Experience. Outcomes.

Trace and Logs analysis

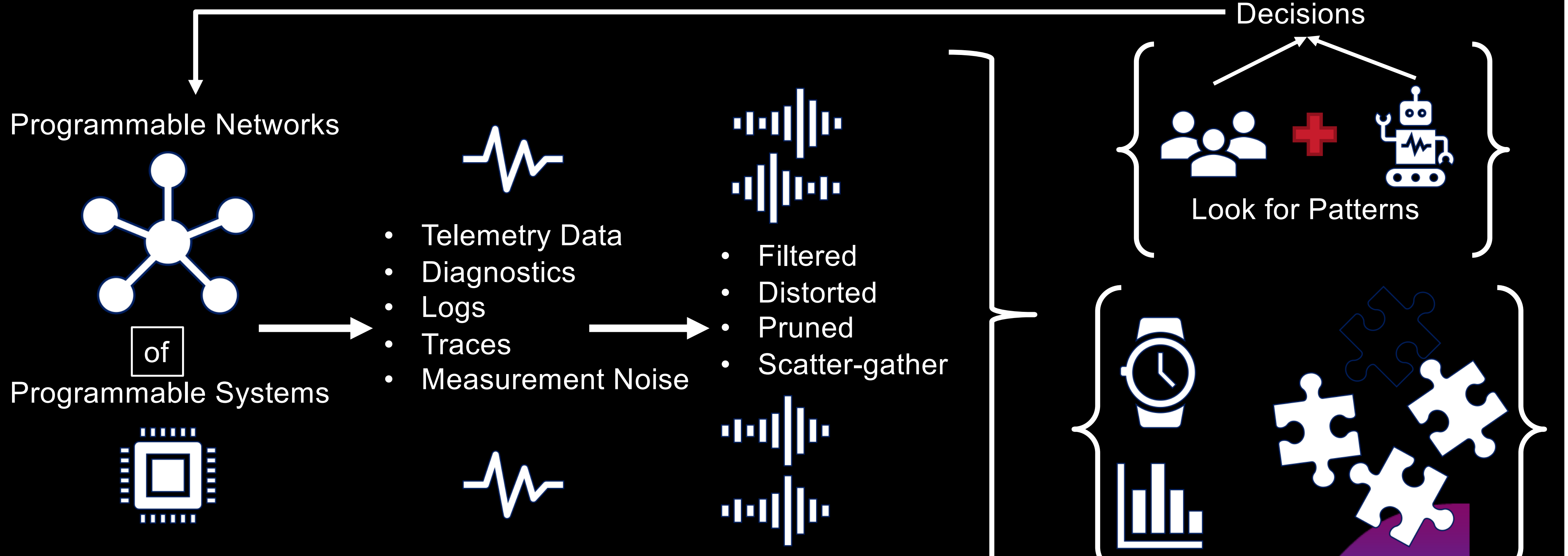
Elastic Search, Kibana, Plotly,…

Anurag Prakash, Cyrus Sadeghi, Alex Young and Peter Pieda

Dec 2018

Target Applications

- Network Time Correlated Debugging / Analysis
- Performance Characterization
- Always-on Tracing



Key Solution Requirements

Logging

- Easily added to large pre-existing on-box code base
- Conversion of existing logging to any new logging system must be largely an automated operation
- Off-box tools scalable to large networks with 100’s of Network Elements
- Simple and fast enough to be practical for debugging on very small development systems
- Support seek and filter

Performance analysis

- Does not require a special load build (resident in a disabled state in all loads)
- Minimal impact when enabled
- Bottleneck analysis
- Benchmark & Comparative analysis

Always-on Tracing

- Analysis/Playback of field issues without needing to reproduce them
- Several hours of trace buffer (time between issues occurrence and data collection)
- Strict constraints on performance and size
- always-on without sacrificing system performance
- Capability to render a message or state sequence from a running system

On-Box Infrastructure

Constraints
Allow filters and on-off capability
Minimal impact on running system
Performance and size
Schema driven
Fits into existing infrastructure

