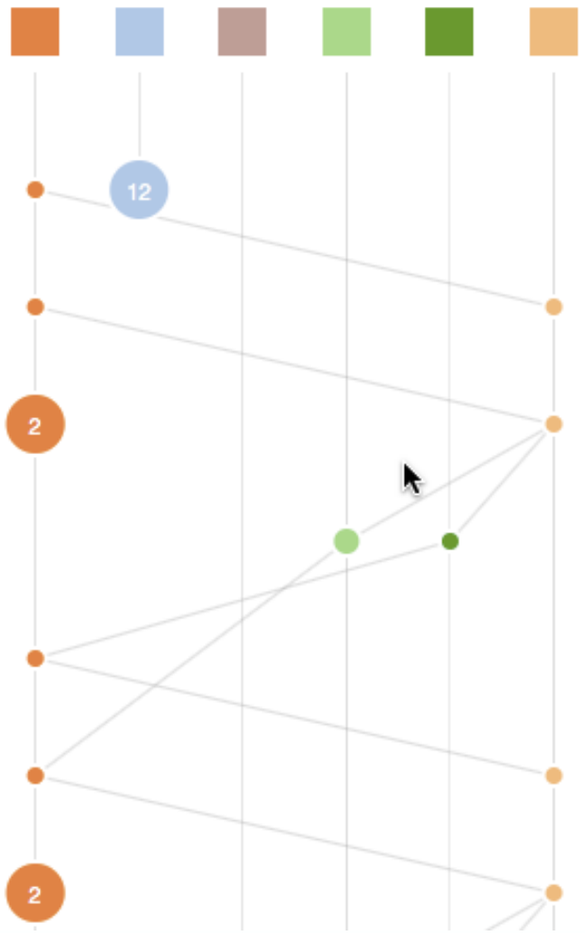


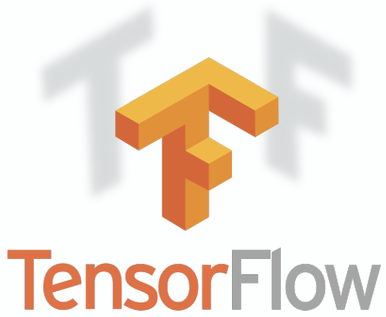
Visualizing distributed system executions

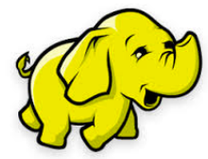
http://bestchai.bitbucket.io/shiviz/

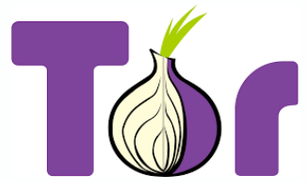
Ivan Beschastnikh, Perry Liu, Albert Xing
Patty Wang, Yuriy Brun, Michael D. Ernst

Distributed systems are everywhere

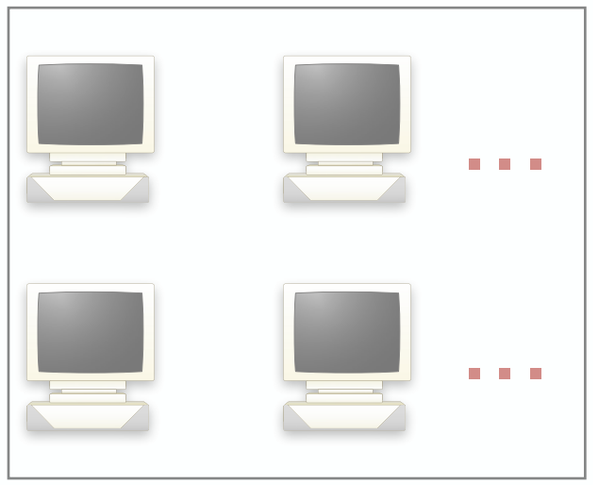
- Parallel data processing
  - TensorFlow
  - Spark, Hadoop
- Data center (cloud computing)
  - Storage: Amazon Dynamo, Google file system, Facebook Haystack
  - Coordination: ZooKeeper, Chubby, Etcd
- Peer-to-Peer and wide-area
  - BitTorrent, Tor
  - DNS, Content distribution networks
- In the small (LAN)
  - Network file system (NFS)







Distributed system pros/cons



- Heterogeneity
  - \+ Resilience (geographic diversity)
  - \- Compatibility
- Distributed state
  - \+ No central point of failure, scalability
  - \- State coherence (distributed consensus)
- Concurrency
  - \+ Parallelism (scalability)
  - \- Race conditions, deadlocks, complexity
- Partial failures
  - \+ Fault tolerance
  - \- Failure recovery, complexity

Coping with SE challenges of dist. sys.

