data processing engines
  - Other simulation targets
  - Comprehensive evaluation:
    Performance, comparison with non-optimistic simulation, …