- events
- traces
- warnings
- info
- errors
- address
- flight-recorder

➢ Files/streams

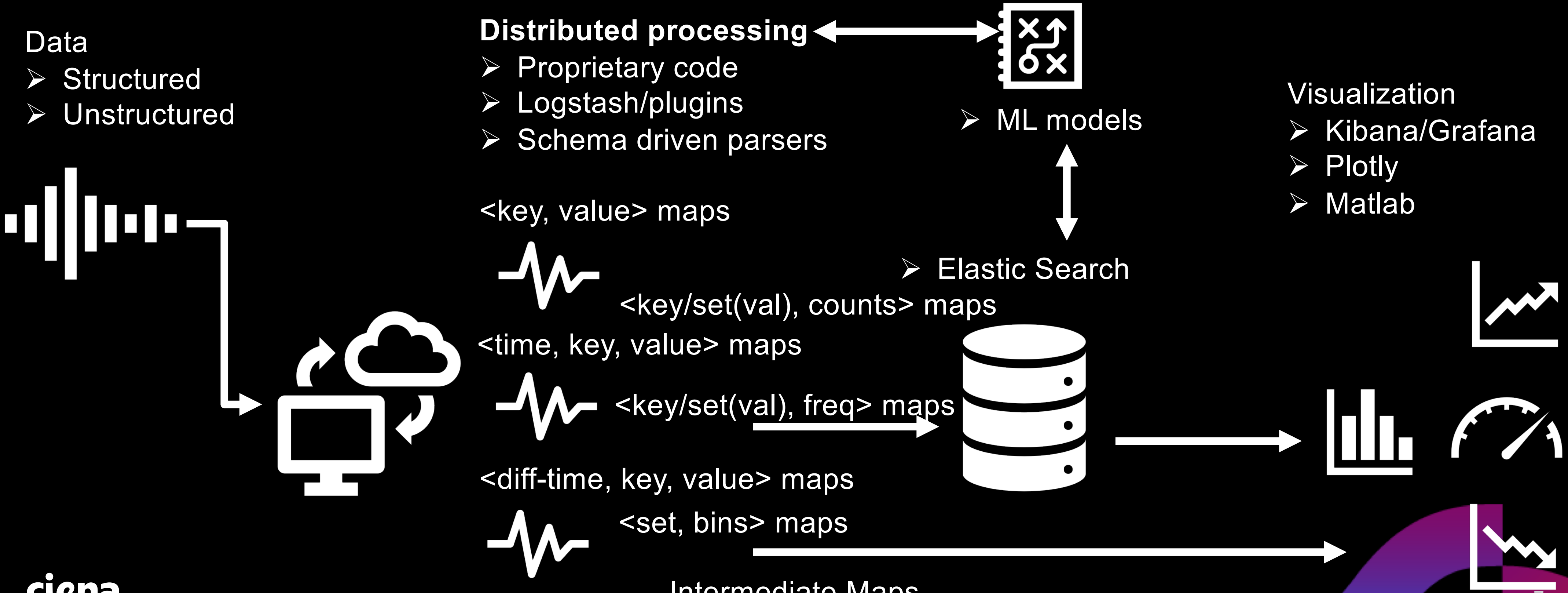
➢ Auth/Encryption

➢ JSON
➢ Protobuf
➢ Binary CTF
➢ Plain Text
➢ File upload

Data
➢ Structured
➢ Unstructured

Off-Box Infrastructure

Constraints
Programmable
Capability to scale to requirements
Responsiveness
Visualization



Data
- Structured
- Unstructured

Distributed processing
- Proprietary code
- Logstash/plugins
- Schema driven parsers