Key categories:

- Testing [Arcuri et al. FSE'15]
- Model checking [MODIST NSDI'09]
- Verification [IronFleet SOSP'15]
- Record and replay [Friday NSDI'07]
- Log analysis [Xu et al. SOSP'09]
- Tracing [Pivot tracing SOSP'15]
- Visualization [De Pauw at al. SoftVis'06]

Exciting research area, increasing industrial relevance

Coping with SE challenges of dist. sys.

Key categories:

- Testing [Arcuri et al. FSE'15]
- Model checking [MODIST NSDI'09]
- Verification [IronFleet SOSP'15]
- Record and replay [Friday NSDI'07]
- Log analysis [Xu et al. SOSP'09]
- Tracing [Pivot tracing SOSP'15]
- Visualization [De Pauw at al. SoftVis'06]

Talk focus

Exciting research area, increasing industrial relevance

Guiding question:

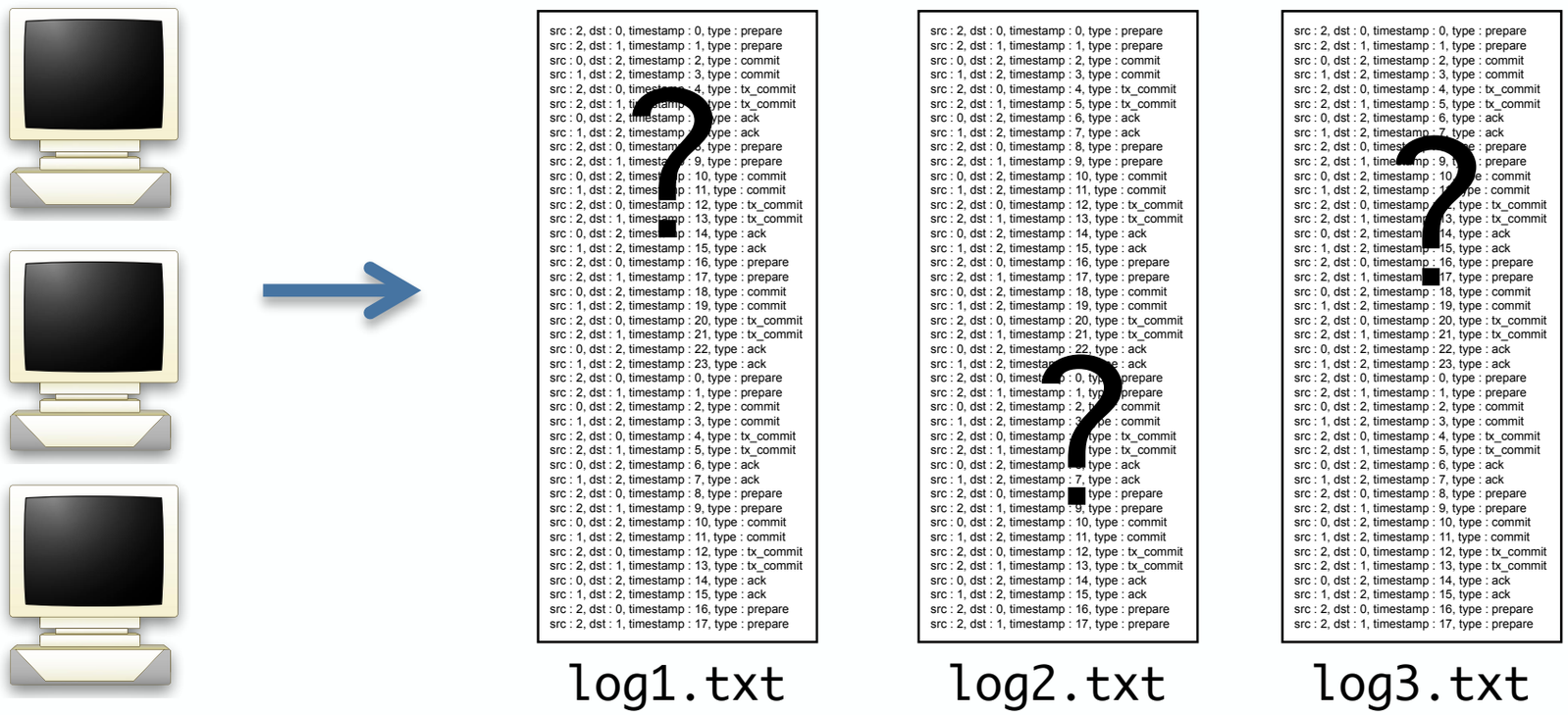
Why does my system behave in a certain manner?

