Strymon

Queryable Online Simulation for Datacenter Management

Vasiliki (Vasia) Kalavri

29 May 2017, LIQ Shonan Workshop
Kanagawa, Japan

ETH zürich

DCModel Team

John Liagouris

Desislava Dimitrova

Moritz Hoffmann

Andrea Lattuada

Zaheer Chothia

Sebastian Wicki

Timothy Roscoe

http://strymon.systems.ethz.ch


The mess we're in

Data centers are dynamic environments

- Workload fluctuations
- Network failures
- Configuration updates
- Software updates
- Scale-up / down

Test deployment?

Can we **predict** the effect of changes?

Can we **prevent** catastrophic scenarios?

Strymon: The Big Picture

Simulation!

Strymon

Enterprise Datacenter

event logs

Queryable Online Simulation with Strymon

- Cross-layer state overview
- Fault diagnosis and prevention
- Automatic management and configuration
- Resource optimization and capacity planning
- Invariant verification

Inputs

- Network **topology** and events, e.g. switch failure, link capacity update, congestion
- Performance **traces** from running applications
- **Static** data, e.g. configurations, VM placement, device specifications
- **Monitoring** data, e.g. resource consumption

Strymon

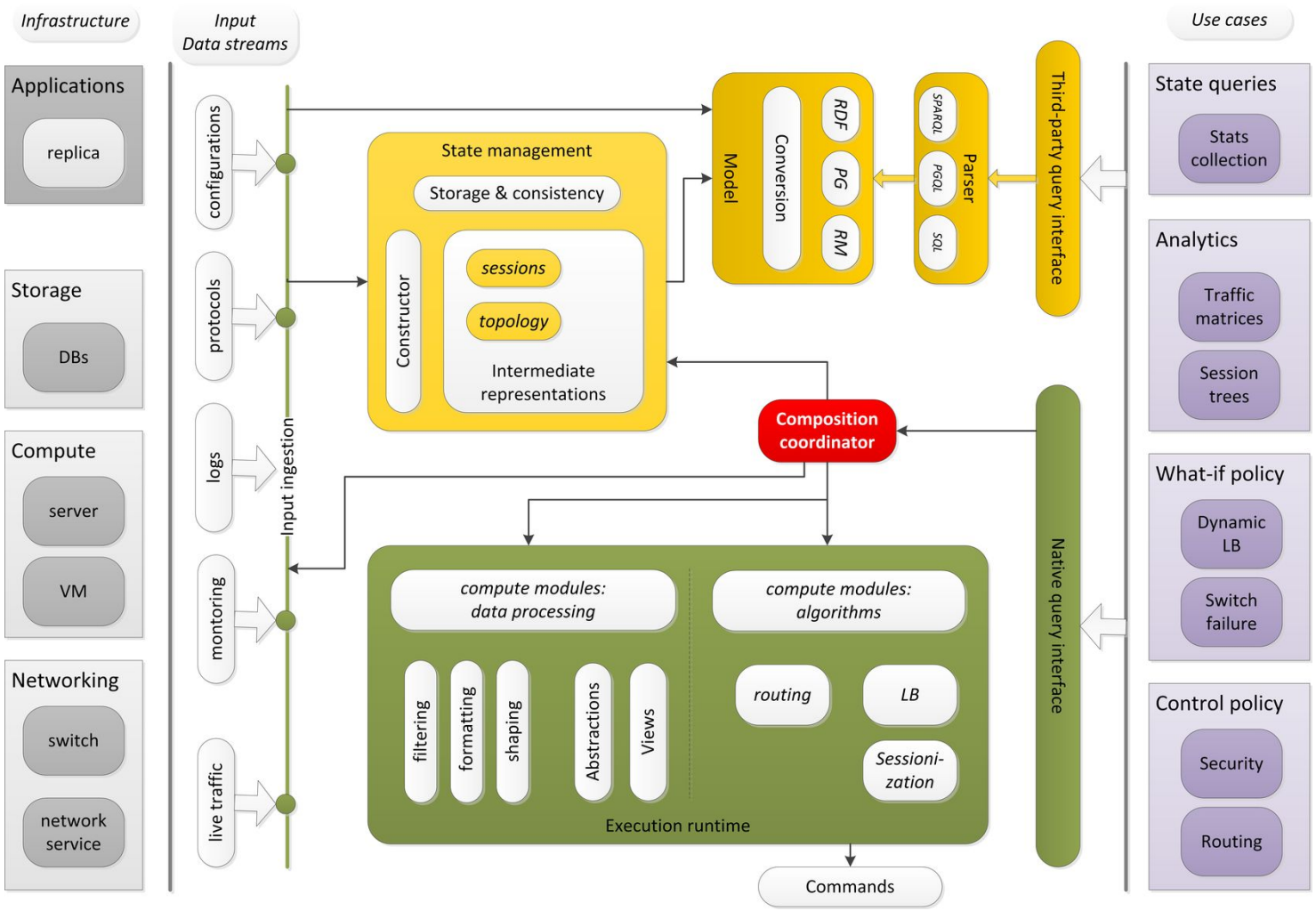
Queries

- Look-up queries
  - Retrieve the core switch network topology
  - What is the load on that link?
- Analytics
  - Traffic matrices
  - Session trees
- What-if
  - We use alternative load-balancing?
