Hecate: AI-driven WAN Traffic Engineering for Science

Mariam Kiran, Energy Sciences Network
Scott Campbell, Energy Sciences Network
Nick Buraglio, Energy Sciences Network

Overview

Hecate in 14 words:

Multi-objective path optimizer driven by historical endpoint behaviors, live and predicted health, and topology.

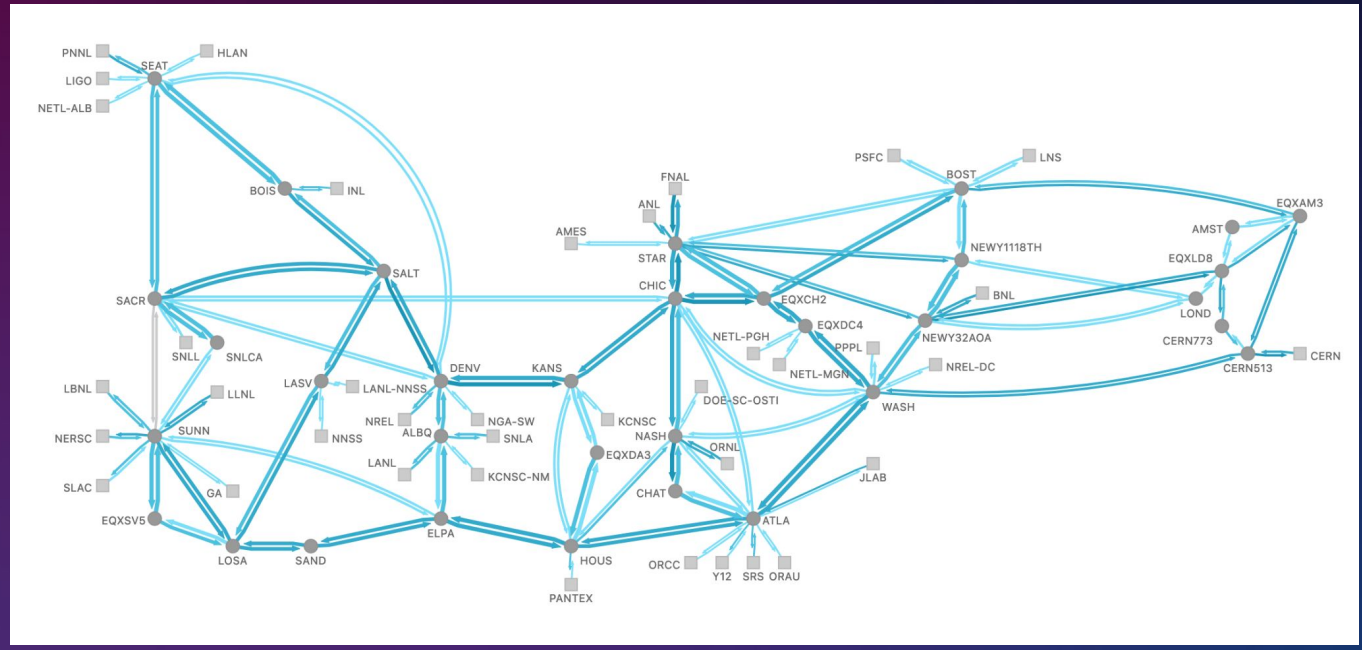
- Problem we are solving
- General solution rational
- What is Hecate?
- Architecture
- Components/Data Details
- PCE
- Conclusion

Problem Statement

Our requirements:

- High performance throughput with low loss for *huge* time sensitive data transfers
- Latency sensitive communications: cloud, video, command/control for engineering
- Bandwidth reservation: OSCARS

