DUST: Resource-Aware Telemetry Offloading with A Distributed Hardware-Agnostic Approach

Mehrnaz Sharifian∗‡, Diman Zad Tootaghaj⋆, Chen-Nee Chuah∗, Puneet Sharma⋆

⋆Hewlett Packard Labs, ∗UC Davis, ‡Hewlett Packard Enterprise (Aruba Networking)

IPDPS 2024 • San Francisco, California USA
San Francisco, California • May 27-31, 2024

Outline

1. Overview and Motivation
2. Proposed Solution System Architecture
3. DUST System Nodes and Workflows
4. Optimized Network Monitoring Placement
5. Performance Evaluations
6. Conclusion And Future Work

UCDAVIS COLLEGE OF ENGINEERING

Hewlett Packard Enterprise

Outline

1. **Overview and Motivation**
2. Proposed Solution System Architecture
3. DUST System Nodes and Workflows
4. Optimized Network Monitoring Placement
5. Performance Evaluations
6. Conclusion And Future Work

UCDAVIS COLLEGE OF ENGINEERING

Hewlett Packard Enterprise

Overview & Motivation

Network Monitoring:

A critical part of network deployment, conventionally deploying in a central approach e.g., SNMP, sFlow, etc.

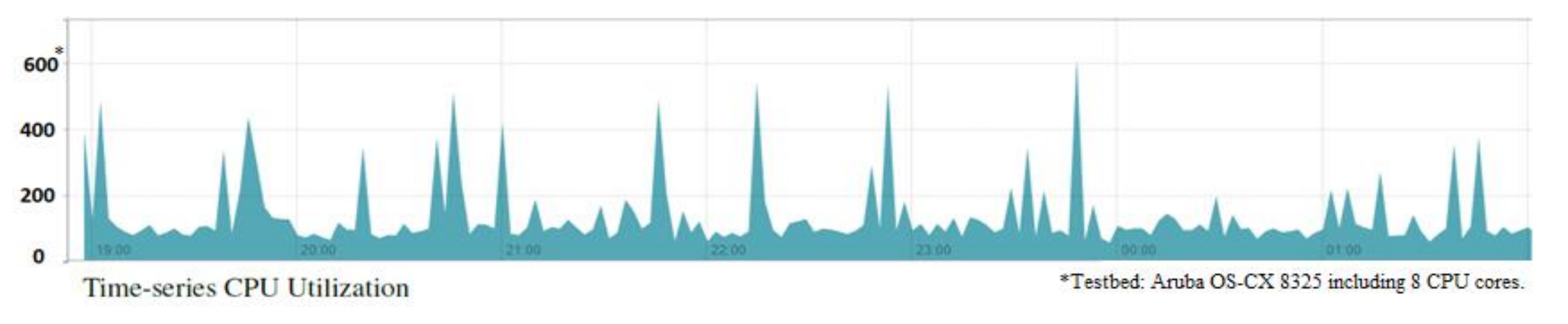
<u>In-device Network Telemetry</u>:

- Leverage *device computing* to empower *distributed*
- Gaining insights into the devices' status and behavior
- Precise collection ➔ Eliminates excessive data traverse
- Usage-based application, fine-grained data collection
- Real-time data collection, processing, and action

- Computational and storage heavy;
- Device-level functionality and scale;
- Resource efficiency and accuracy dilemma;
- Lack of central configuration & management
- Unsupported on low-profile switches;

Problem Motivation

Illustration of the extent of excessive resource usage of in-device telemetry:



- CPU consumption of a modular in-device analytic engine on a database-driven network OS enterprise switch.
- User-defined modular in-device monitoring agents; e.g., network protocol health, Rx/TX packet rates, resource utilization.

Outline

1. Overview and Motivation
2. **Proposed Solution System Architecture**
3. DUST System Nodes and Workflows
4. Optimized Network Monitoring Placement
5. Performance Evaluations
6. Conclusion And Future Work

Opportunities and Solution:

There are **ample computing resources** available in high-performance and industry-leading devices as opposed to certain nodes facing resource constraints while running in-device network monitoring,

➔ **Discarding these resources is not cost-effective for users.**

A study of NERSC's Perlmutter (HPC system), revealed that a **quarter of CPU node-hours achieved high utilization**, while **GPU-accelerated nodes were utilized for only 0-5% of the node-hours**.

➔Typical issue in HPC systems with primarily assigned resources

With the **emergence of DPUs and smart Fabrics/ASICs**, diverse computational capabilities in **servers and network switches are available** in data centers and High-Performance Computing (HPC)

- **DUST relies on utilizing the abundant but often underutilized computing resources in top-performing devices across the network.**