  - A link or switch fails?
- Policy enforcement
  - Security
  - Packet forwarding

Operational Requirements

- **Low latency:** react quickly to network failures, prevent faults
- **High throughput:** keep up with high stream rates
- **Fault-tolerance:** operate reliably and provide state guarantees
- **Incremental computation:** reuse already computed results when possible, e.g. do not recompute forwarding rules after a link update

Strymon Architecture Vision



Timely Dataflow

- A steaming framework for data-parallel computations
  - Arbitrary cyclic dataflows
  - Logical timestamps (epochs)
  - Asynchronous execution
  - Low latency, modest resources

Rust implementation: **github.com/frankmcsherry/timely-dataflow**

D. Murray, F. McSherry, M. Isard, R. Isaacs, P. Barham, M. Abadi. ***Naiad: A Timely Dataflow System.*** *In SOSP, 2013.*

Differential Dataflow

- A “high”-level API on top of Timely Dataflow
  - Common data-parallel operators, e.g. group_by, join, map
  - Automatic incremental computation
  - Evolving collections abstraction, indexed by logical timestamp

Rust implementation: **github.com/frankmcsherry/differential-dataflow**

F. McSherry, D. Murray, R. Isaacs, M. Isard. ***Differential Dataflow.*** *In CIDR, 2013.*

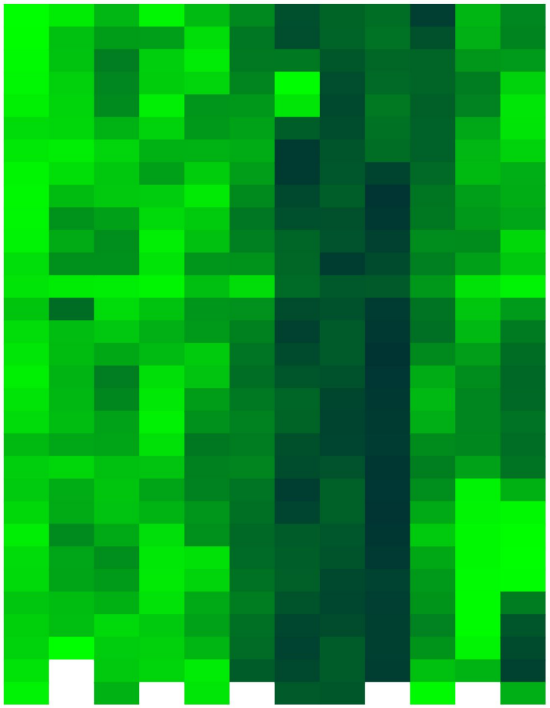
A what-if use-case

Use-case: evaluate load balancing strategies

Industry partner data center:

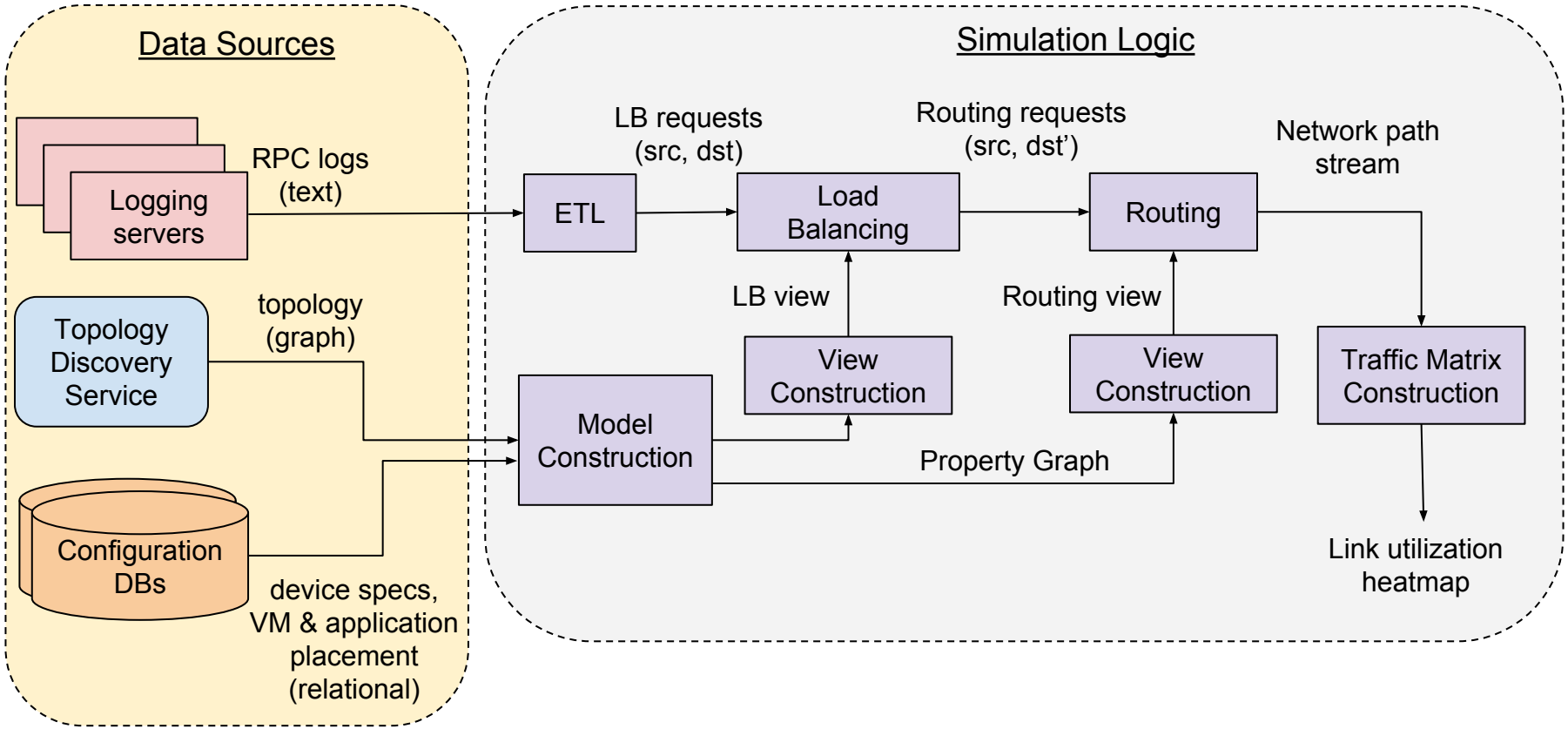
- 13k services
- ›100K user requests/s
- Each request is assigned a unique id on entry
- All participating components gather traces locally and periodically send them to logging servers
- OSPF routing
- Weighted Round-Robin load balancing

Can we estimate the network link utilization under topology and load balancing strategy changes?