This is a Multi-Objective optimization problem

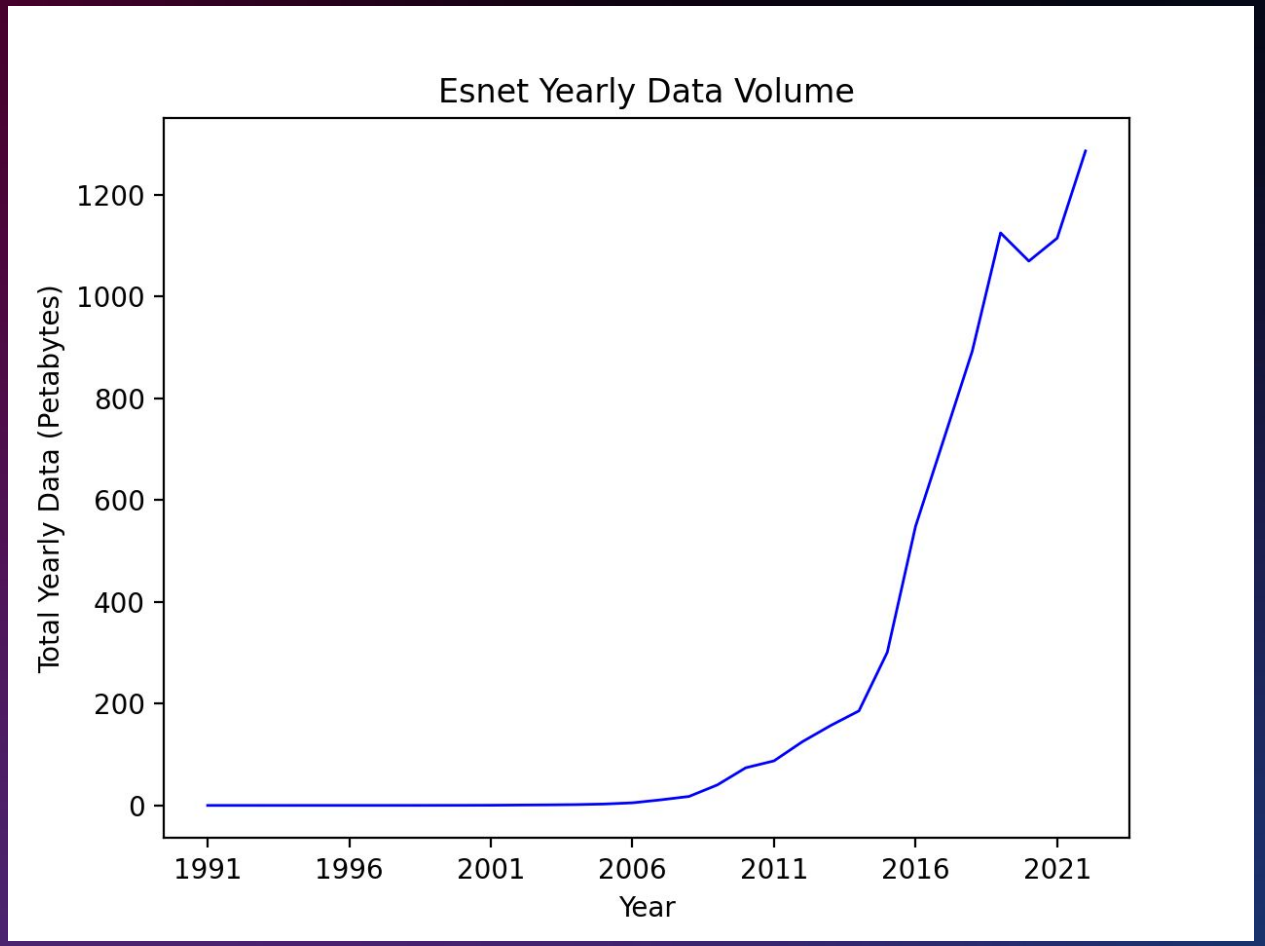


ESnet Background

Data Volume over time:

- User data
- LHC1 Traffic
- OSCARS

Data volume is increasing at an *exponential* rate.



ESnet6: A Brief into…

Development of ESnet6 gives a number of new tools to help with this problem

- Optical Layer
- Integrated Telemetry
- Network Automation
- Segment Routing as Traffic Engineering capability
- PCE as controller for Segment Routing

Other R&E networks are developing similar technologies in parallel which is exciting