- ML models

- Elastic Search

Visualization
- Kibana/Grafana
- Plotly
- Matlab

<key, value> maps

<key/set(val), counts> maps

<time, key, value> maps

<key/set(val), freq> maps

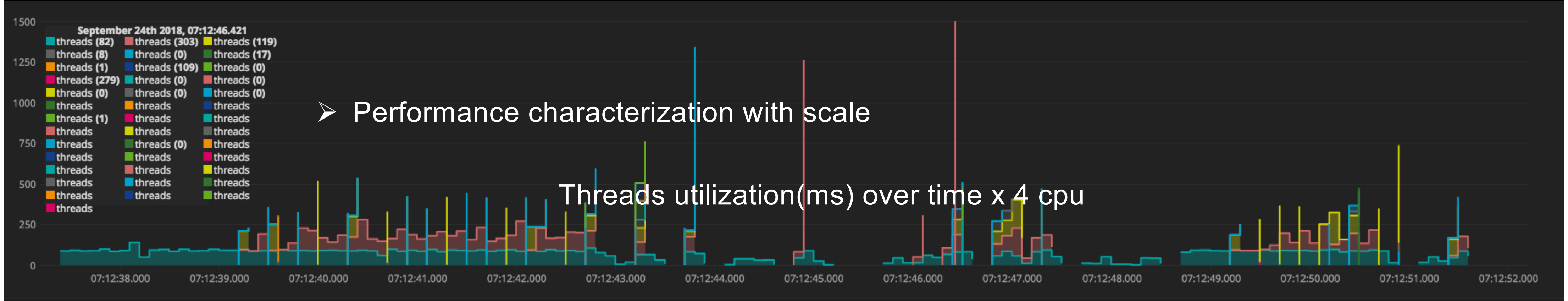
<diff-time, key, value> maps

<set, bins> maps

Intermediate Maps

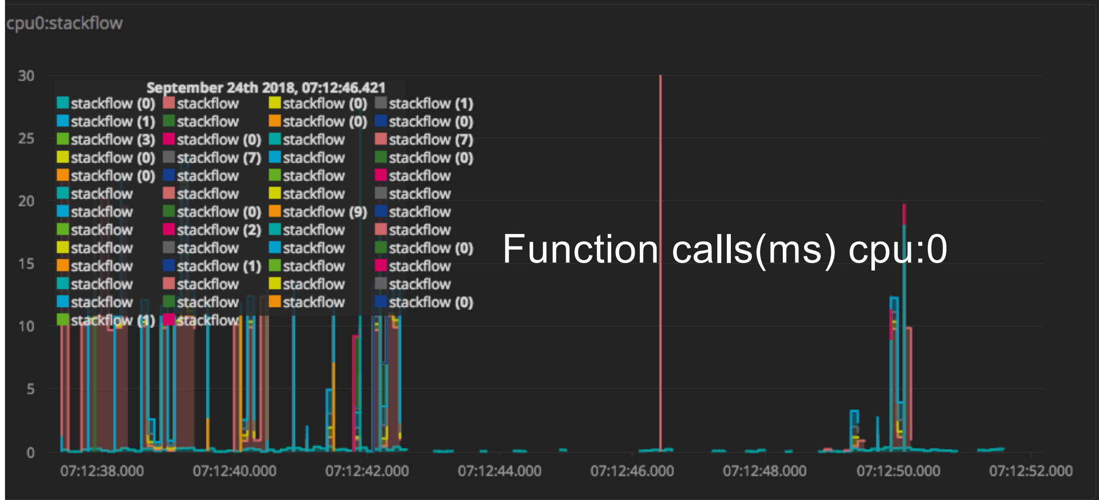
Trace data

X-axis: Time
Y-axis: utilization (ms)
Traces: threads or unique trace-backs

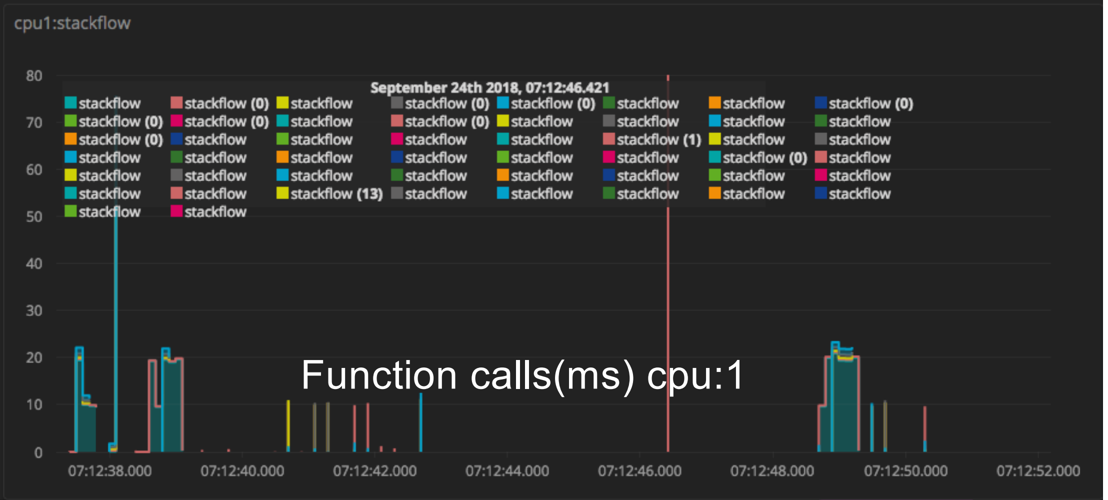


- ➢ Performance characterization with scale