Proposed Solution System Architecture

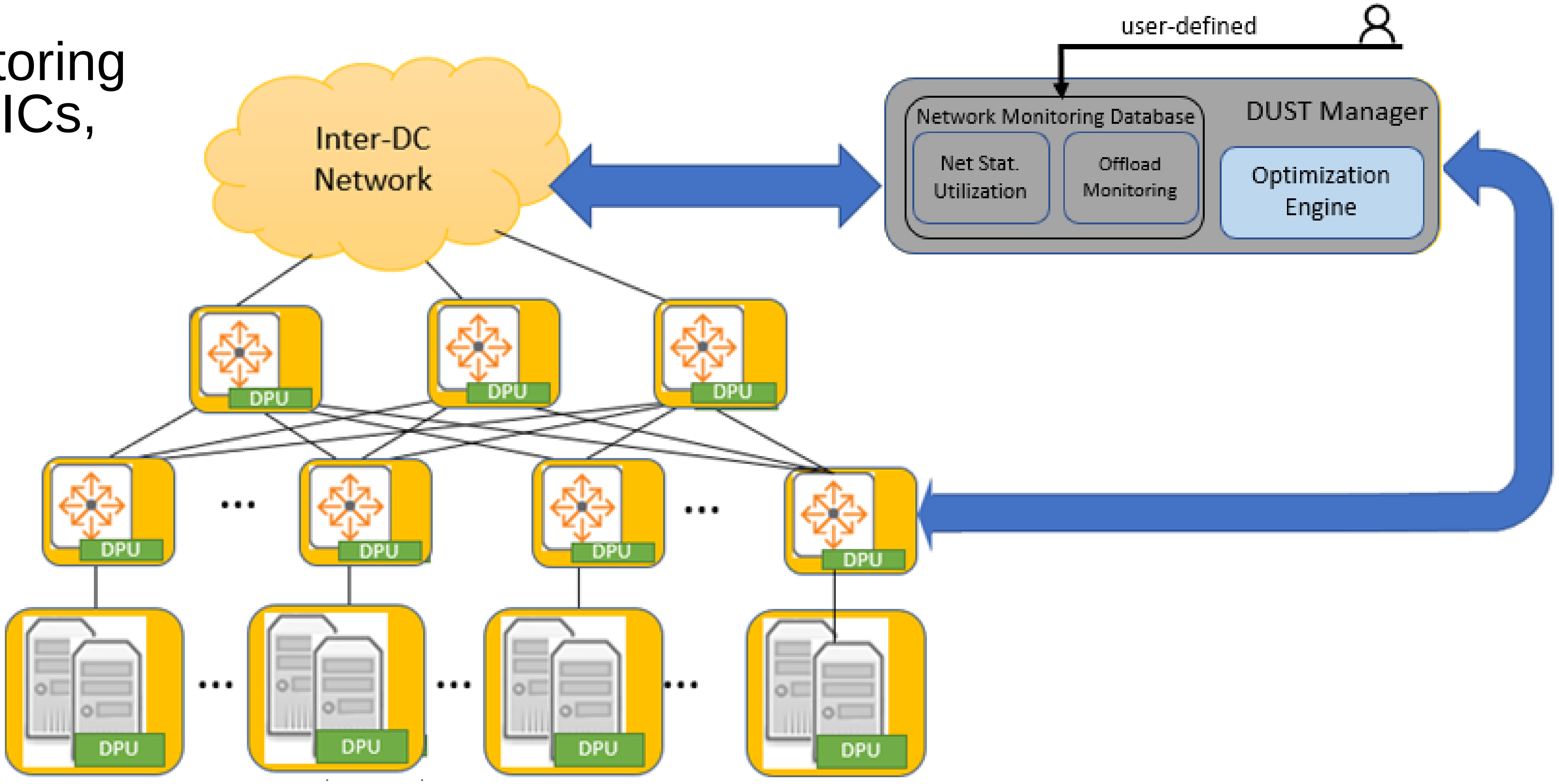
DUST: Dynamic and Distributed Usage-based Service Telemetry

Novel distributed usage-based monitoring service, can be deployed on SmartNICs, DPUs, Switches, or Servers.

Goal: Distribute and monitoring placement by leveraging available compute resources across network' nodes.

DUST two main parts:

- DUST –Client
- DUST –Manager



Outline

1. Overview and Motivation
2. Proposed Solution System Architecture
3. **DUST System Nodes and Workflows**
4. Optimized Network Monitoring Placement
5. Performance Evaluations
6. Conclusion And Future Work

Network Monitoring Placement Problem

Goals:
- Distribute the network monitoring tasks
- Optimal remote node selection for offloading

Constraints:
- Minimize monitoring data movement & response time

Solution:
- System-level architecture for workflow redistribution and packet flow communication
- Optimal remote node selection algorithm (ILP and heuristic)
- Controllable routing decisions