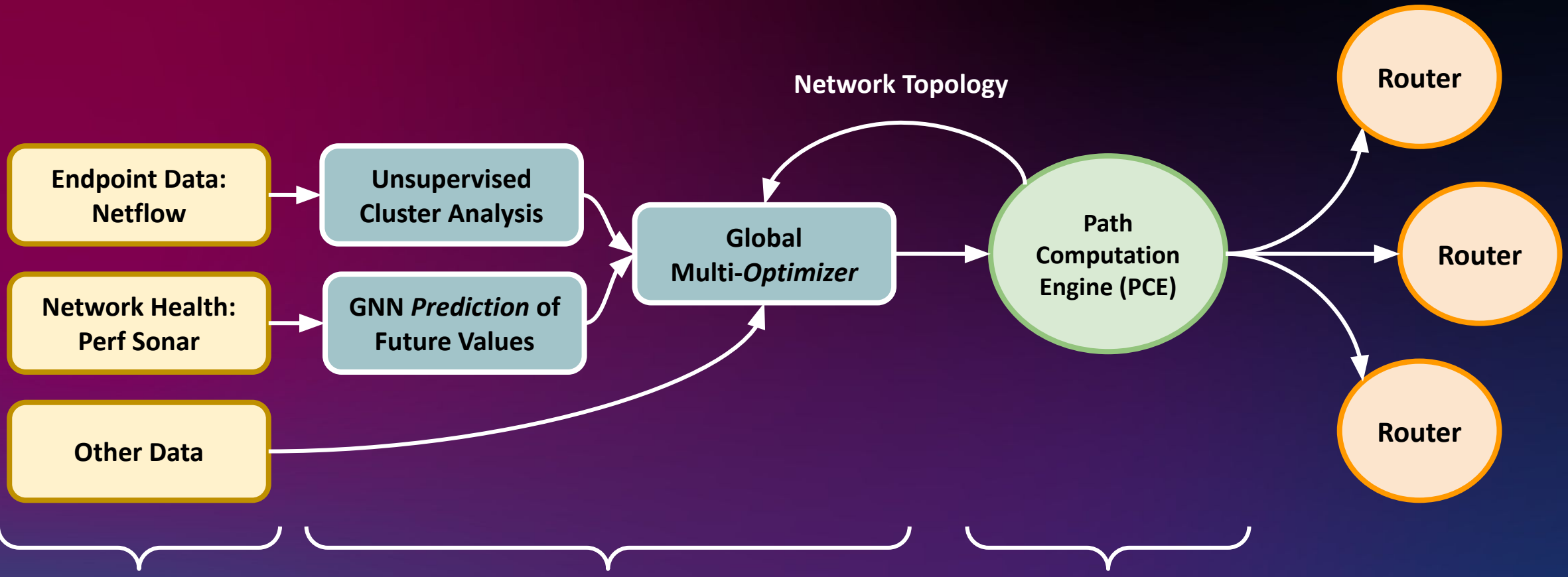
Back to the Problem

Need to use the network we have more efficiently and with greater dexterity

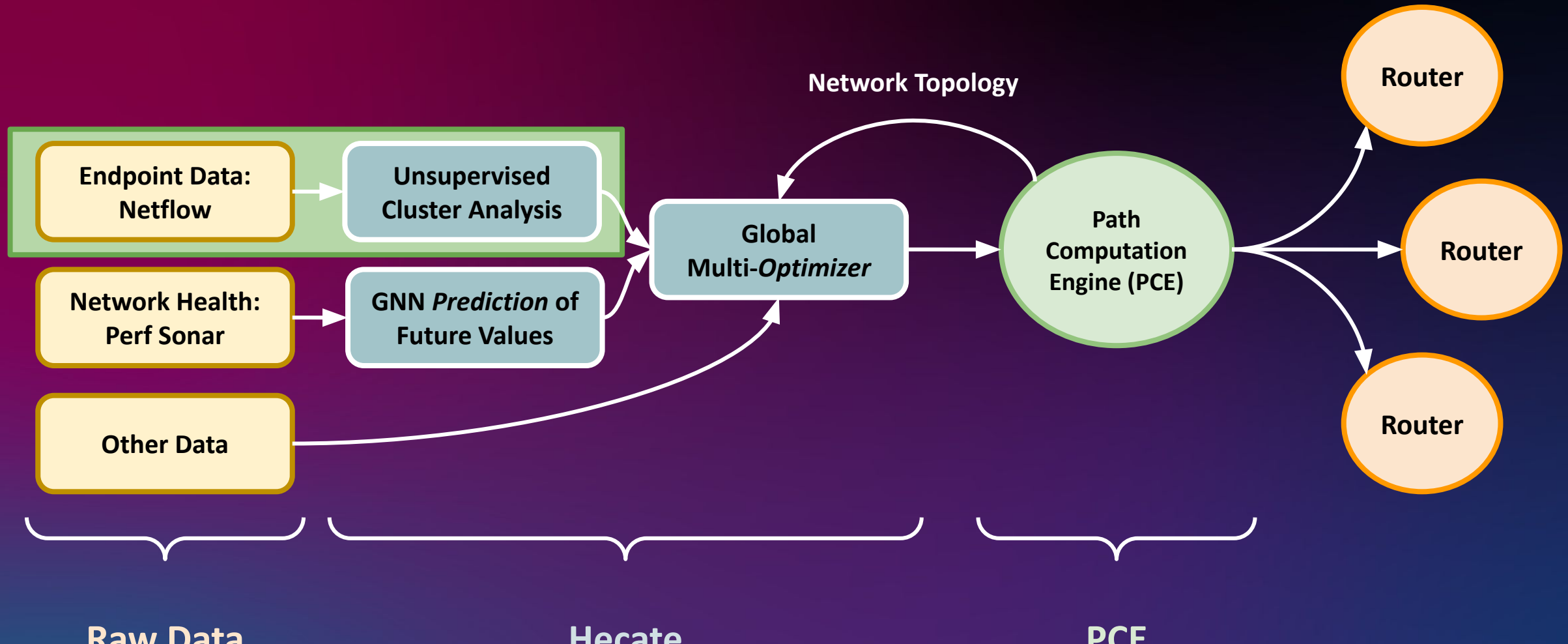
Use *traffic engineering* as a way to address both of these concerns - new network is designed with a less static format which allows us to run it much “hotter” while being able to change the flow control characteristics without logging into each router. Becomes an *optimization* problem.

