Scalable Near Real-Time Failure Localization of Data Center Networks

Herodotos Herodotou, Bolin Ding, Shobana Balakrishnan

Geoff Outhred, Percy Fitter

Microsoft Research and Azure

Microsoft

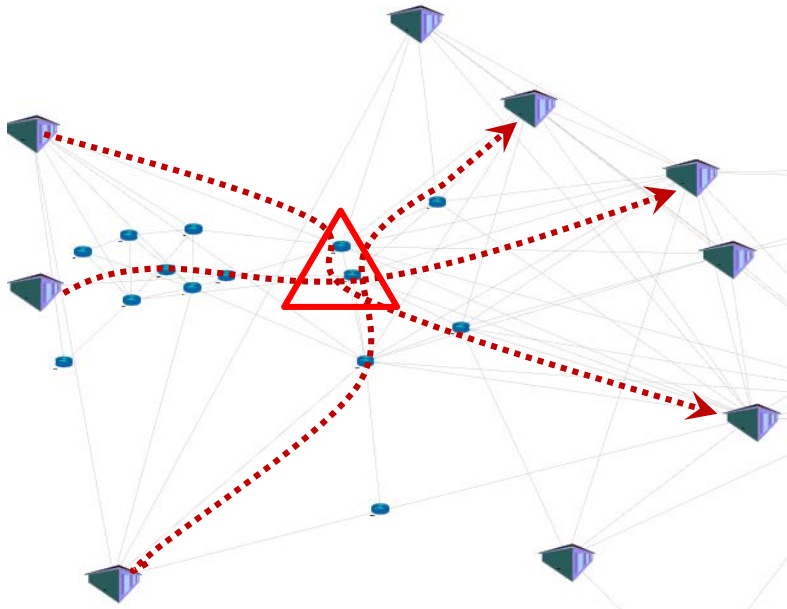
Windows Azure Network

- Worldwide network with ~10K devices and ~200K links
- All Azure services depend on it!
- Goal: Localize *user-perceived network failures* in entire Azure network in near real-time
- Sources of monitoring data
  - Events and alerts from device/link-level telemetry
    - Often noisy
  - Canary Service
    - Proxy for user-perceived network availability & latency

Canary Service

- Operation
  - One canary instance per rack per compute cluster
  - Periodically send *pings* to all Azure clusters

- Failed pings
  - Indicate presence of issue in the network
  - *Do not pinpoint* which device/link has the issue

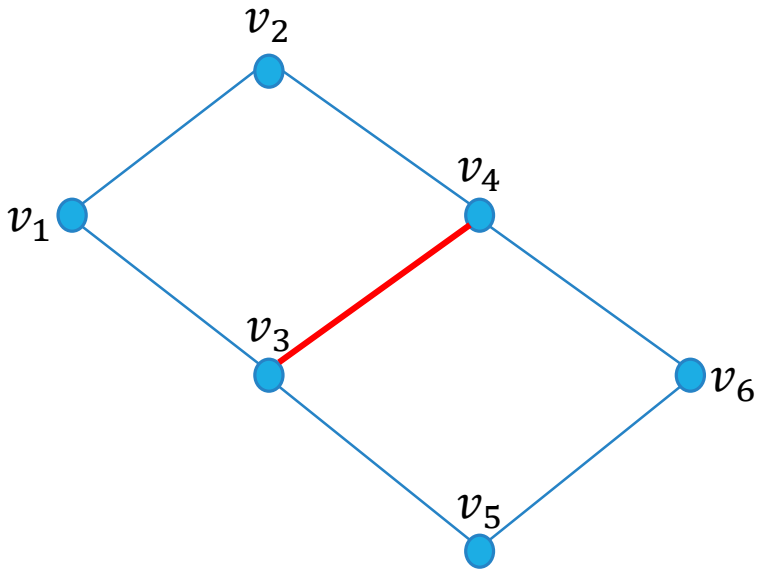


- Our approach: Use statistical data mining techniques on canary ping data to localize user-perceived network failures