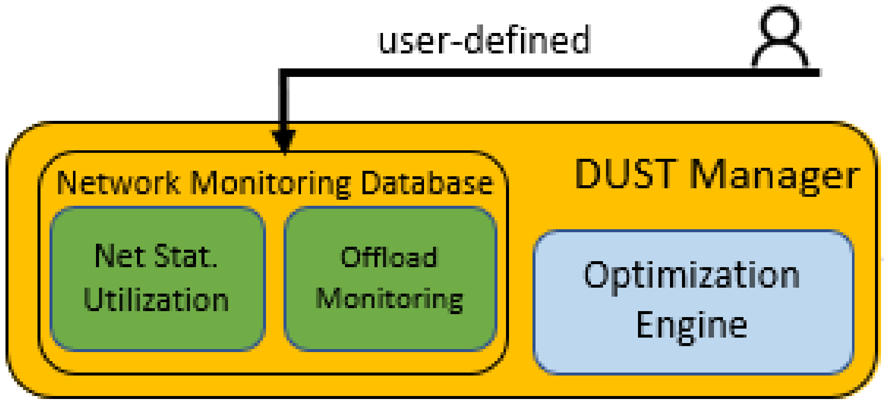
Network Monitoring Placement Overview

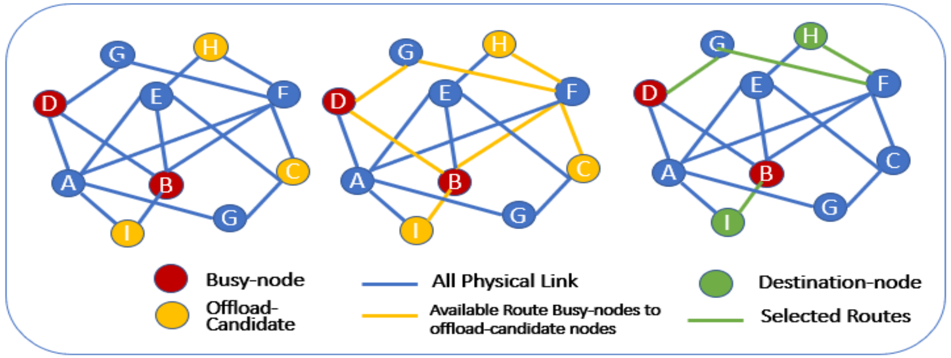
Based on **resource util** and **network stats**, DUST-Client is defined as:

- ’**Busy-nodes**’,
- ’**Offload-Candidate** nodes’,
- **’Offload destination**’ node,
- ’**none-offloading**’ nodes.

Goal: Proposed **controllable route algorithm**.

1. Leverage data sourced from DUST Database to facilitate DUST-Manager in identifying nodes.
2. Optimization engine evaluates all routes between each busy to destination node ➔ selects the minimum cost destination nodes