- Were events X and Y concurrent, or did one precede another?
- Did node A ever communicate with node B? When?
- Did node B ever communicate with node C? When?

Guiding question:

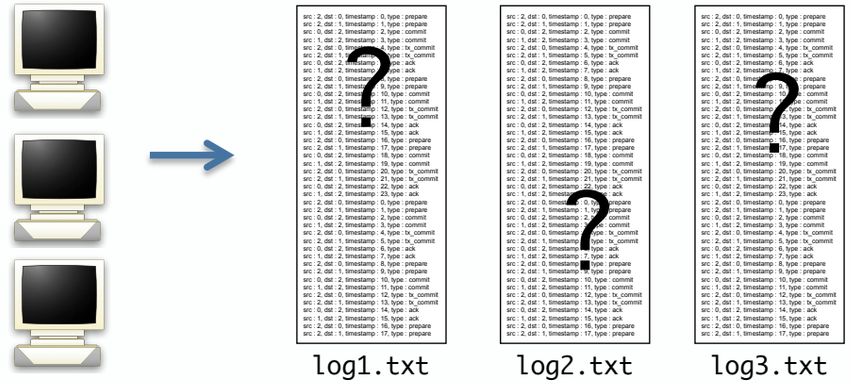
Why does my system behave in a certain manner?

Common solution: log analysis



Guiding question:

Why does my system behave in a certain manner?



Log analysis challenges

- Ordering events between host logs
- Understanding communication patterns
- Comparing two or more logged executions

Missing the right tools

Guiding question:

Why does my system behave in a certain manner?

XVector and ShiViz approach:

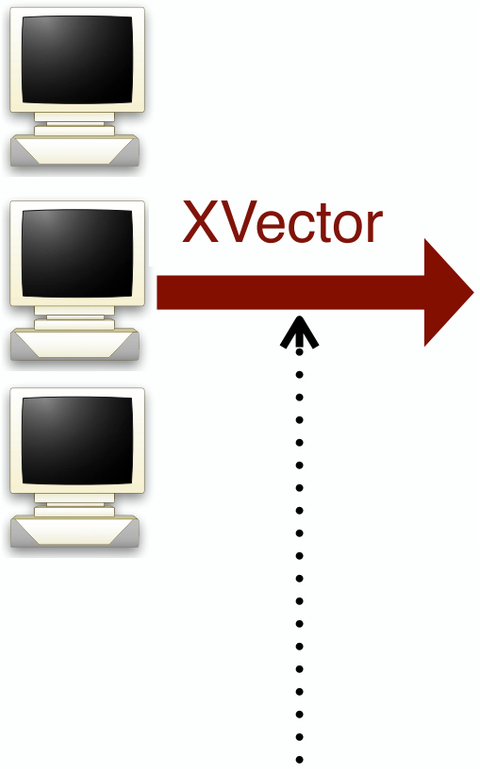
Instrument and analyze

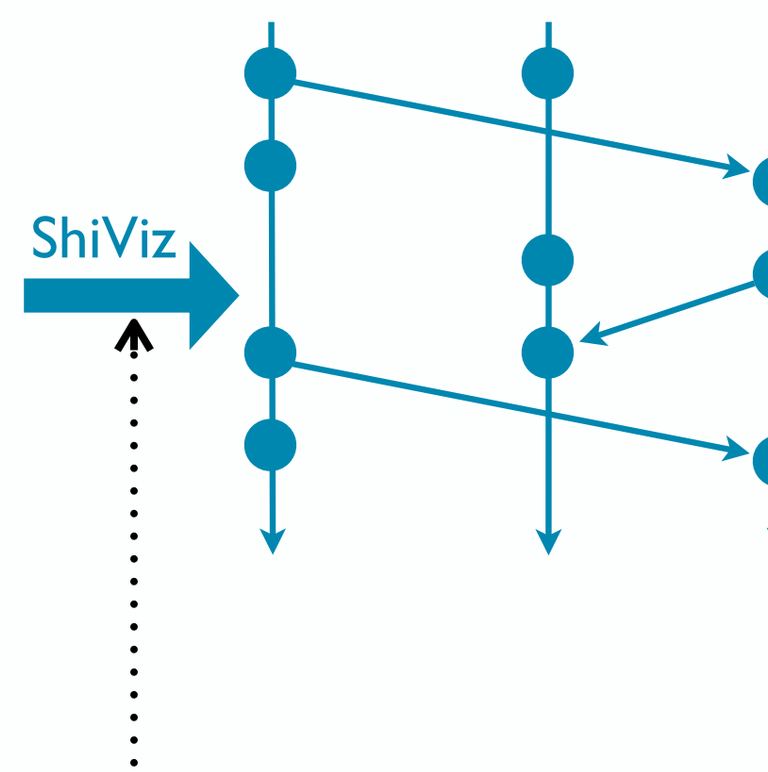
Events

Visualization

Dynamic analysis

Log analysis with ShiViz





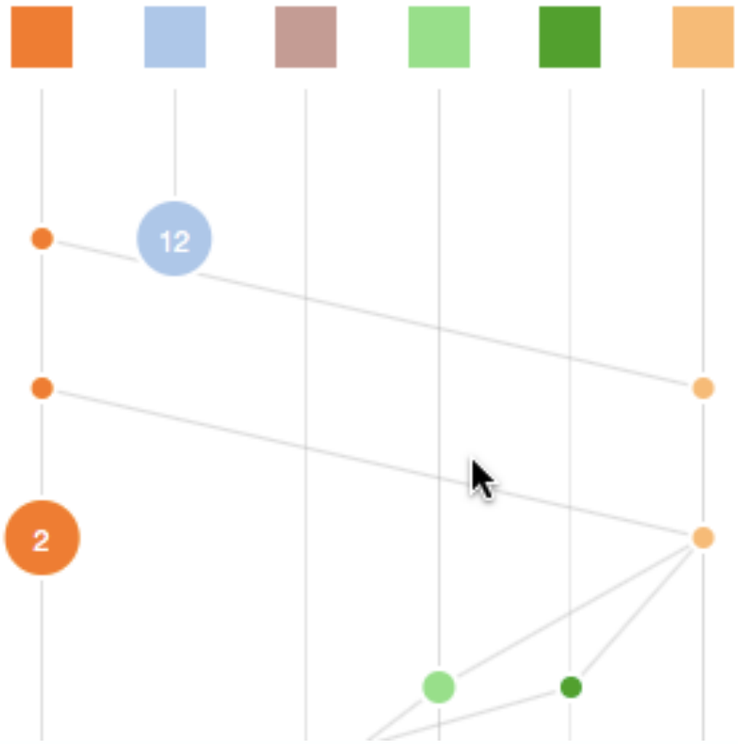
- System instrumentation
  - Nodes maintain vector clocks
  - Logged msgs have vector timestamps
- Used to capture partial order of events

https://github.com/DistributedClocks

- Visualize partial order
- Help developers
  - Understand ordering
  - Query for patterns
  - Compare executions