Threads utilization(ms) over time x 4 cpu

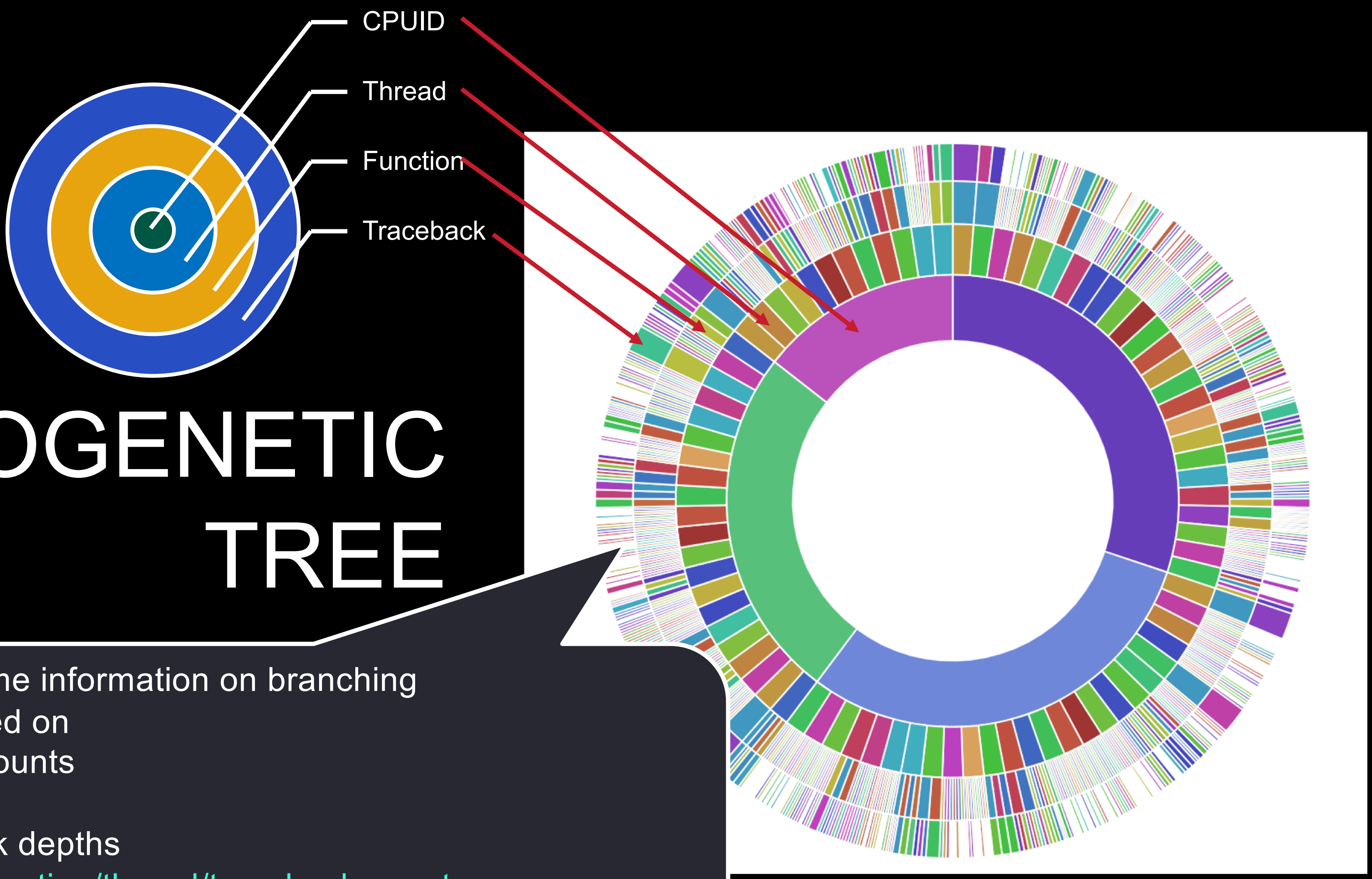


Function calls(ms) cpu:0



Function calls(ms) cpu:1

ciena



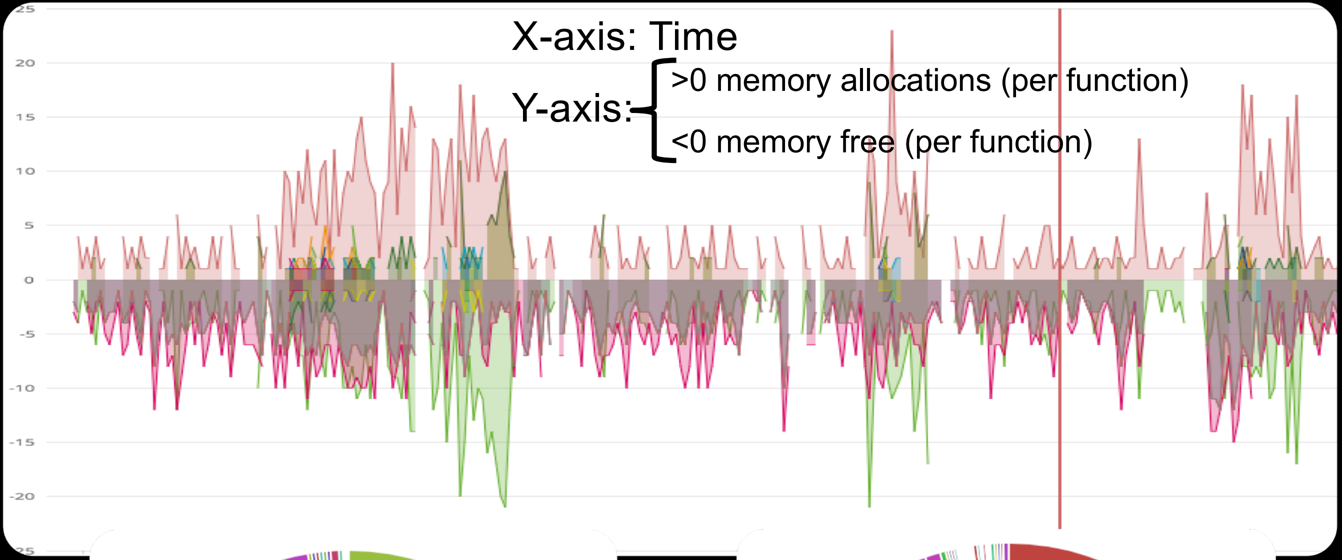
PHYLOGENETIC TREE

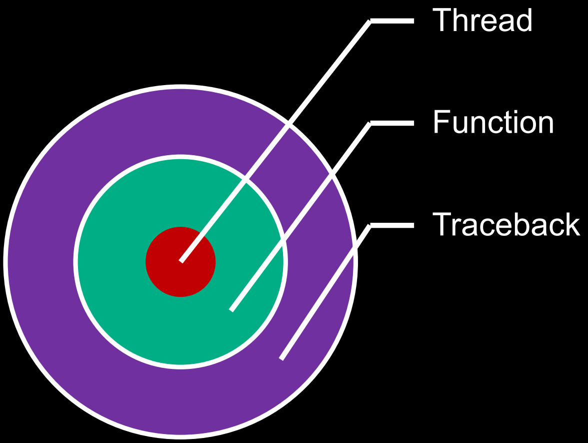
Provides run time information on branching

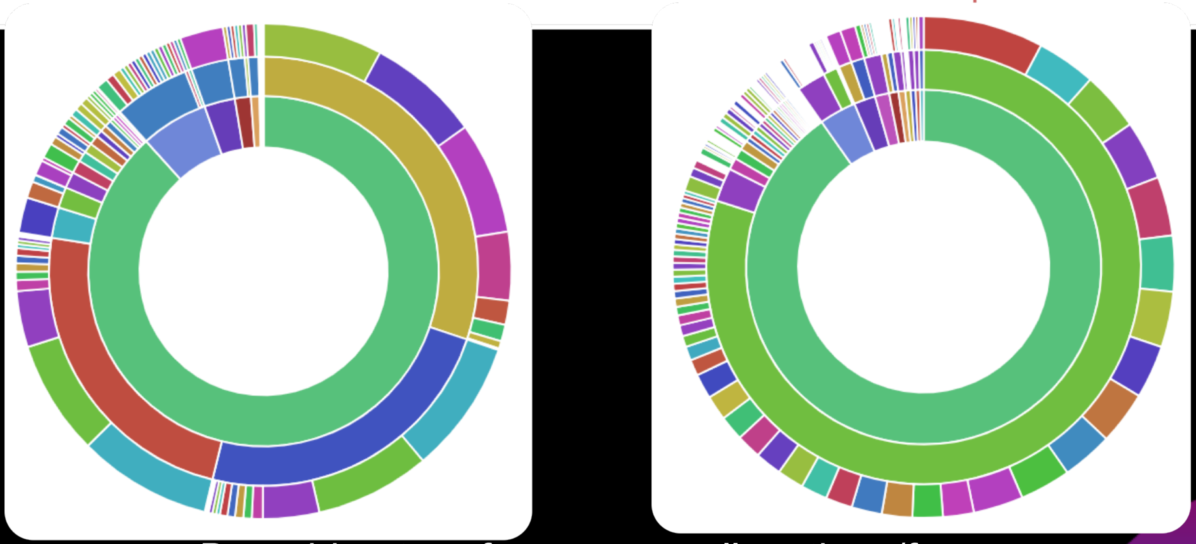
- Can be tailored on
    - ➢branch counts
    - ➢utilization
    - ➢traceback depths
    - ➢unique function/thread/traceback counts
    - ➢…