Tool for this is our research subject: Hecate

Hecate Architecture: Overview



Hecate Architecture: Overview

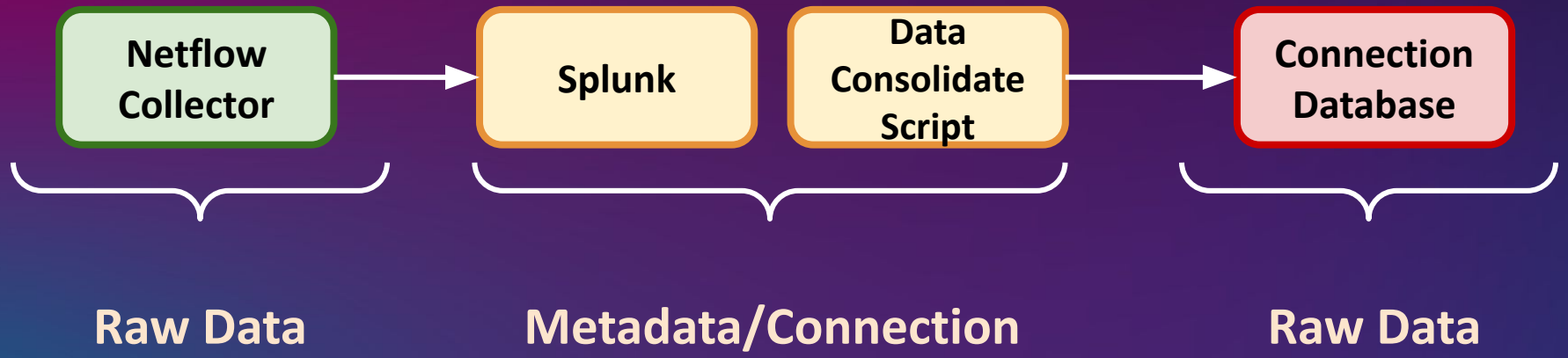


Hecate Architecture: Endpoint/Sites

Key idea - looking at historical behavior provides a slower moving statistical view of behaviors of network sized objects

Can analyze sampled netflow records and do a sliding window analysis to adjust for changing behaviors

Workflow:



```
# Script for Splunk query
# Record Filter
index=netflow protoid = 6 src_sysnum=SRC_ASN dest_sysnum=DEST_ASN
# Initial basic parsing
| eval server=if(src_port<dest_port, "src", "dest")
| eval server_ip=if(server="src", src_ip_clean, dest_ip_clean)
| eval client_ip=if(server="dest", src_ip_clean, dest_ip_clean)
...
# Stateful bookkeeping during processing loop
| eventstats max(flow_end_time_milli) as end_time by index
| eventstats min(flow_start_time_milli) as start_time by index
| eventstats sum(server_bytes) as total_server_bytes by index
# Final data bokeeping
| eval dt = (end_time - start_time)/1000
| eval start_time_sec = start_time/1000
| eval total_packets = total_client_packets + total_server_packets
| eval total_bytes = total_client_bytes + total_server_bytes
| eval server_avg_size = if(total_server_packets>0,round(total_server_bytes/total_server_packets,0),0)
| eval client_avg_size = if(total_client_packets>0,round(total_client_bytes/total_client_packets,0),0)
| eval client_velocity = if(dt>0,round(total_client_bytes/dt,1),0)
| eval server_velocity = if(dt>0,round(total_server_bytes/dt,1),0)
| eval cs_data_ratio = if(total_server_bytes>0,round(total_client_bytes/total_server_bytes,3),0)
| eval cs_data_ratio_norm = if(cs_data_ratio<1 AND cs_data_ratio!=0,round(1/cs_data_ratio,5),cs_data_ratio)
| eval cs_psize_ratio = if(server_avg_size>0,round(client_avg_size/server_avg_size,5),0)
| eval cs_psize_ratio_norm = if(cs_psize_ratio<1 AND cs_psize_ratio!=0,round(1/cs_psize_ratio,5),cs_psize_ratio)

# Noise filter
| where total_client_packets > 5 AND total_server_packets > 5
| dedup index

# Output
| table start_time_sec,dt,client_sysnum,server_sysnum,index,client_ip,client_port,server_ip,server_port ...
```

Data Acquisition and Parsing

Record Loop

Post Loop Eval

Hecate Architecture: Endpoint/Sites

Key idea - l ical view of behaviors c

Can analyz o adjust for changing b

Workflow:

client_site_id	LBNL
server_site_id	ANL
start_time_sec	1657324815.796
dt	21584.099
client_sysnum	16
server_sysnum	683
…	…
cs_data_ratio_norm	1.05263
cs_psize_ratio_norm	1.07649
client_velocity	10.3
client_avg_size	380.0
server_velocity	10.8
server_avg_size	353.0
total_packets	1243
total_bytes	454925
total_client_packets	583
total_client_bytes	221670
total_server_packets	660
total_server_bytes	233255