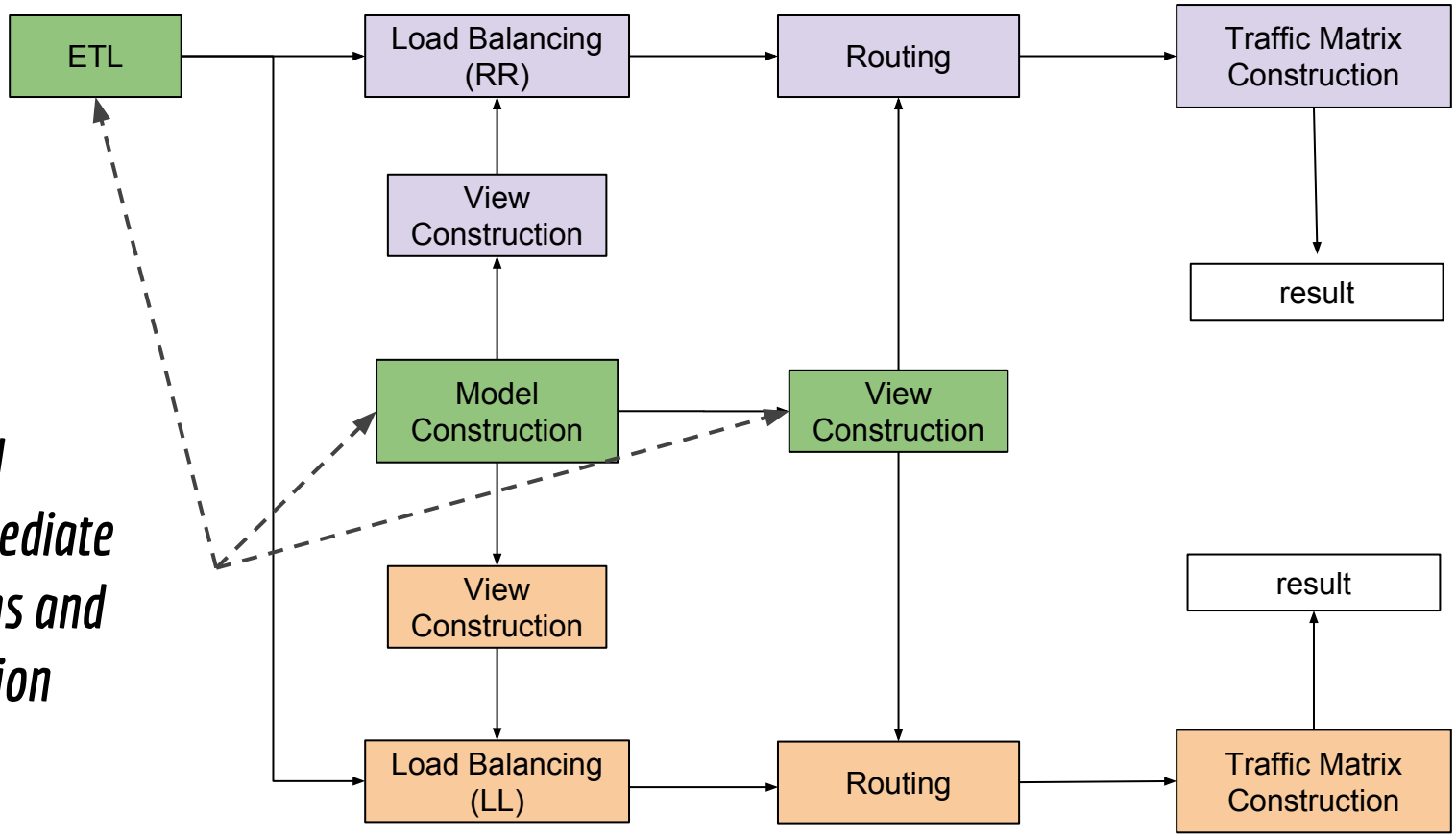


Traffic matrix simulation

Use-case: evaluate load balancing strategies



Use-case: evaluate load balancing strategies



What else can Strymon do already?