Trace data

Memory characterization

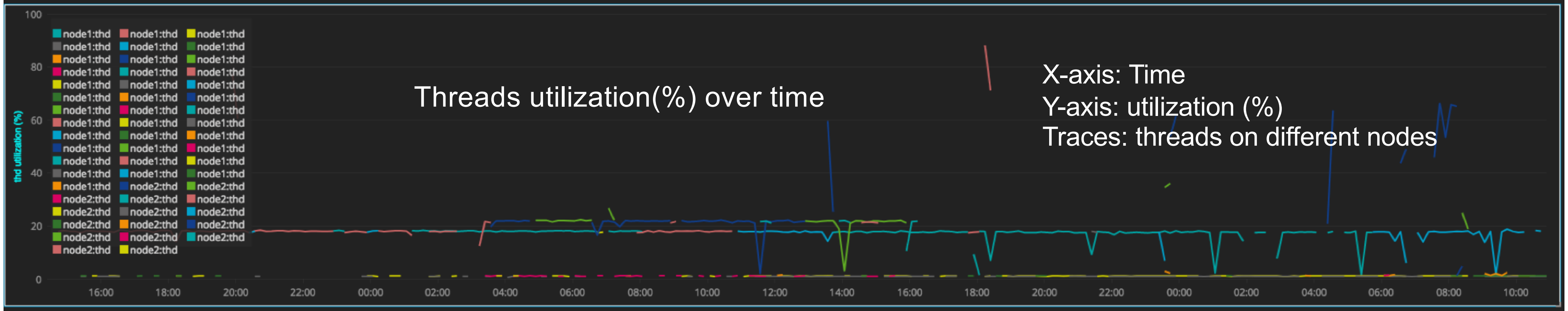






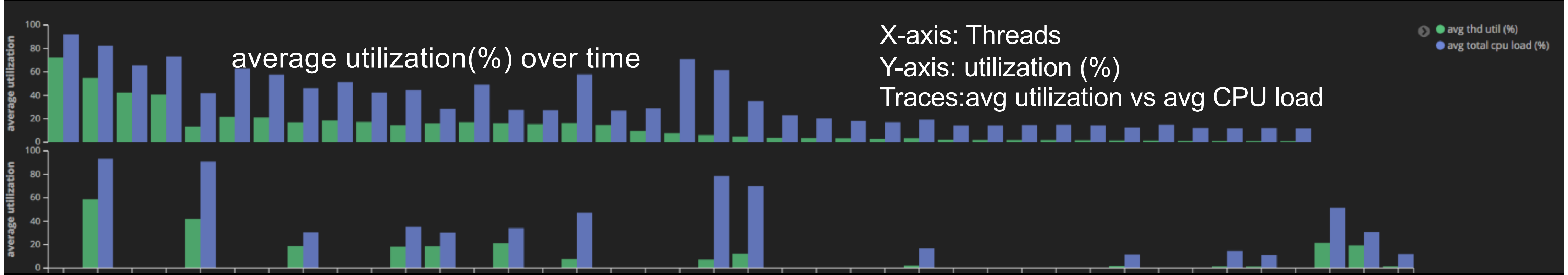
Branching tree for memory allocations/frees