A Data Mining Problem

Input

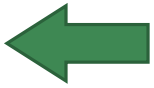
- Graph - Network topology
  - Vertices: Network devices
  - Edges: Network links
- Canary pings between vertices
  - Successful
  - Failed (or w/ high latency)



Output

- A ranked list of edges/vertices associated with failure scores

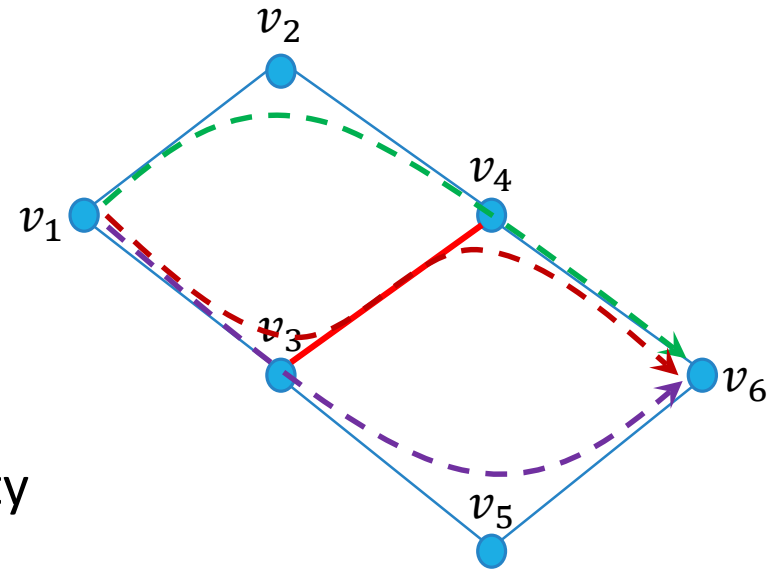
Edge	Score
(v_3, v_4)	0.71
(v_1, v_3)	0.42
(v_1, v_2)	0.37
…	…



Source	Dest	#Succ Pings	#Fail Pings
v_1	v_4	88	8
v_1	v_6	98	5
v_5	v_6	91	0
…	…	…	…

Localizing Network Failures

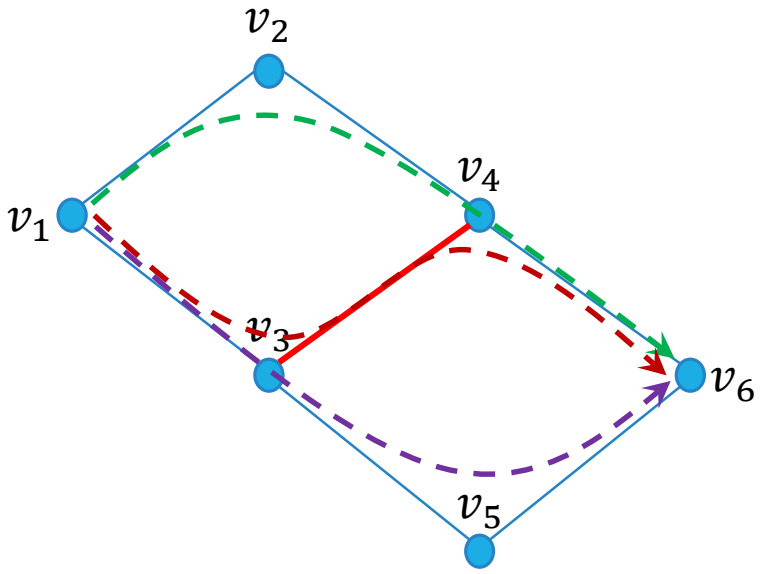
- Challenges
  - Unknown ping routes
  - Partial network failures
- Probabilistic failure model
  - Routing based on shortest paths
  - Model failure scores as probability
- Example
  - Edge $e = (v_3, v_4)$ is dropping packets
  - Vertex v_1 sends $n_{1,6} = 40$ pings to v_6
  - $\Pr[\, v_1 v_6 \text{ ping passes } e] = P_{1,6}(e) = 0.25$
  - $\Pr[\, \text{ping passing } e \text{ will fail}] = X_e = 0.2$
  - Expected number of ping failures: $f_{1,6} = 40 \cdot 0.25 \cdot 0.2 = 2$