https://bestchai.bitbucket.io/shiviz/

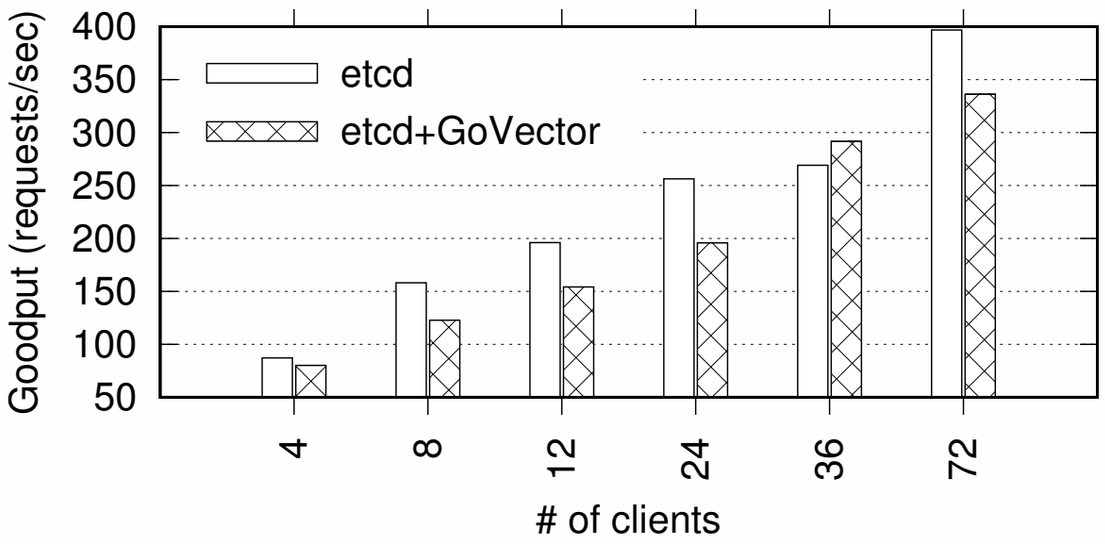
ShiViz demo

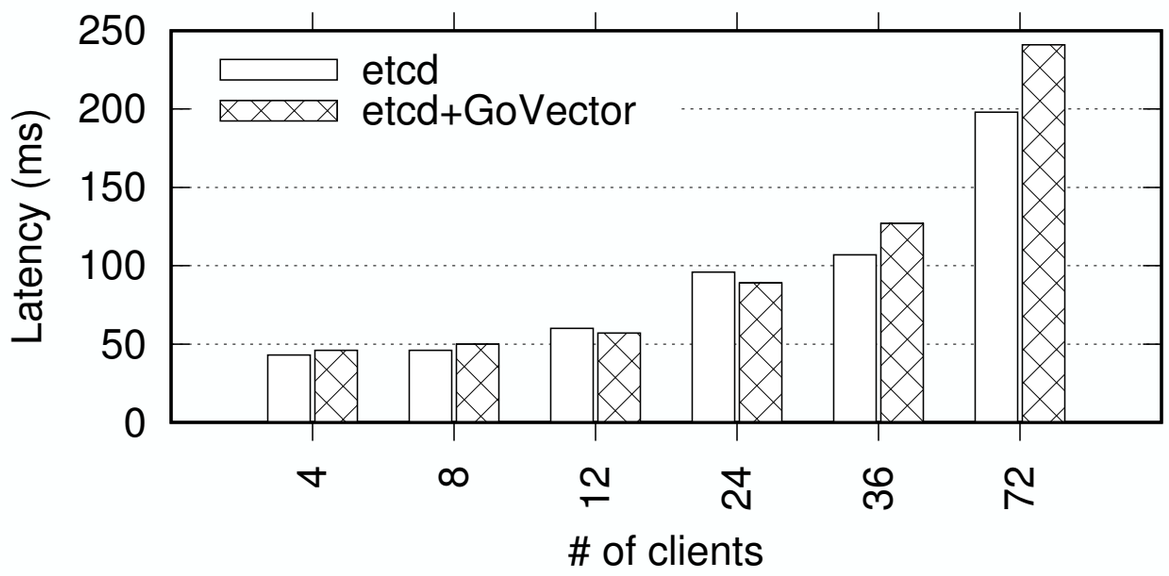


http://bestchai.bitbucket.io/shiviz/

Evaluation and impact

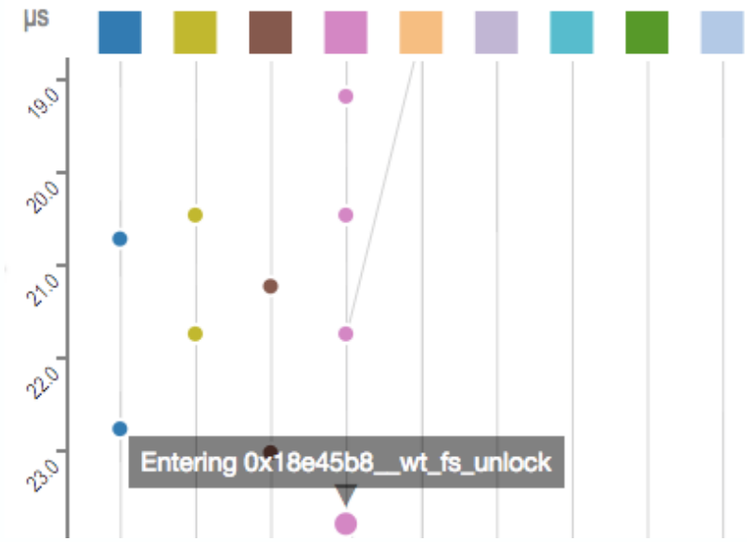
- XVector tools: viable for debugging during development
- Experiments with etcd storage system
  - 1 logging statement takes ~20 microseconds
  - Can execute ~50k logging statements per node before perturbing the system