- Online sessonization

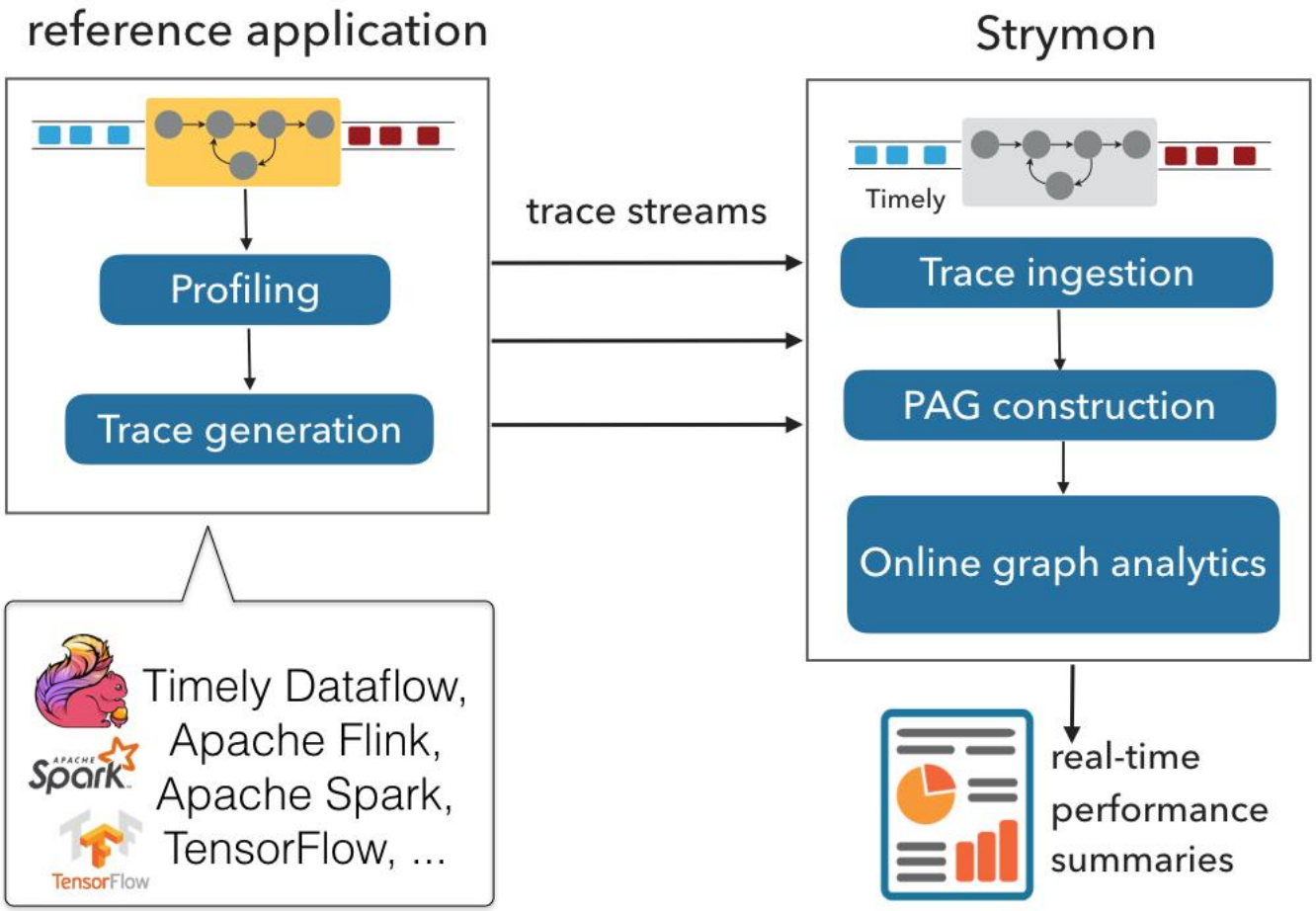
Zaheer Chothia, John Liagouris, Desislava Dimitrova, Timothy Roscoe. Online Reconstruction of Structural Information from Datacenter Logs. EuroSys, 2017.