Localizing Network Failures (2)

- Failure Model
  - $f_{i,j} = n_{i,j} \cdot P_{i,j}(e) \cdot X_e$

observed values — Routing probability — unknown value



- Pings between multiple vertices
  - $f_{1,6} = n_{1,6} \cdot P_{1,6}(e) \cdot X_e$
  - $f_{1,5} = n_{1,5} \cdot P_{1,5}(e) \cdot X_e$
  - $f_{2,5} = n_{2,5} \cdot P_{2,5}(e) \cdot X_e$
  - …

- Goal: Find X_e that **best fits** the observed data
  - Approach: Least Squares
  - Find X_e's for all edges/vertices and produce ranked list

Other Approaches

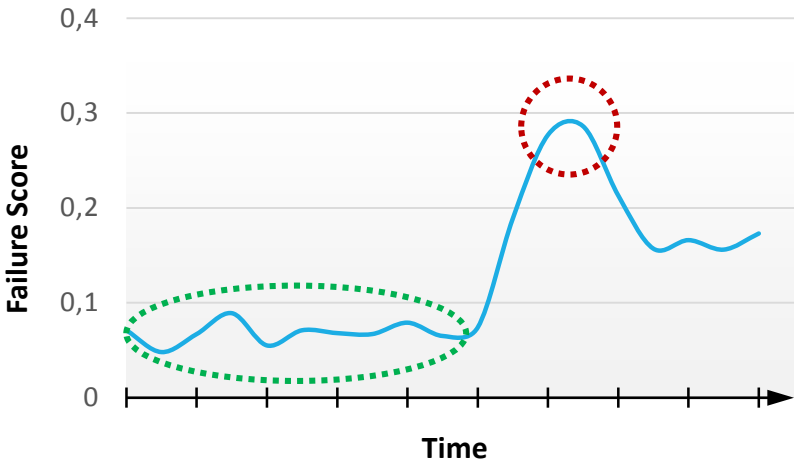
- Basic Counting
  - Estimate failure probability X_e for edge/vertex as the ratio of aggregated failed pings over aggregated total pings
  - $X_e = \frac{\sum_{i,j}[f_{i,j} \cdot P_{i,j}(e)]}{\sum_{i,j}[n_{i,j} \cdot P_{i,j}(e)]}$
- Probabilistic Coverage
  - Calculate the responsibility R of an edge/vertex set E for the observed failed pings
  - Find E^* that maximizes responsibility
  - $R(E) = \frac{\sum_{i,j}\left[\frac{f_{i,j}}{n_{i,j}} \cdot P_{i,j}(\mathrm{E})\right]}{\sum_{i,j}\left[\frac{f_{i,j}}{n_{i,j}}\right]}$

Other Approaches

- Cosine similarity
  - Calculate similarity S between vector of observed data and vector of estimated probability failure
  - $$S(E) = \frac{\sum_{i,j}\left[\frac{f_{i,j}}{n_{i,j}} \cdot P_{i,j}(\mathrm{E})\right]}{\sqrt{\sum_{i,j}\left[\left(\frac{f_{i,j}}{n_{i,j}}\right)^2\right]}\sqrt{\sum_{i,j}\left[\left(P_{i,j}(\mathrm{E})\right)^2\right]}}$$
- Maximum Likelihood Estimation (MLE)
  - Find best settings of failure probabilities $\{X_e\}$ of edges/vertices that maximize the likelihood of observations
  - $$\max_{\{X_e\}} \prod_{i,j}\left[\left(P_{i,j}(e)\cdot X_e\right)^{f_{i.j}}\left(1 - P_{i,j}(e)\cdot X_e\right)^{(n_{i,j}-f_{i.j})}\right]$$
  - Use gradient-descent algorithm to find (local) optimum