nnection
atabase

Raw Data | Metadata/Connection | Raw Data

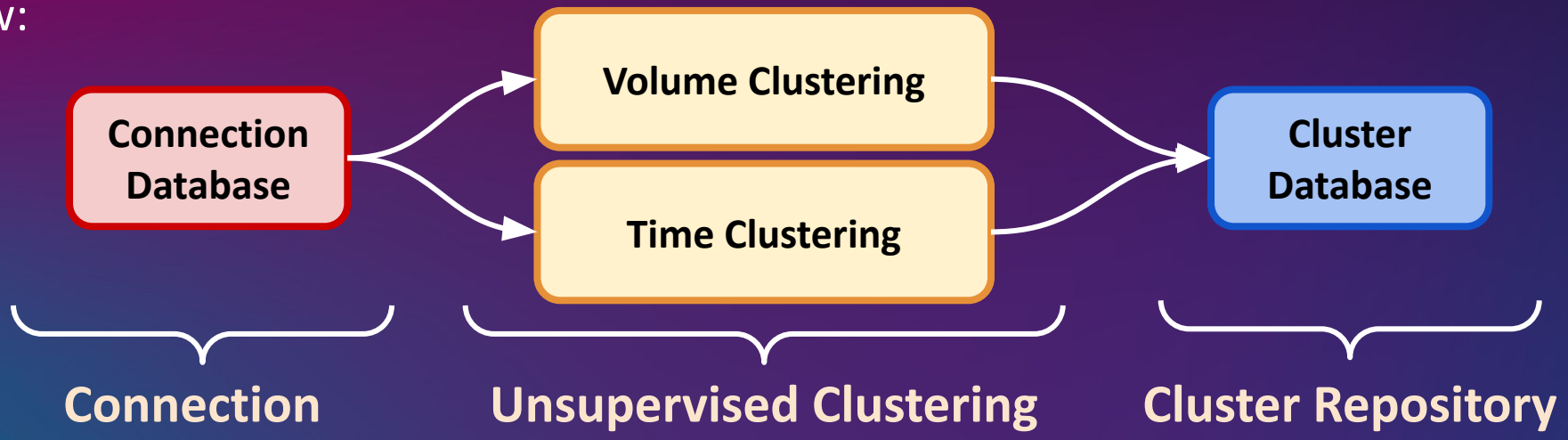
Hecate Architecture: Endpoint/Sites : Clustering

Apply unsupervised **k-means clustering** to data in Connection Database to identify site/source net ⇒ destination net connections that exhibit jumbo/mice/long/short behaviors.

Also looked at Gaussian Mixture Model (GMM) re clustering, and Stochastic Neighbor Embedding (t-SNE) for data dimension reduction.

Used to identify long term structural behaviors of network users since highly transient behavior is less useful for large scale optimization.

Workflow:



Hecate Architecture: Endpoint/Sites : Clustering

Apply unsupervised k-means clustering to data in Connection Database to identify site/source net ⇒ destination net connections that exhibit jumbo/mice/long/short behaviors.

Used to identify long term structural behaviors of network users since highly transient b

V

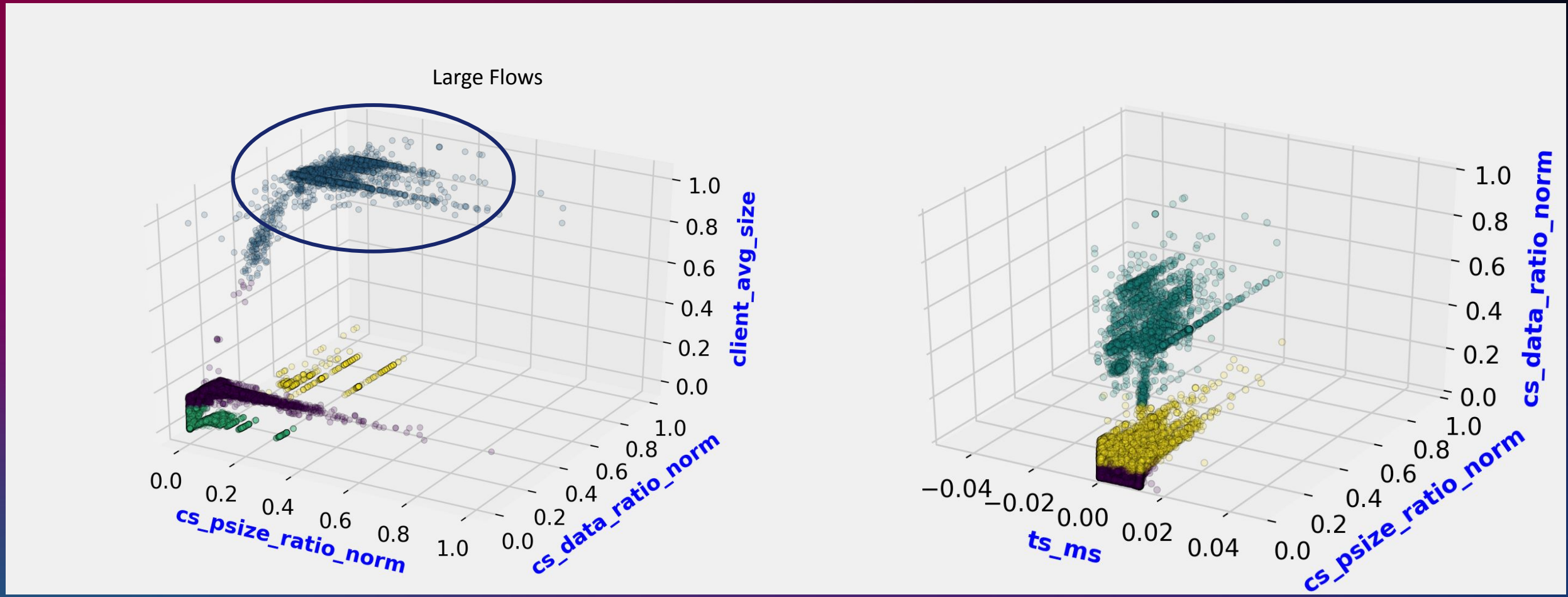
```
src_net ..... 192.148.94.0
dest_net ..... 192.148.93.0
type1 ..... 544 (INTERACT|SHORT)
weight1 ..... 4
type2 ..... 512 (INTERACT|NA)
weight2 ..... 27
update_date ..... 1657537872.783
insert_date ..... 1657419031.675
server_site ..... ORNL
client_site..... ANL
```