- Interactive explanation of simulations

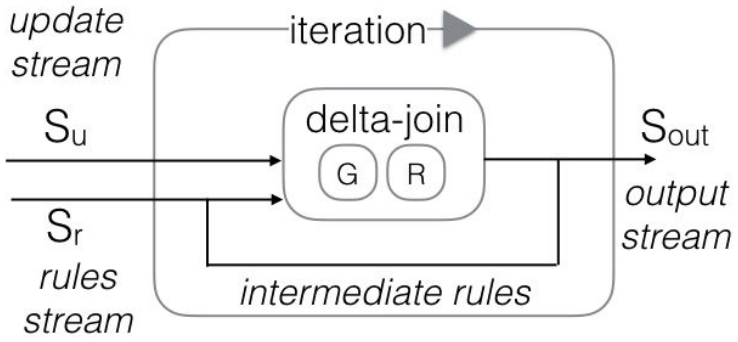
Zaheer Chothia, John Liagouris, Frank McSherry, Timothy Roscoe. Explaining Outputs in Modern Data Analytics. PVLDB9(12):1137-1148, 2016.

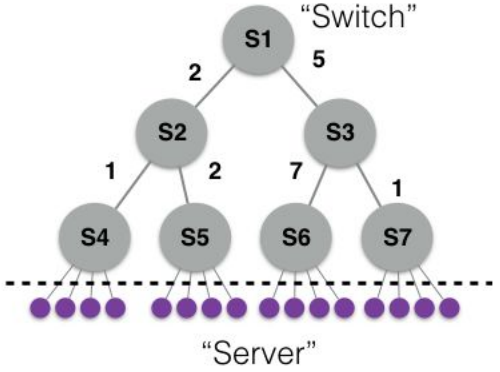
- Online Performance Analysis
- Fast Incremental Routing

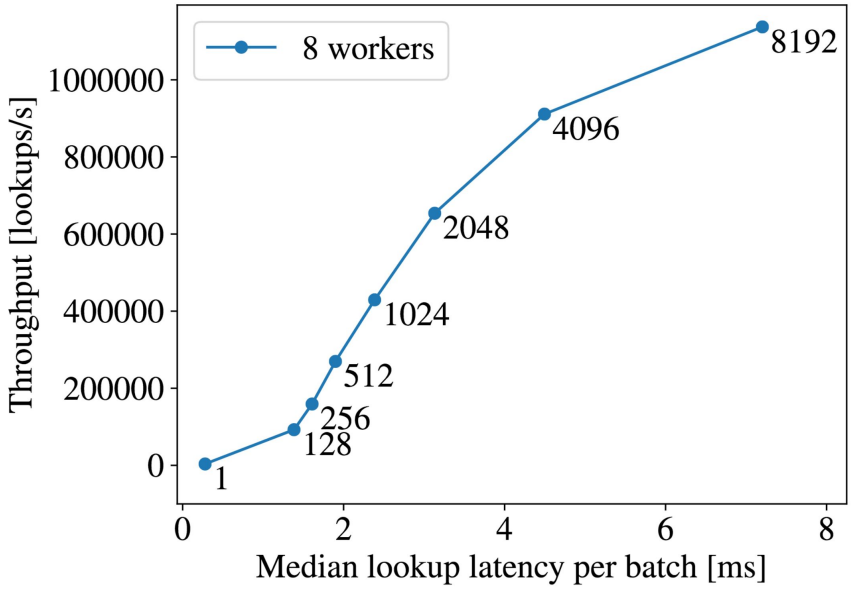
Online performance analysis



Fast incremental routing







Open issues and work-in-progress

Is Timely right for Strymon?

- Low latency ✓
- High throughput ✓
- Incremental computation ✓
- Deployment & resource management ✗
- API / programming model ✗
- Fault-tolerance & state management ✗

A unified query language for Strymon

- Look-up queries
  - Retrieve the core switch network topology
  - What is the load on that link?
- Analytics
  - Traffic matrices
  - Session trees
- What-if
  - We use alternative load-balancing?
  - A link or switch fails?
- Policy enforcement
  - Security
  - Packet-forwarding