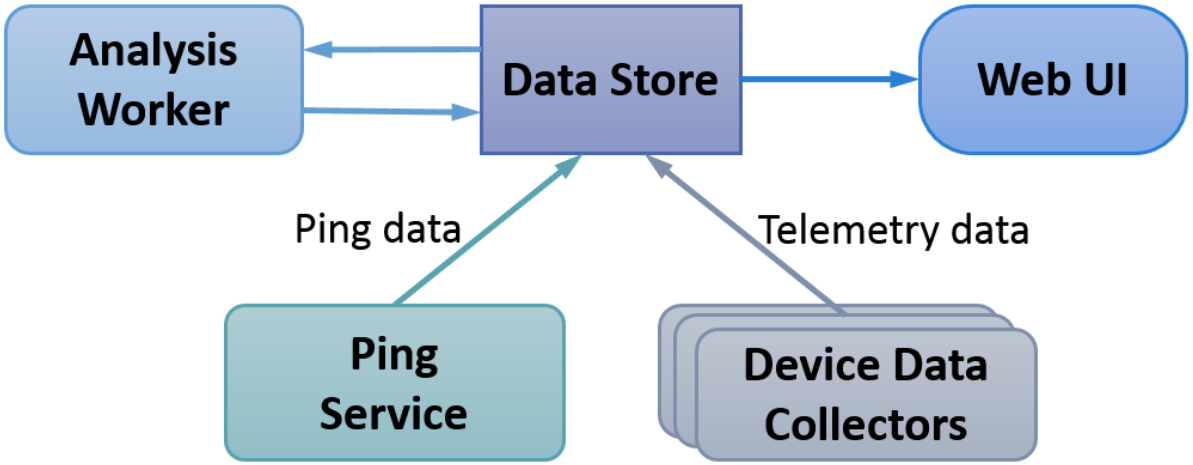
Localizing Network Failures (3)

- Non-zero failure scores for healthy edges/vertices
  - Pings may fail without network issues
  - Failed ping routes determined probabilistically
- Hard to distinguish from partial network failures
- Key idea: Look at failure scores over time
  - Establish a score "baseline" for each edge/vertex
  - Use hypothesis testing to detect statistically significant deviations from the norm