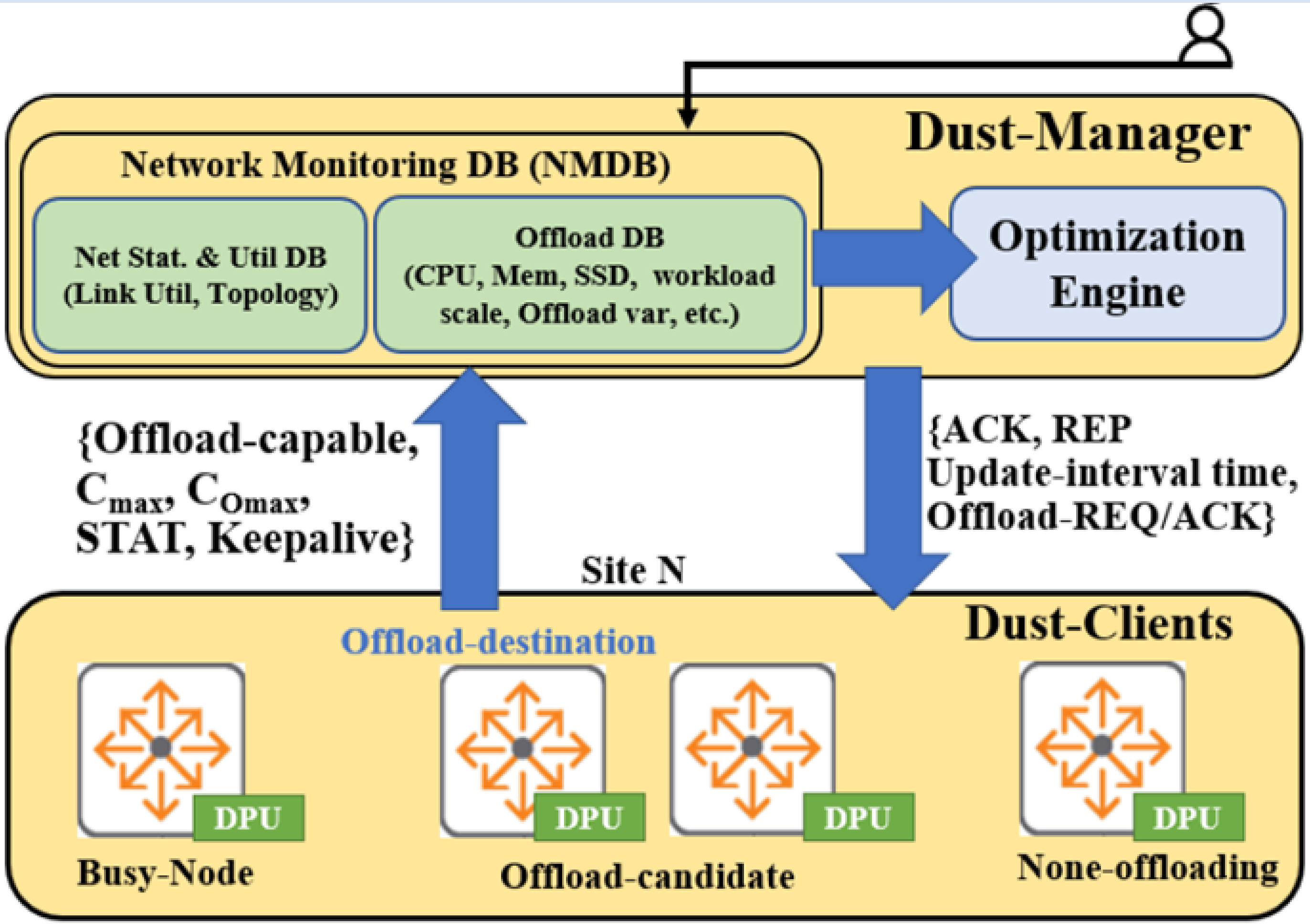


DUST System Nodes and Workflows

Post-offloading Process in DUST:

Following a successful offloading, designed to uphold system performance while simultaneously monitoring the health status of the offloaded workloads

- **QoS Guarantees**
- **Offload-destination Node Status**



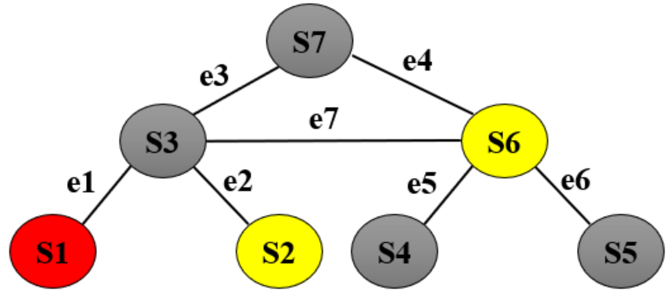
Outline

1. Overview and Motivation
2. Proposed Solution System Architecture
3. DUST System Nodes and Workflows
4. **Optimized Network Monitoring Placement**
5. Performance Evaluations
6. Conclusion And Future Work

UC DAVIS COLLEGE OF ENGINEERING

Hewlett Packard Enterprise

Optimization Algorithm: ILP Formulation



Variables: Max utilized capacity of a node being Busy-node ; Capacity to be considered as offload-candidate node below

$$Cmax\ (\%) \quad , C_b \geq C_{max}$$
$$Comax\ (\%) \quad , C_o \leq CO_{max}$$

Variables: Minimum Response time between node i to j, Monitoring data Di, Link utilization Luij

$$Tr_{i,j} = \frac{D_i(\text{Mb})}{Lu_{i,j}(\text{Mbps})}$$
$$Tr_{i,j}(r_k) = \sum_{\forall e \in r_k} \left(\frac{D_i}{Lu_e}(sec)\right) \quad \forall i \in V_b, \forall j \in V_o, \quad \forall r_k \in p$$

Objective: objective function β of minimizing response time for every selected decision variable

$$\text{Minimize} \quad \beta = \sum_{i \in V_b} \sum_{j \in V_o} x_{ij} * T_{rmin,i,j}$$

Capacity to offload from every busy node ∀i ∈ Vb
Available Capacity of every offload Candidate node ∀j ∈ Vo

$$Cs_i = C_i - C_{max} \quad \forall i \in V_b$$
$$Cd_j = CO_{max} - C_j \quad \forall j \in V_o$$

Constraint for ∀i ∈ Vb to fully offload the overloaded all extra capacity after Cmax (%)

$$\sum_{j \in V_o} x_{ij} = Cs_i \quad \forall i \in V_b$$

Constraint for ∀j ∈ Vo to not overload the selected destination node after monitoring placement

$$\sum_{i \in V_b} x_{ij} \leq Cd_j \quad \forall j \in V_o$$

Heuristic Algorithms

- **Heuristic Model:** Offload-destination node being selected merely from any available next-hop node of Busy-node, if exists.

- **Goal**: Solving the minimum cost offload problem in large-scale networks in a time-efficient manner by reducing the computational complexity.

- **Heuristic Failure Rate:** defined as amount of monitoring data failed to be offloaded with heuristic node selection to the total of resources required to be offloaded at any given network state.

$$HFR(\%) = \frac{\sum_{i \in V_{hb}} Cse_i}{\sum_{i \in V_{hb}} Cs_i} * 100$$

Algorithm 1: Heuristic Algorithm of min-cost problem

```
for every node k ∈ V of graph G = (V, E) and node capacity C_k do
    Determine set of Busy nodes, V_hb = {V_hb0, V_hb1, ..} if C_k ≥ C_max
    for every Busy node i ∈ V_hb do
        Determine set of Offload-candidate nodes,
          V_ho = {V_ho0, V_ho1, ..}, including every node j if C_j
          ≤ CO_max and within shortest path of max-hop=1 to a
          given Busy node V_hbi
        Calculate Cs_i = C_i - C_max
        for every Offload-candidate node j in set V_ho , calculate
          Cd_j = CO_max − C_j do
            Define continuous optimization decision variable X_ij
              for every given Busy node i and within its
              Offload-candidate nodes set V_ho do
                Minimize β (Equation 3) for defined heuristic set
                  of nodes and X_ij
```