Telemetry data



Threads utilization(%) over time

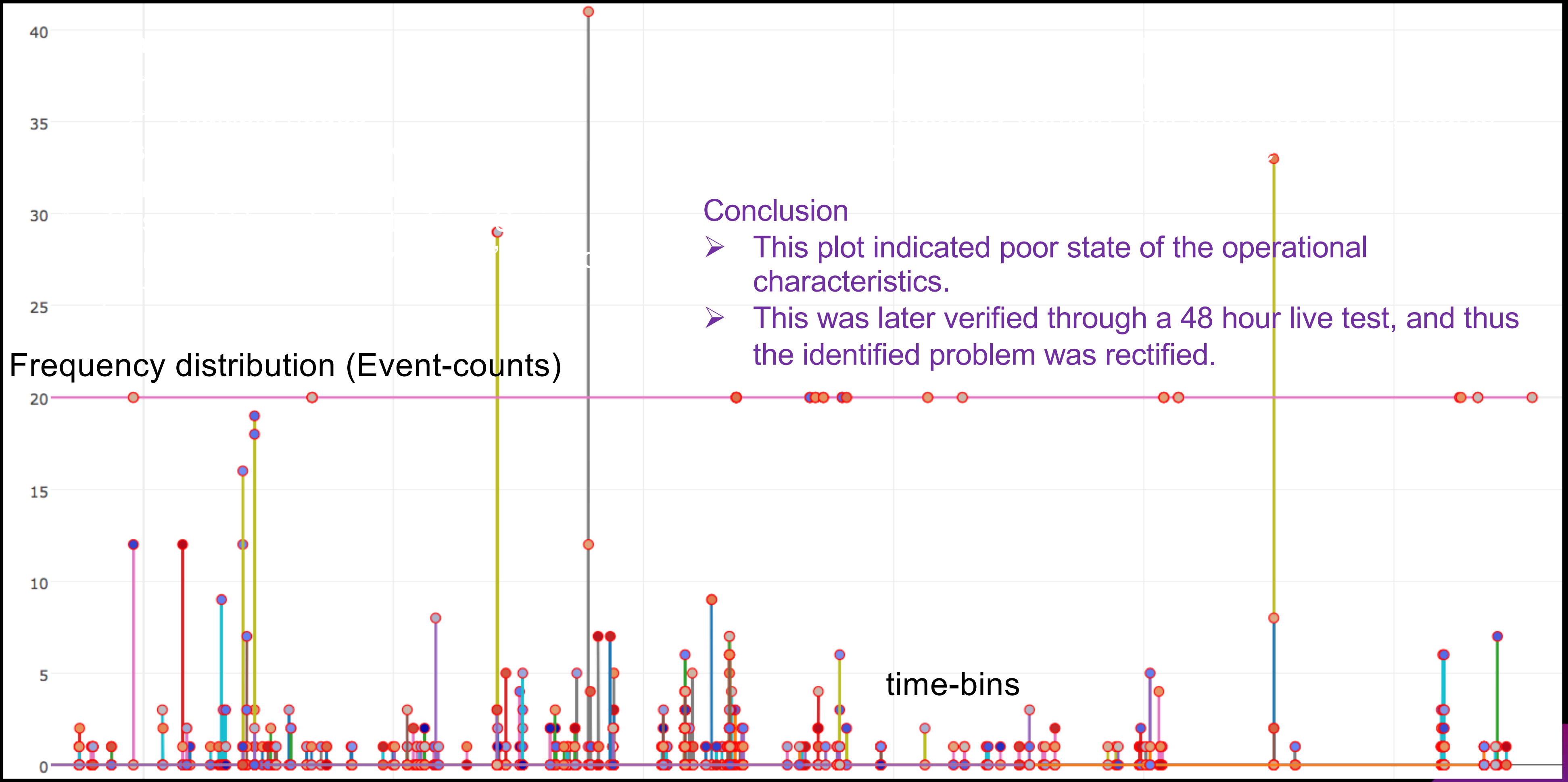
X-axis: Time
Y-axis: utilization (%)
Traces: threads on different nodes



average utilization(%) over time

X-axis: Threads
Y-axis: utilization (%)
Traces:avg utilization vs avg CPU load

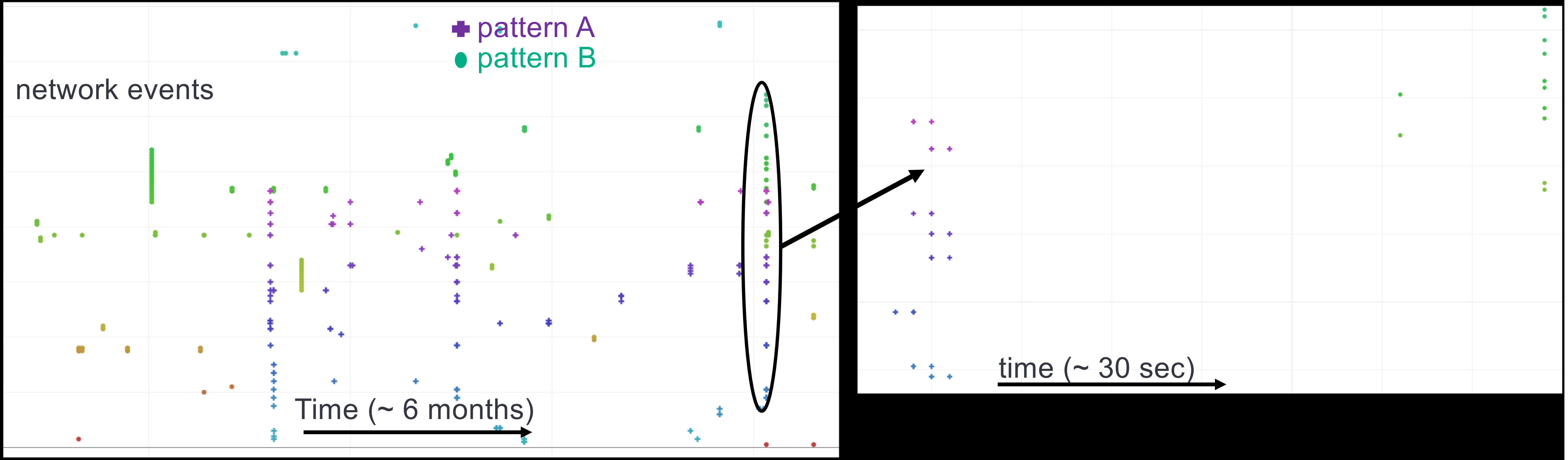
Unknown patterns but known time range



Conclusion

- ➢ This plot indicated poor state of the operational characteristics.
- ➢ This was later verified through a 48 hour live test, and thus the identified problem was rectified.