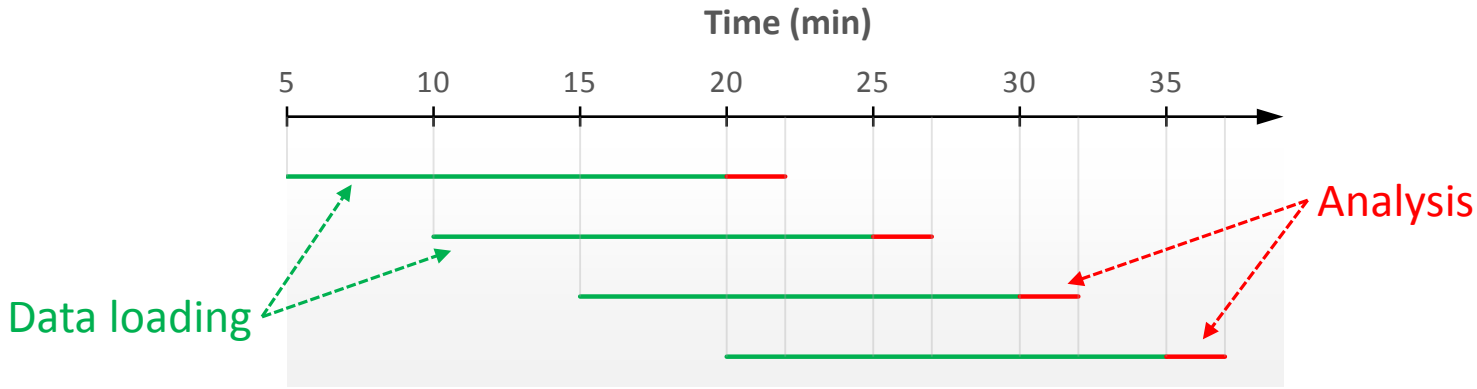
Deployment

- System Architecture



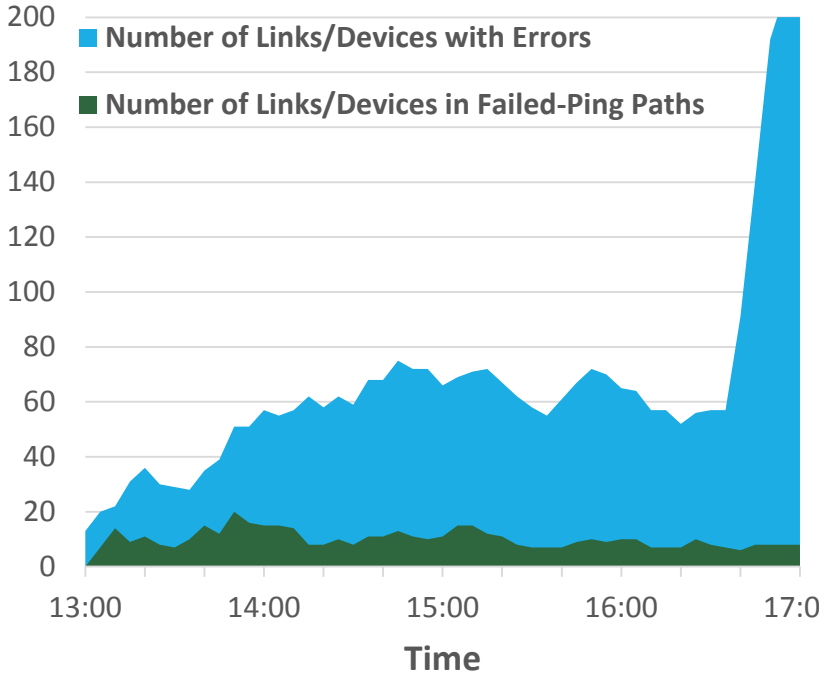
Real-Time Analysis

- Data sizes
  - ~200 clusters, ~10K devices, ~200K links
  - ~200K pings per 15 minutes
- Incremental processing
  - Window size: 15 mins, Step size: 5 mins
  - Generate ranked list of links/devices
  - Intersect with current error events & alerts



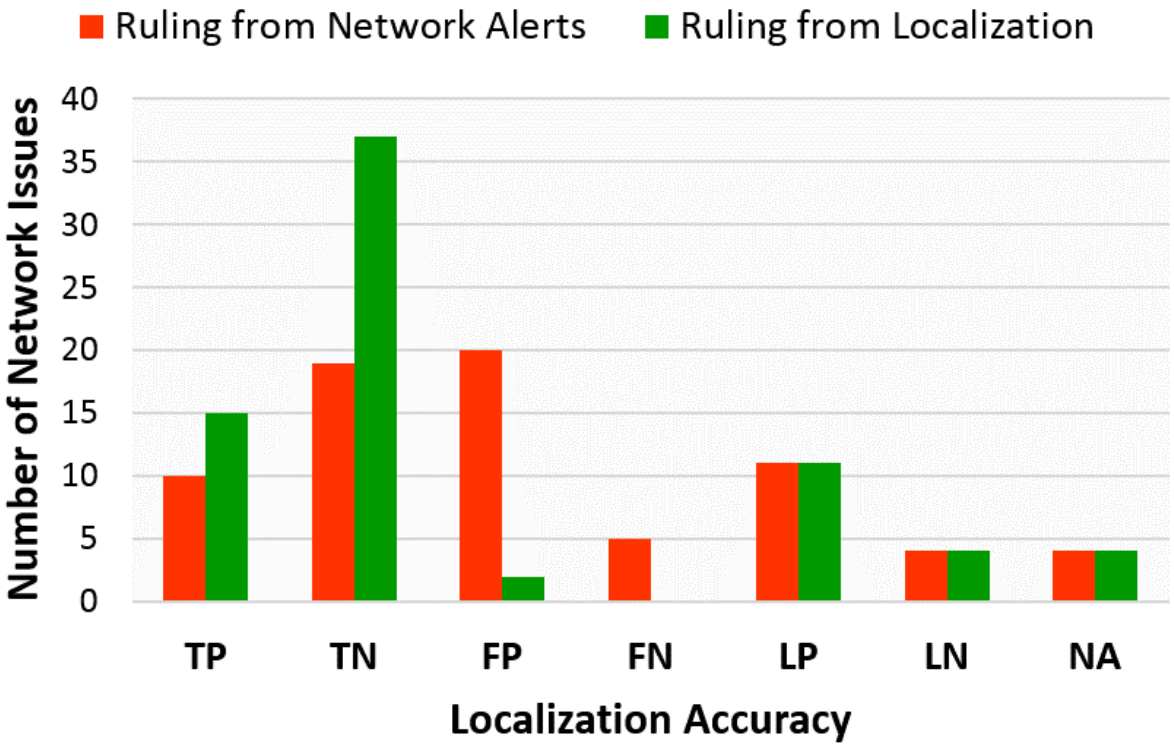
Validation (1)