Connection

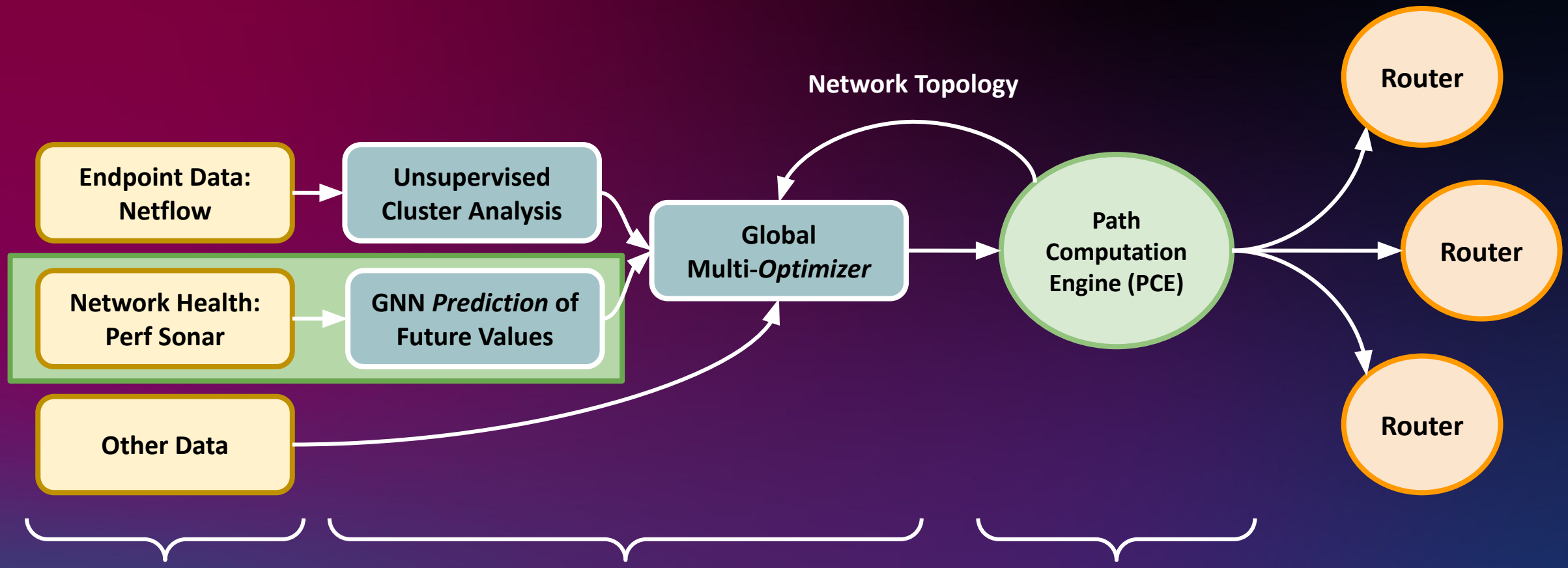
Unsupervised Clustering

```
Type Table
type: data 0xAAA
     0x000: default, no change :
     0x100: jumbo              : 256
     0x200: interact           : 512

type: time 0xBB
     0x00: default, no change  :
     0x10: long                : 16
     0x20: short               : 32

     So long jumbo is 0x110 = 272
```

Hecate Architecture: Endpoint/Sites : Clustering



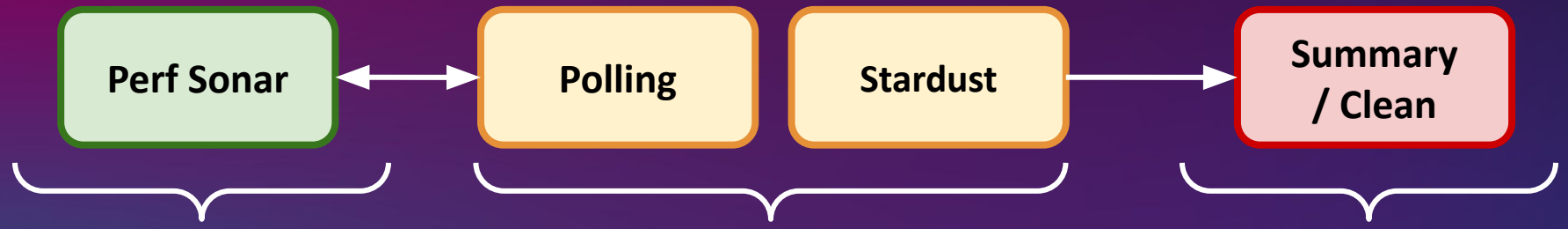
Hecate Architecture: Overview



Hecate Architecture: Network Health Data

Network health data contains highly transient data values for:

Packet Loss	Latency
Jitter	Utilization



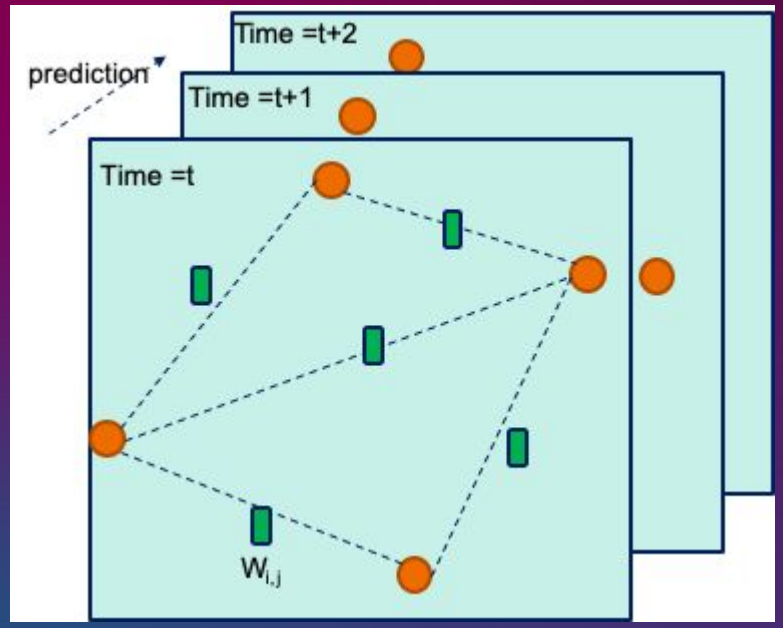
Hecate Architecture: Network Health : Predict

Take most current health data and use to predict values for the next several time steps.

Based on work previously done

Model the network as discrete aggregated network traffic at time t, $G_t = (V, E, W)$

V: Measurement Nodes
E: Edges of Network
W: Distance among Sites



Model:

- Stack of spatio-temporal convolution **blocks**
- **Output** layer.

Each block consists of two temporal gated layer and a spatial graph layer in between.

Output layer consists of convolution, normalized and a fully convolution layer.

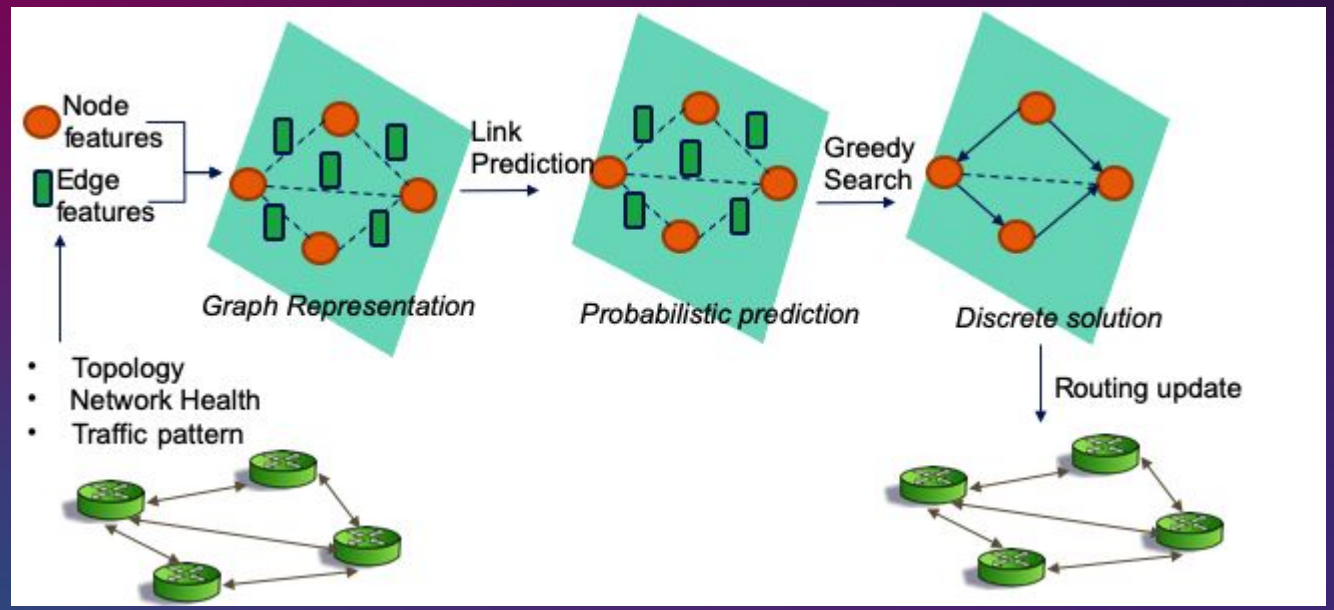
Hecate Architecture: Global Optimizer

Based on work from DeepRoute research project

Use DRL to greedy Q-learning to simulate networks and learn optimal routing strategies for single optimizations

Significant movement in this field - exploring additional options

Hecate uses four types of reward functions in DRL for Graph Optimization: Loss, Latency, Jitter, Utilization