“g/re/p” for known patterns



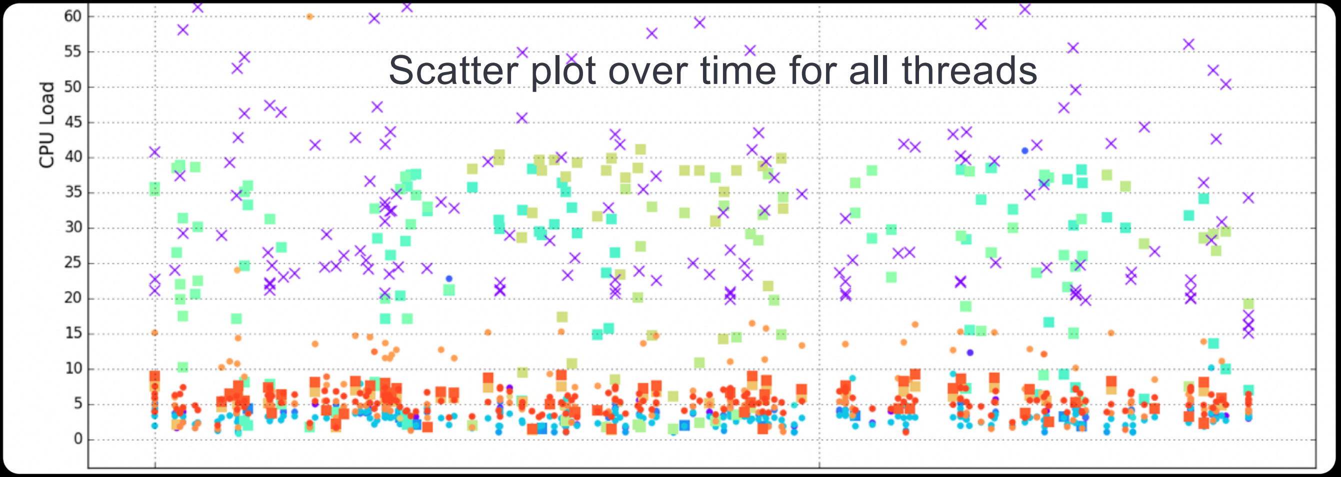
UNSTRUCTURED DATA

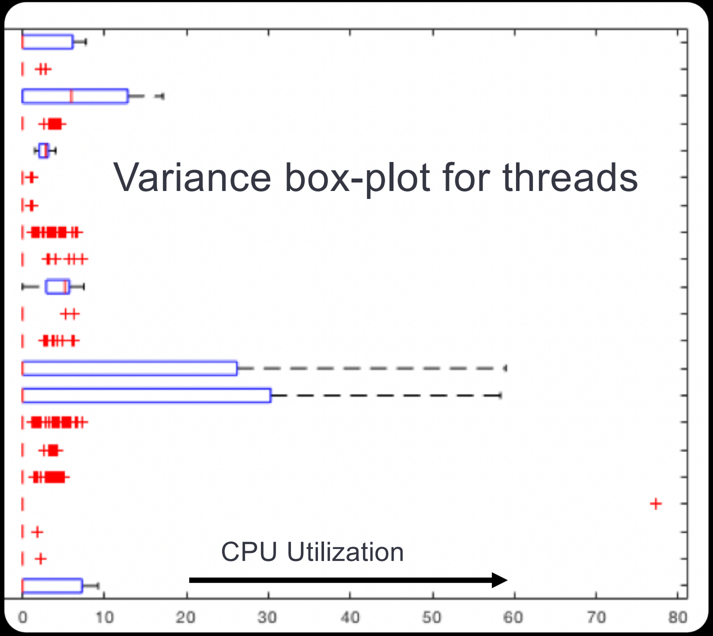
- pattern A and B are of interest
- plot all such patterns over data collected across the network for few months

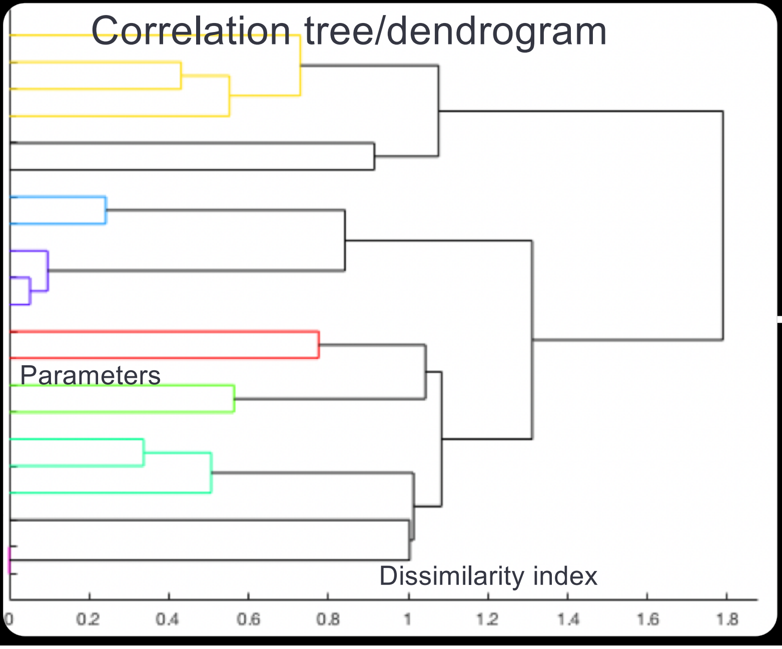
Conclusions

- Pattern A or B happen independent
- Pattern A and B when happen together with higher frequency of pattern A was the problem

Time series correlation



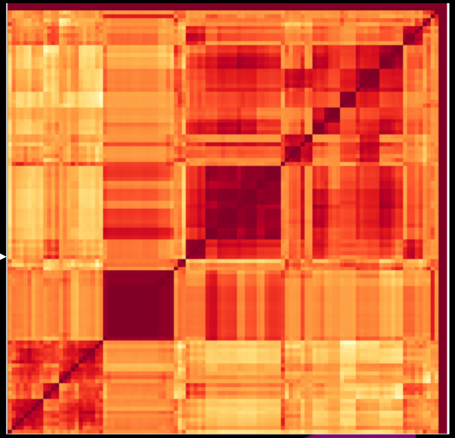




Distributed processing

- Time series data
- Sequential on-line data
- Find correlation patterns

- **Variance plot gives an idea of system activity**
- **Correlation plots provide relationships for pattern identification**