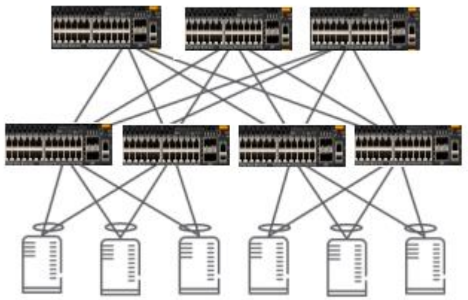
Outline

1. Overview and Motivation
2. Proposed Solution System Architecture
3. DUST System Nodes and Workflows
4. Network Monitoring Placement
5. **Performance Evaluations**
6. Conclusion And Future Work

UCDAVIS COLLEGE OF ENGINEERING

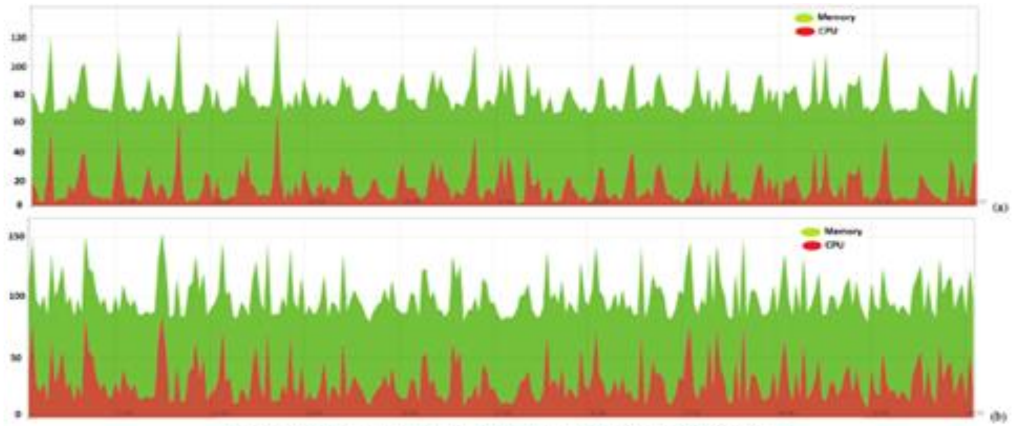
Hewlett Packard Enterprise

Performance Evaluations



Testbed:

- Real prototype with a VxLAN data center spine/leaf topology and high-performance ToR commercial enterprise switches, installed 10 monitoring agents.



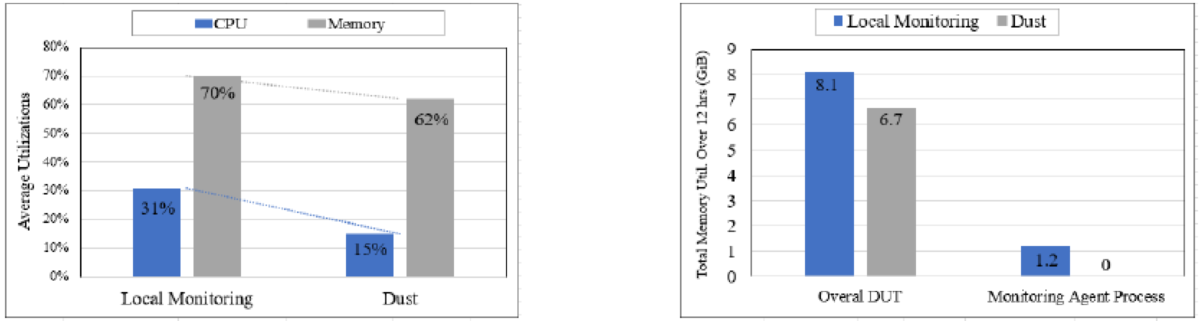
Average resource's Utilization (a) Dust (b) Local Monitoring

Experimental Parameters:

- Performance comparison and resource utilization between DUST (where user-defined monitoring agents were offloaded) and local monitoring.

Results:

- By average, 52% and 12% of CPU and memory saving by offloading, respectively.
- indicates that monitoring workloads are perfect offloading candidates for network switches



(a) Avg. resource utilization.

(b) Total resource utilization.

Memory and CPU resource utilization comparison in DUST and local monitoring.