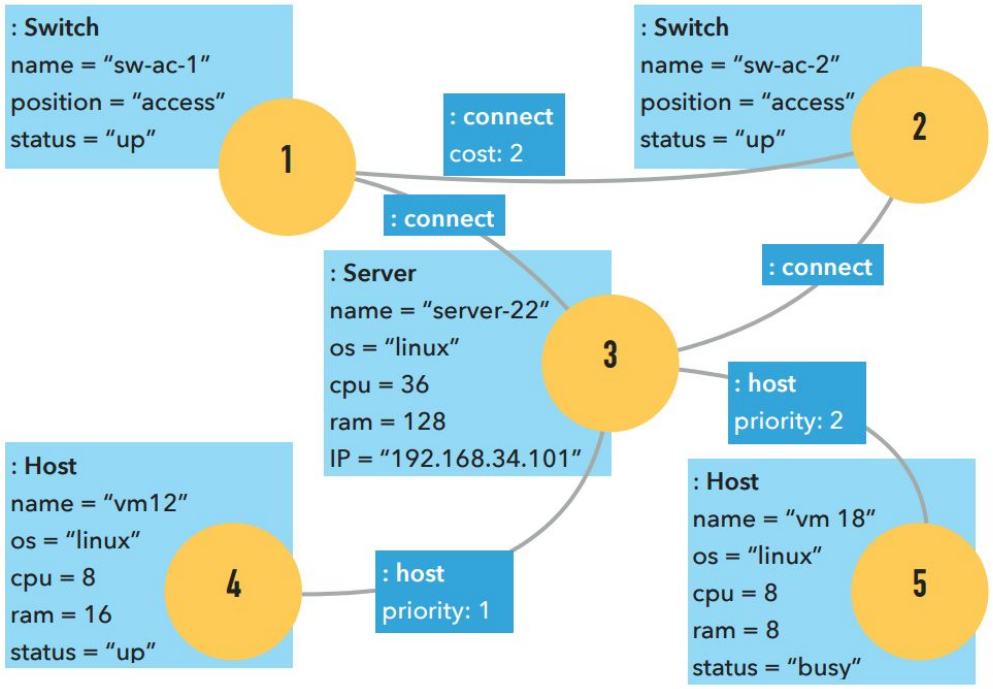
SQL? Cypher? PGQL?

Dataflows?

Frenetic?

A unified data model for Strymon

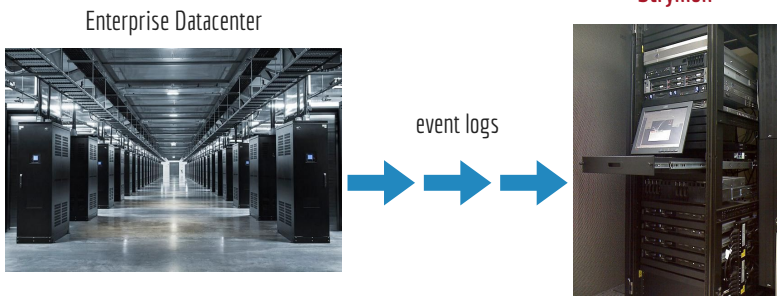
Property Graphs are suitable for modeling the network and building controller applications.