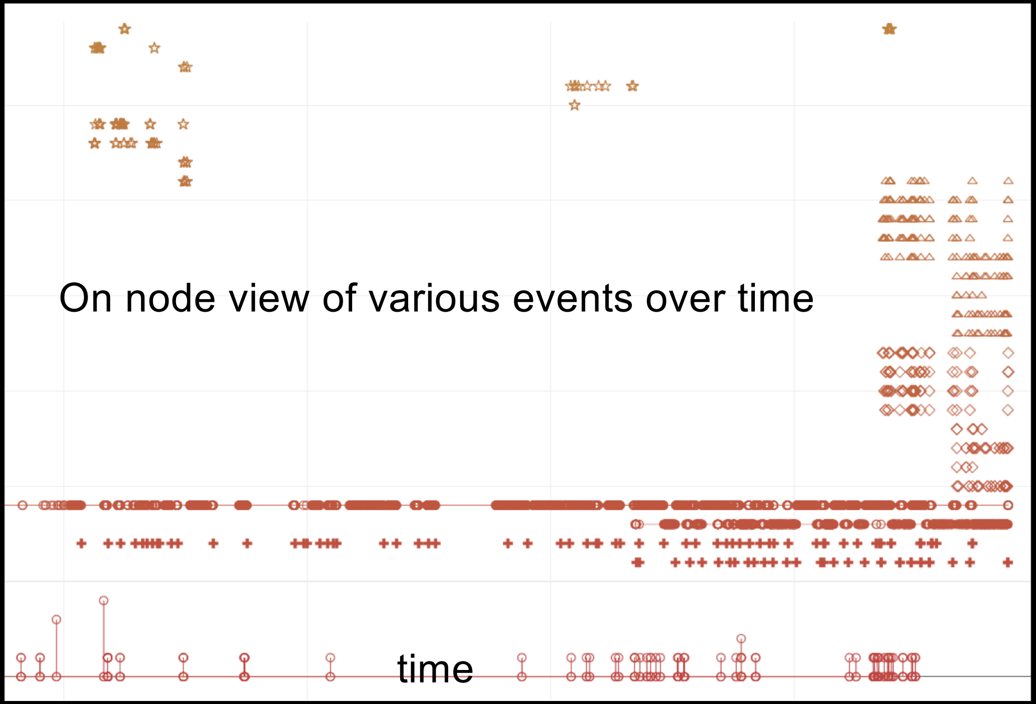
Heatmap (symmetric/asymmetric)

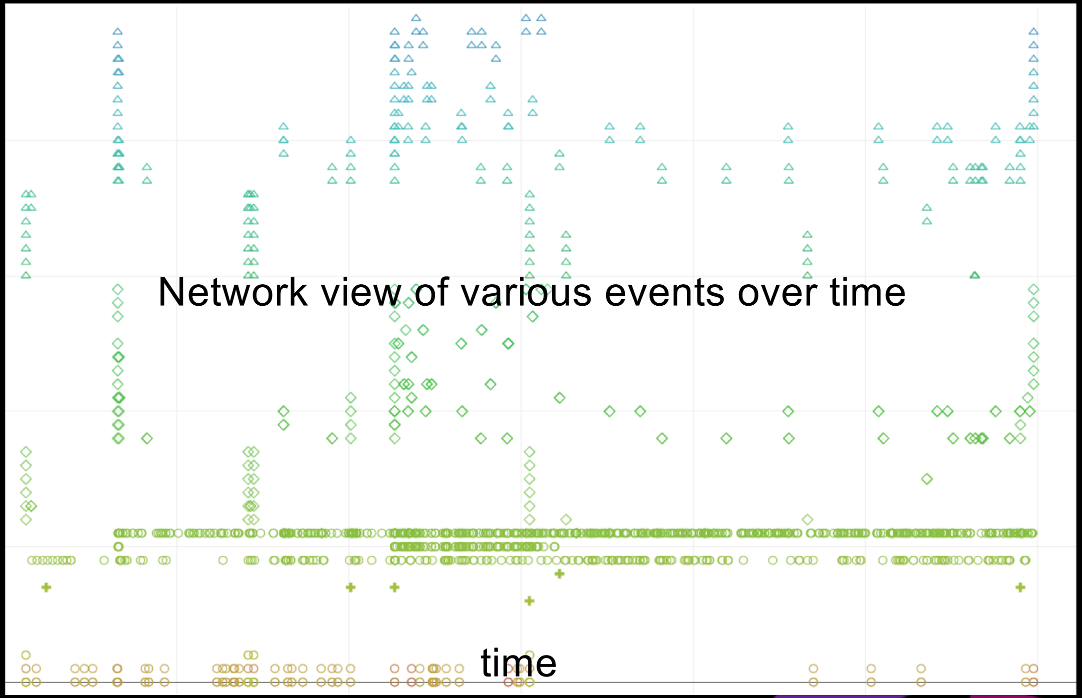
Unknown patterns (models)

Conclusion

- **The similarity amongst the two views indicated that the problem hops across systems, thus involved communication across elements**
- **Eventually lead to misconfiguration diagnosis**

- Problem had no known signature and was not specific to any known configuration
- Thus one approach was to seek comparisons between “nodal patterns vs network patterns”





Future

- Make existing infrastructure user friendly and faster for routine jobs
- Keep gathering and benchmarking the always-on tracing data
- Build and test new ML models on existing infrastructure

Questions ?

Thank You

ciena®

Experience. Outcomes.