UC DAVIS COLLEGE OF ENGINEERING

Hewlett Packard Enterprise

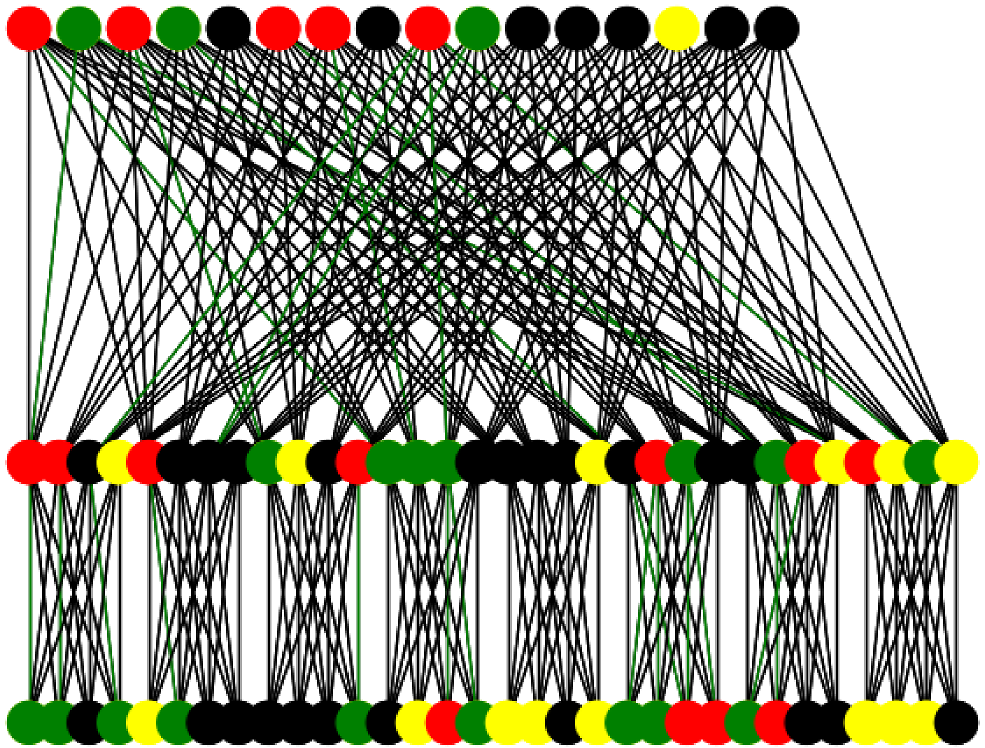
Fat-tree Topology Monitoring Placement

Testbed: A simulator implemented by using the Gurobi optimization toolkit - model **fat-tree** topology.

Goal: Addressing the scalability of our optimization algorithms in large-scale networks.

- **Busy node (red)**
- **Offload candidate (yellow)**
- **Destination node (green)**
- **Controllable data route (green edges)**

Flexible Offloading is supported**.**



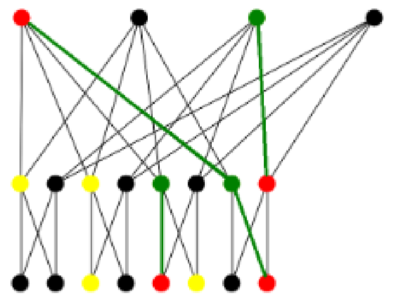
DUST Scalability Analysis

k-port	Nodes	Edges
4	20	32
8	80	256
16	320	2048
64	5120	131072

- *Small-Scale Network Evaluation*
  - Defined as a network, data center pod, or network zone comprising 20 or fewer network nodes (equivalently represented as a three-level 4-k port fat-tree topology with 20 nodes and 32 edges.)

- *Large-Scale Network Evaluation*
  - Defined as a network or data center architecture with more than 20 network nodes (equivalently represented as a three-level 8-k, 16-k, and 64-k ports fat-tree topology with 80, 320, and 5120 nodes along with 256, 2048, and 131072 edges, respectively.)

Small-Scale Network Evaluation

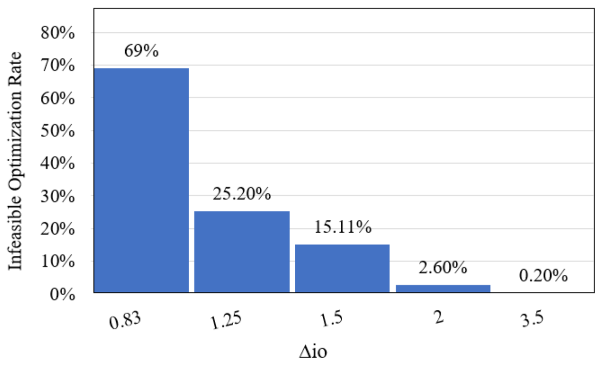


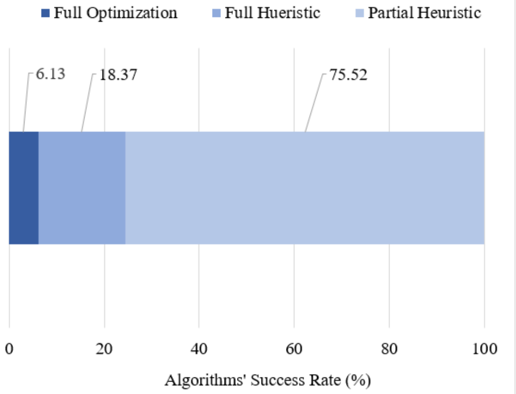
Testbed: Fat-tree 4-K three-level data center over 1000 iterations.