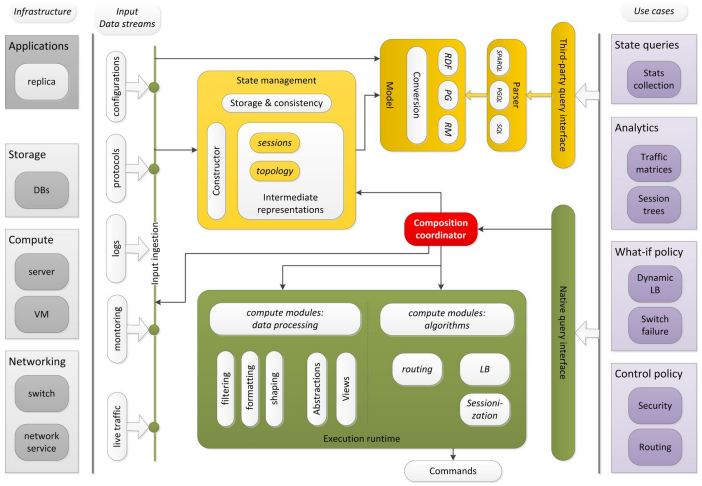


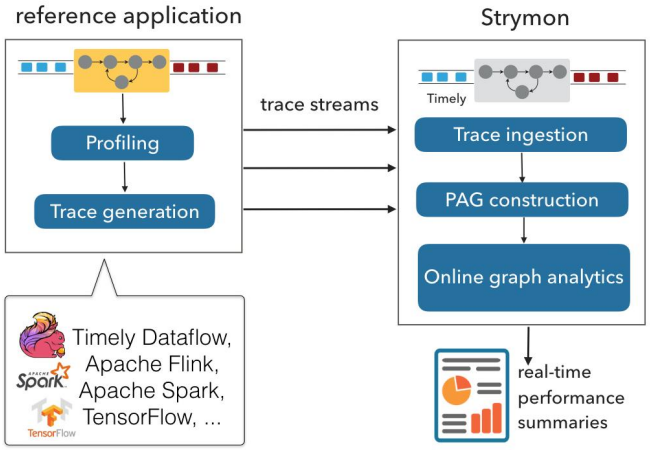
A unified data model for the data center

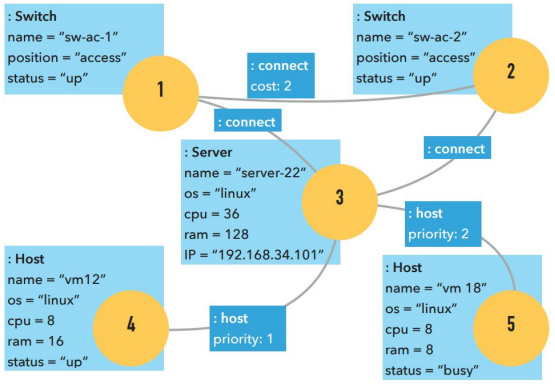
- Is the Property Graph a suitable model for the data center **across layers**?
- Is there a better model for **highly dynamic** topologies?
- Can we create a **unified model** that encapsulates all Strymon input data?
- How to **expose the streams** to applications in a common way?
- Shall we use multiple models and **convert** from one to another?
- How to efficiently **translate** a high-level query language to a Timely dataflow?

Conclusion


Enterprise Datacenter
event logs
Strymon


Infrastructure
Input Data streams
Use cases
Applications
replica
Storage
DBs
Compute
server
VM
Networking
switch
network service
configurations
protocols
logs
monitoring
live traffic
Input ingestion
State management
Storage & consistency
sessions
topology
Intermediate representations
Constructor
Model
Conversion
RDF
PG
RM
Parser
Cypher
PGQL
SQL
Third-party query interface
Composition coordinator
compute modules: data processing
compute modules: algorithms
filtering
formatting
shaping
Abstractions
Views
routing
LB
Sessioni-zation
Execution runtime
Commands
Native query interface
State queries
Stats collection
Analytics
Traffic matrices
Session trees
What-if policy
Dynamic LB
Switch failure
Control policy
Security
Routing


reference application
Strymon
Profiling
Trace generation
trace streams
Timely
Trace ingestion
PAG construction
Online graph analytics
Timely Dataflow,
Apache Flink,
Apache Spark,
TensorFlow, …
real-time
performance
summaries


: Switch
name = "sw-ac-1"
position = "access"
status = "up"
1
: connect
cost: 2
: Switch
name = "sw-ac-2"
position = "access"
status = "up"
2
: connect
: Server
name = "server-22"
os = "linux"
cpu = 36
ram = 128
IP = "192.168.34.101"
3
: connect
: host
priority: 2
: Host
name = "vm12"
os = "linux"
cpu = 8
ram = 16
status = "up"
4
: host
priority: 1
: Host
name = "vm 18"
os = "linux"
cpu = 8
ram = 8
status = "busy"
5

Strymon

Queryable Online Simulation for Datacenter Management

Vasiliki (Vasia) Kalavri

29 May 2017, LIQ Shonan Workshop
Kanagawa, Japan

ETH zürich