Evaluation and impact

- XVector tools: viable for debugging during development
- Three user studies with ShiViz:
  - Controlled study with 39 students: structured tasks
  - Open-ended study with 70 students: homework
  - Case study with two systems researchers
- Evidence that ShiViz helps developers understand systems:
  - Relative ordering of events
  - Interaction patterns between hosts
  - Compare executions
- ShiViz used in other projects:
  - P, P#, and TLA+ projects in Microsoft
  - Akka actor-based framework
  - TSViz tool built-on top of ShiViz



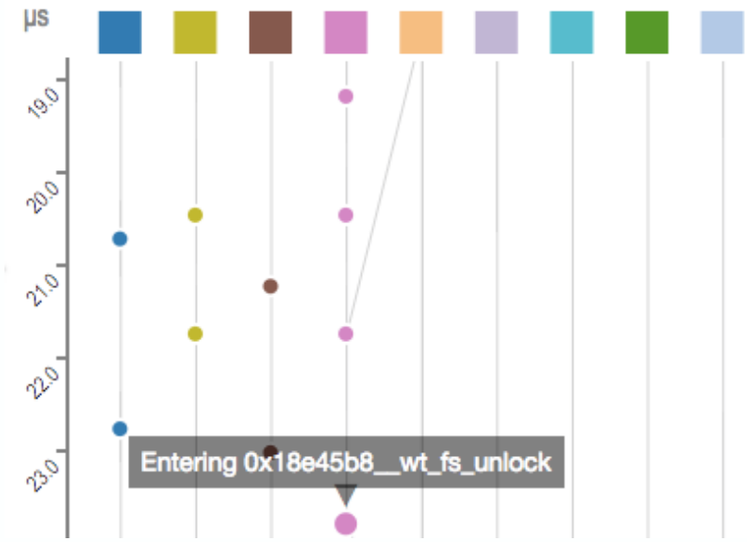
TSViz: *https://bestchai.bitbucket.io/tsviz/*

Evaluation and impact

- XVector tools: viable for debugging during development
- Three user studies with ShiViz:
  - Controlled study with 39 students: structured tasks
  - Open-ended study with 70 students: homework
  - Case study with two systems researchers

See paper for more details!

- Evidence that ShiViz helps developers understand systems:
  - Relative ordering of events
  - Interaction patterns between hosts
  - Compare executions
- ShiViz used in other projects:
  - P, P#, and TLA+ projects in Microsoft
  - Akka actor-based framework
  - TSViz tool built-on top of ShiViz



TSViz: *https://bestchai.bitbucket.io/tsviz/*

Why does my system behave like this?

Approach: instrument and analyze

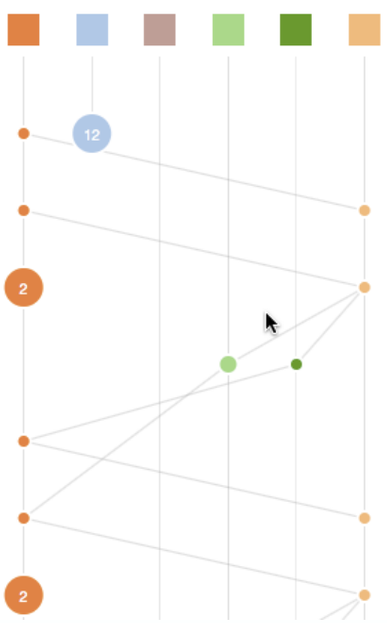
Events

Visualization

Dynamic analysis

★ XVector: capture ordering of events

★ ShiViz: visualize and explore distributed executions



- Understand ordering of events
- Query for interactions between nodes
- Compare pairs of executions

https://github.com/DistributedClocks

http://bestchai.bitbucket.io/shiviz/