- Benchmark for selecting optimal user-defined values of Cmax,COmax to improve the optimization efficiency

  ➔ reduce the impossible optimization rate (%).

$$\Delta_{io} = \frac{\Delta_o}{\Delta_b} = \frac{CO_{max} - x_{min}}{100 - C_{max}} \quad , \quad \Delta_{io} \geq K_{io}$$

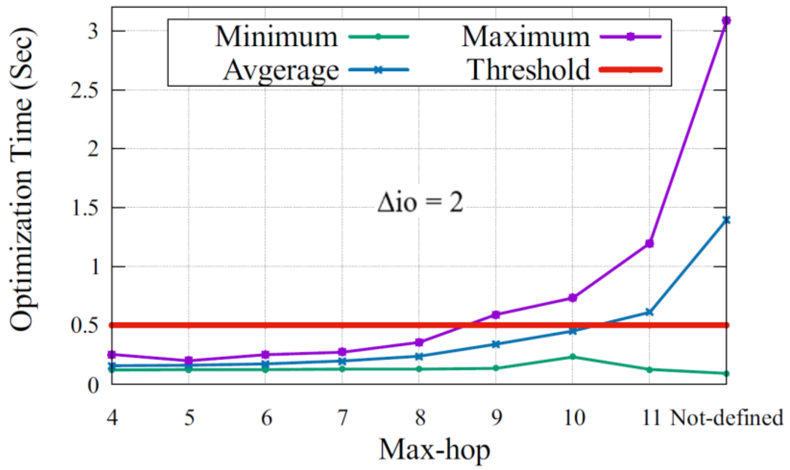




Results:

- Maximum optimization time remains below 3.5 sec

➔ Suitable for time-constrained applications.



Large-Scale Network Evaluation

Testbed: Fat-tree 8-K, 16-K three-level data center

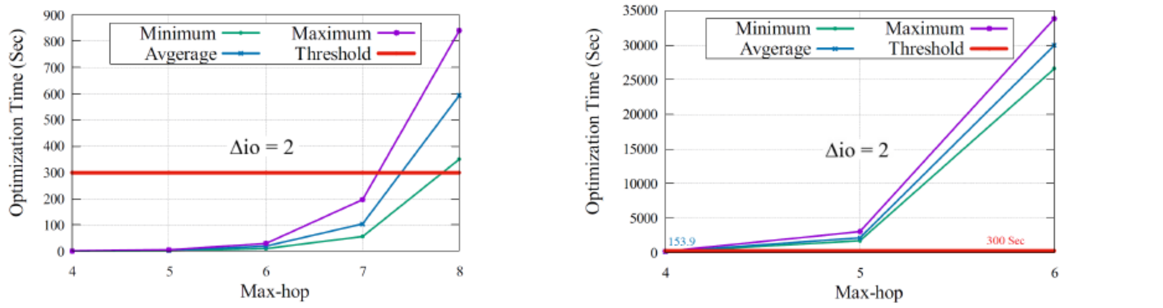
Results:

- max-hop with a threshold response time.
- Number of hops is significantly decisive and cost-effective.

As the network scale grows:

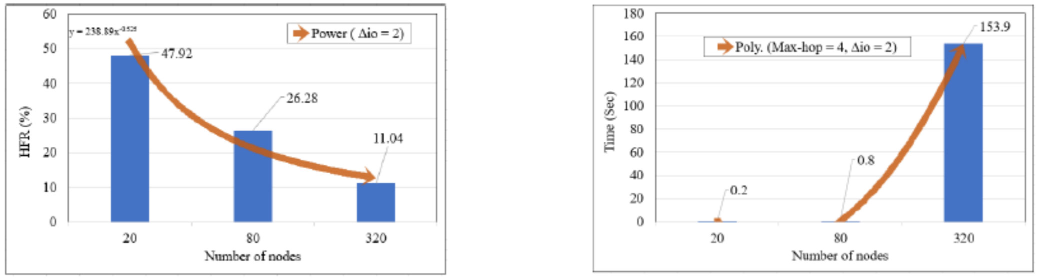
➔Average optimization time increases, HFR decreases

➔Heuristic algorithm gains an advantage over optimization.



(a) 8-k ports fat-tree network (b) 16-k ports fat-tree network

DUST ILP optimization computation time in large-scale network equivalent of (a) 8-k and (b) 16-k ports fat-tree.



(a) HFR (%) rate. (b) Average optimization time (sec).

scalability evaluation (a) HFR rate of heuristic algorithm (b) average computation time of optimization algorithm.