Path Computation Engine

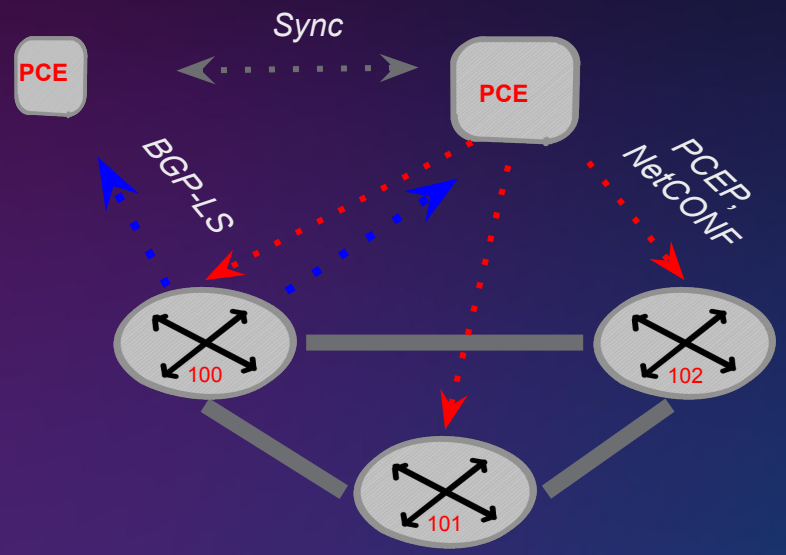
"Brains" of the segment counting core infrastructure

Like MPLS, can stack labels which define path through network

Provides programmatic access to network control:

- Read network topology, router details, performance data
- Write to API to provide "suggestions" for path selection

We do not want to replace a routing protocol, just provide good advice



Challenges

Technical Scope

- Almost every box and arrow in the architecture diagram is a research project.
- Measuring benefit in a complex system

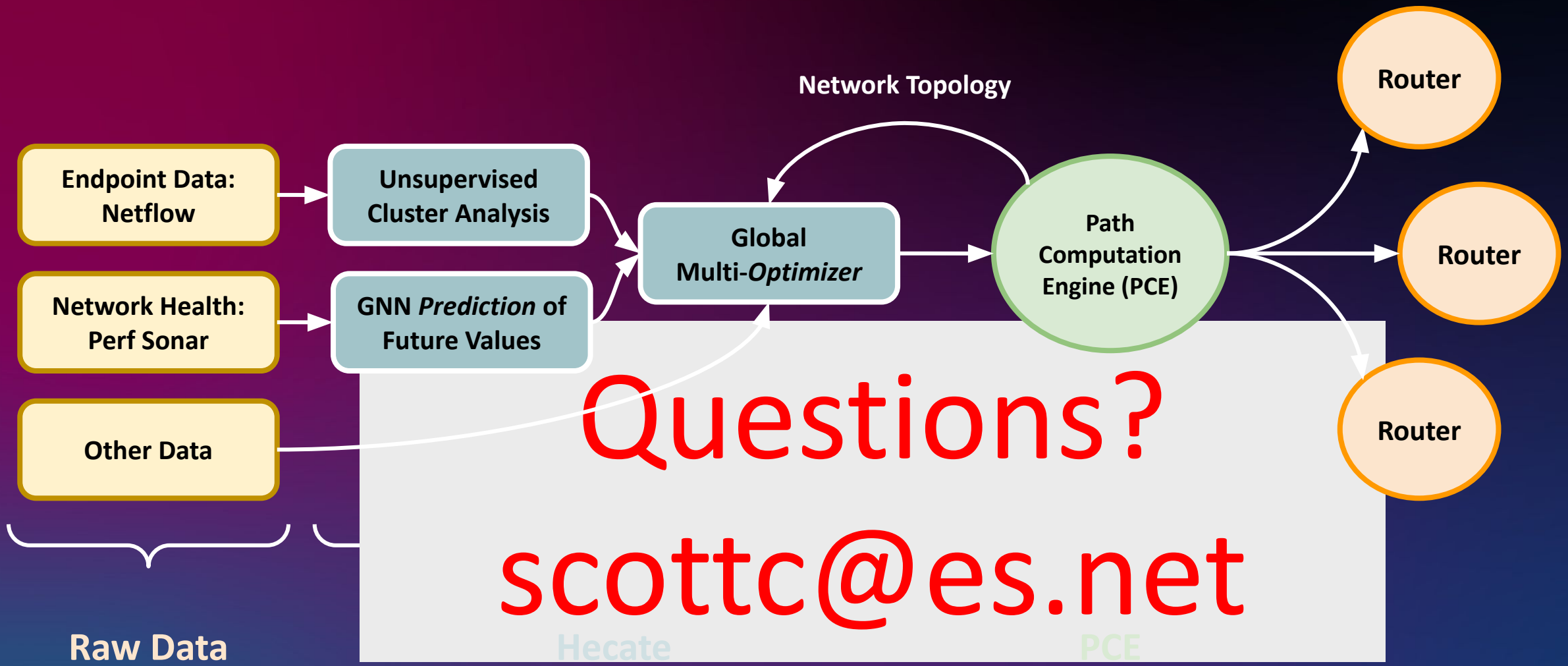
Safety

- When shouldn't we make updates? (Too much, too little)
- Quantify changes and add guard rails

Data Handling:

- Real data is cranky and resistant to being helpful for fragile tools

FIN



Questions?

scottc@es.net