- Issue: inside *some* cluster
- Duration: 14:20pm-17:00pm UTC on *someday*



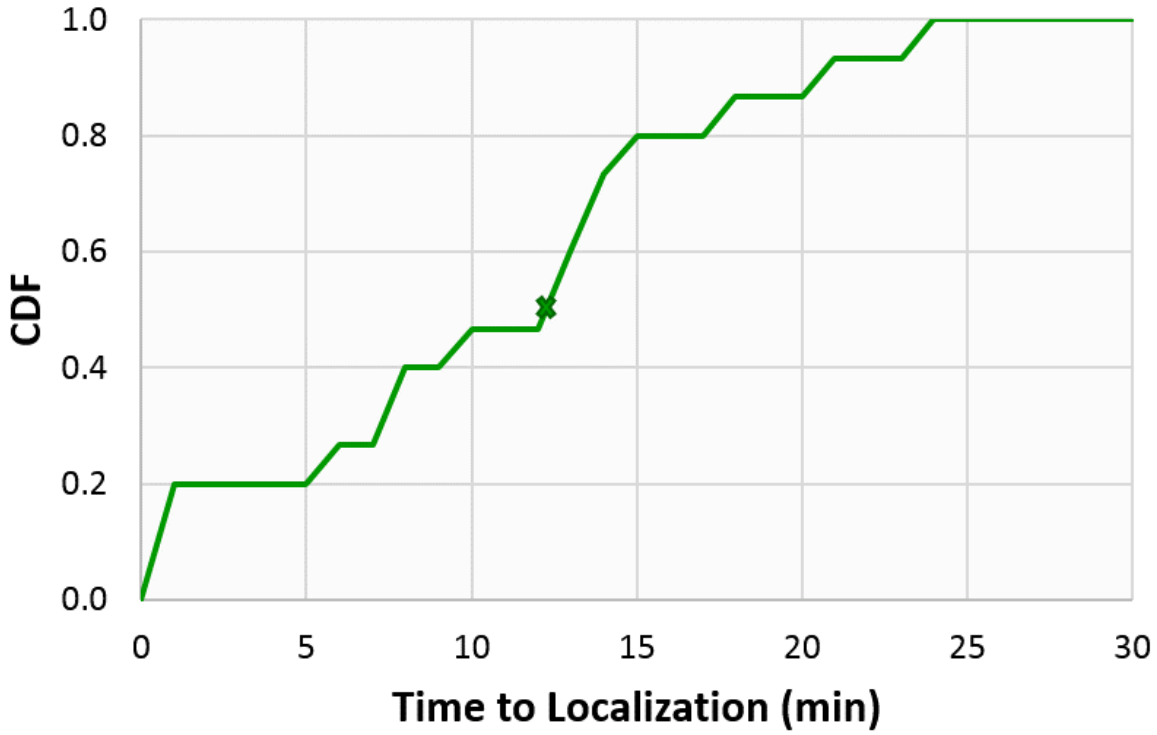
Validation (2)

- 73 real-life Issues



Validation (3)

- Monitoring efficiency



Future Work

- Use targeted active testing (e.g., trace route) to further localize failures
- Extend our analyses and algorithms to larger graphs (e.g., partner DCs/peering networks)

Questions