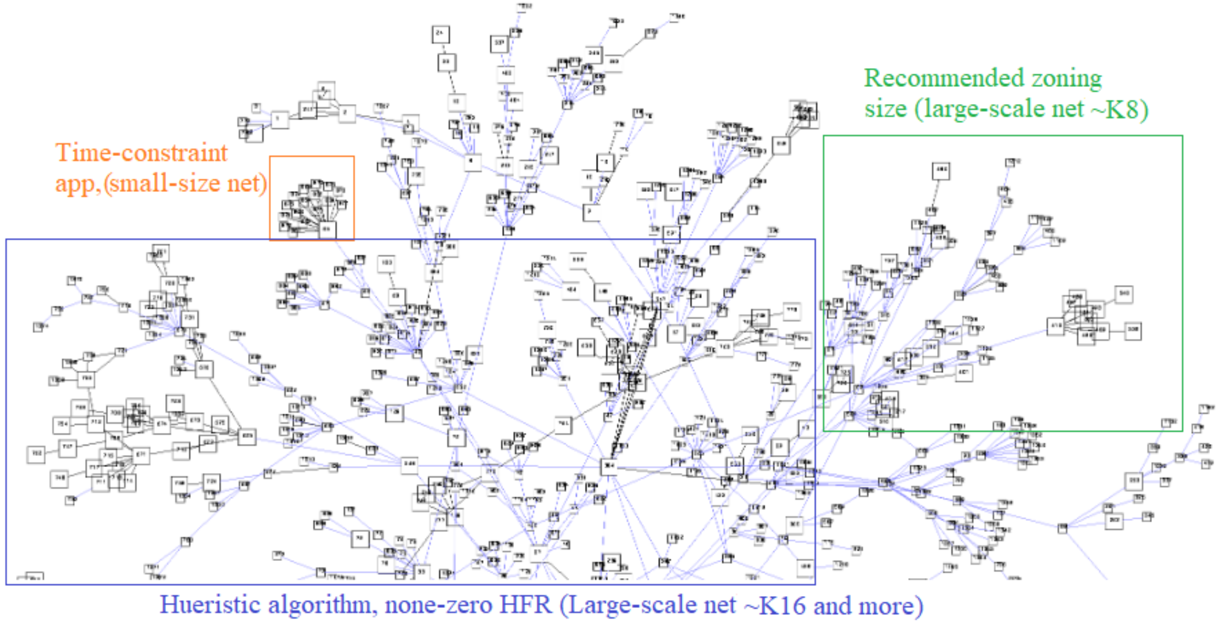
Scalability of Heuristic Algorithm

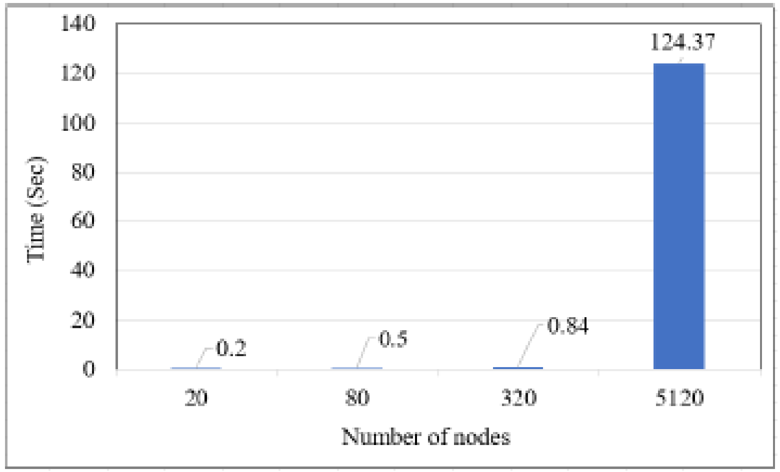
Dividing large-scale networks into zones with a max of 80 nodes

Results:

➔ **Optimization** cost of 0.8 sec, max-hop of 7 nodes

➔ ***Heuristic algorithm*** For larger networks, 124 sec with 5120 nodes.





Scalability evaluation of heuristic algorithm

Outline

1. Overview and Motivation
2. Proposed Solution System Architecture
3. DUST System Nodes and Workflows
4. Network Monitoring Placement
5. Performance Evaluations
6. **Conclusion And Future Work**

UC DAVIS COLLEGE OF ENGINEERING

Hewlett Packard Enterprise

Conclusion

- **Distributed Resource and Traffic Awareness** network in-device monitoring
  - **Hardware agnostic** by harnessing available compute resources of network,
  - Supporting network monitoring for **low-profile switches** over remote offloading.
  - **Offloading decision** is central while **monitoring** is distributed
  - Comprehensive **system-level** solution with **node designation** and **workflows** for optimized distributed offloading
  - **Dynamic** & **flexible offloading** with a **controllable route** solution

- Optimized distribution based on network-defined **constraints**, targeting **minimum response time** to mitigate the compute vs connectivity trade-off
- Mathematically formulated the optimal relocation of computations as an ILP, along with a **heuristic algorithm,** addressing the **scalability** concerns associated with the **computational complexity of optimization**

Future Work

- DUST-Manager to integrated with large-scale and cloud-based network providers e.g., **HPE GreenLake, AFC**, AMD Pensando **Policy Service Manager (**PSM) **; render a one-stop solution**
- Facilitating **dynamic policy updates** by interacting with policy managers
- DUST-Client to integrate with **DSS switch** equipped with a Smart ASIC e.g. **AOS-CX 10000**
- Applicable to **broader applicability** than service telemetry, finding relevance and utility in diverse **use cases and network services**.
- **AI** Improvement for **optimization model** and **offloading workload; traffic engineering**

Thank You!

Questions?

Mehrnaz Sharifian
email: msharifian@ucdavis.edu

UC DAVIS COLLEGE OF ENGINEERING

Hewlett Packard Enterprise

Complexity Analysis:

ILP Optimization vs. Heuristic Algorithm:	
p: feasible routes between each desired pair of nodes {i,j},	
maximum |p| : order of k^2 between each node pair (k-port fat-tree)	
F(|p|): time complexity of moving data between nodes.	
Assuming : all k(k + 1) nodes being either Busy nodes or Offload-candidate nodes	
ILP Optimization Complexity	**Heuristic algorithm Complexity**
F(|p|) : $k^4(k+1)^2 \approx k6$	F(|p|) : $k^2(k+1) \approx k3$