PULCEO IN ACTION

Towards an API-driven Approach for Universal and Lightweight Cloud-Edge Orchestration

Sebastian Böhm and Guido Wirtz

University of Bamberg, Germany

https://github.com/spboehm/pulceo-misc

Cloud-Edge Orchestration

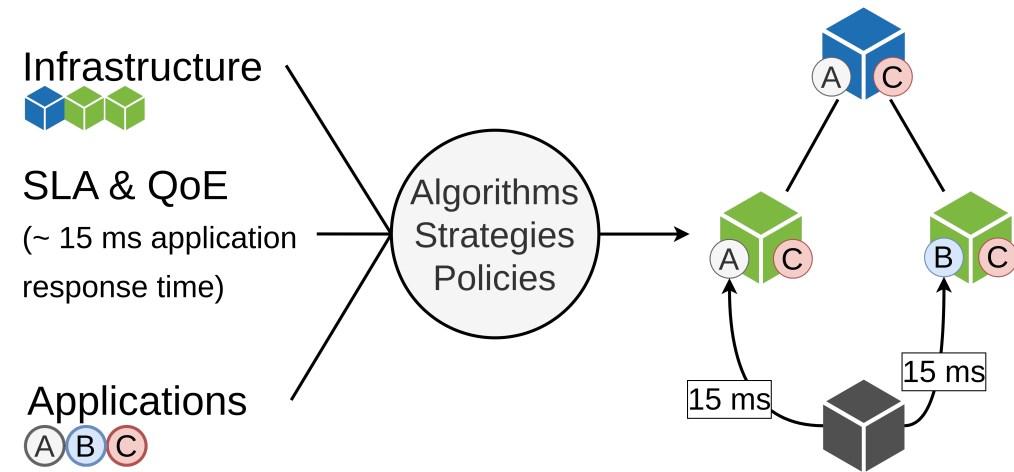**Edge computing:** Placing of computational resources close to end-users.

Many solutions exist for **service placement**

But, limited

- **Reproducibility**

- (General) **Applicability**

because of

- **custom** implementations

- missing **real-world experiments**

Infrastructure

SLA & QoE
(~ 15 ms application response time)

Applications
(A)(B)(C)

Algorithms
Strategies
Policies

A C

A C   B C

15 ms   15 ms

Similar **infrastructures**, **optimization goals**, and **orchestration** operations.

Simulations are prevalent: only 19 out of 99 solutions used a small test-bed.[1]

1. S. Smolka and Z. Á. Mann, "Evaluation of fog application placement algorithms: a survey," Computing, vol. 104, no. 6, pp. 1397–1423, Jun. 2022.

Holistic Management

Platform for Universal and Lightweight Cloud-Edge Orchestration (PULCEO)

Primary Activities for Cloud-Edge Orchestration (Experiments)
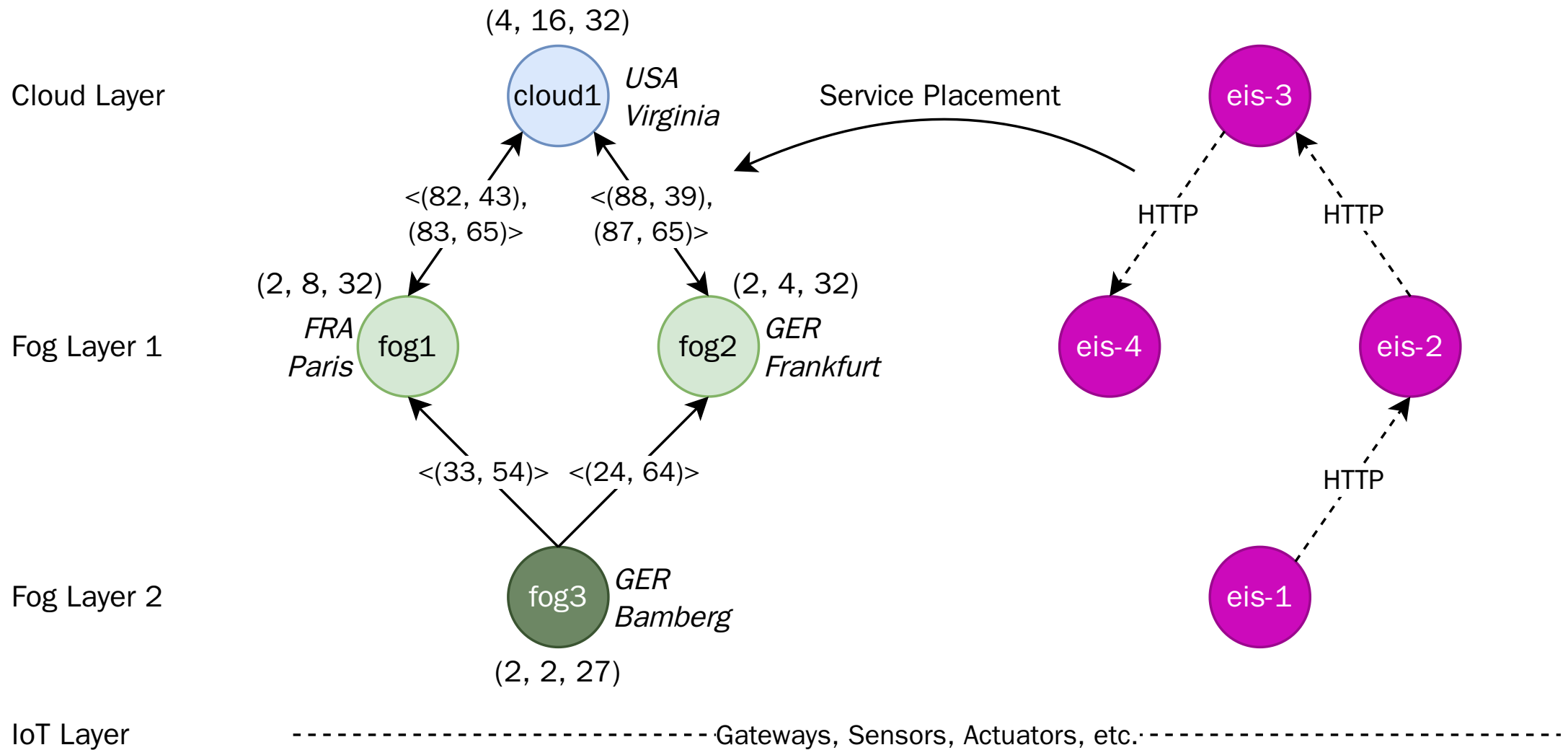
https://github.com/spboehm/pulceo-misc
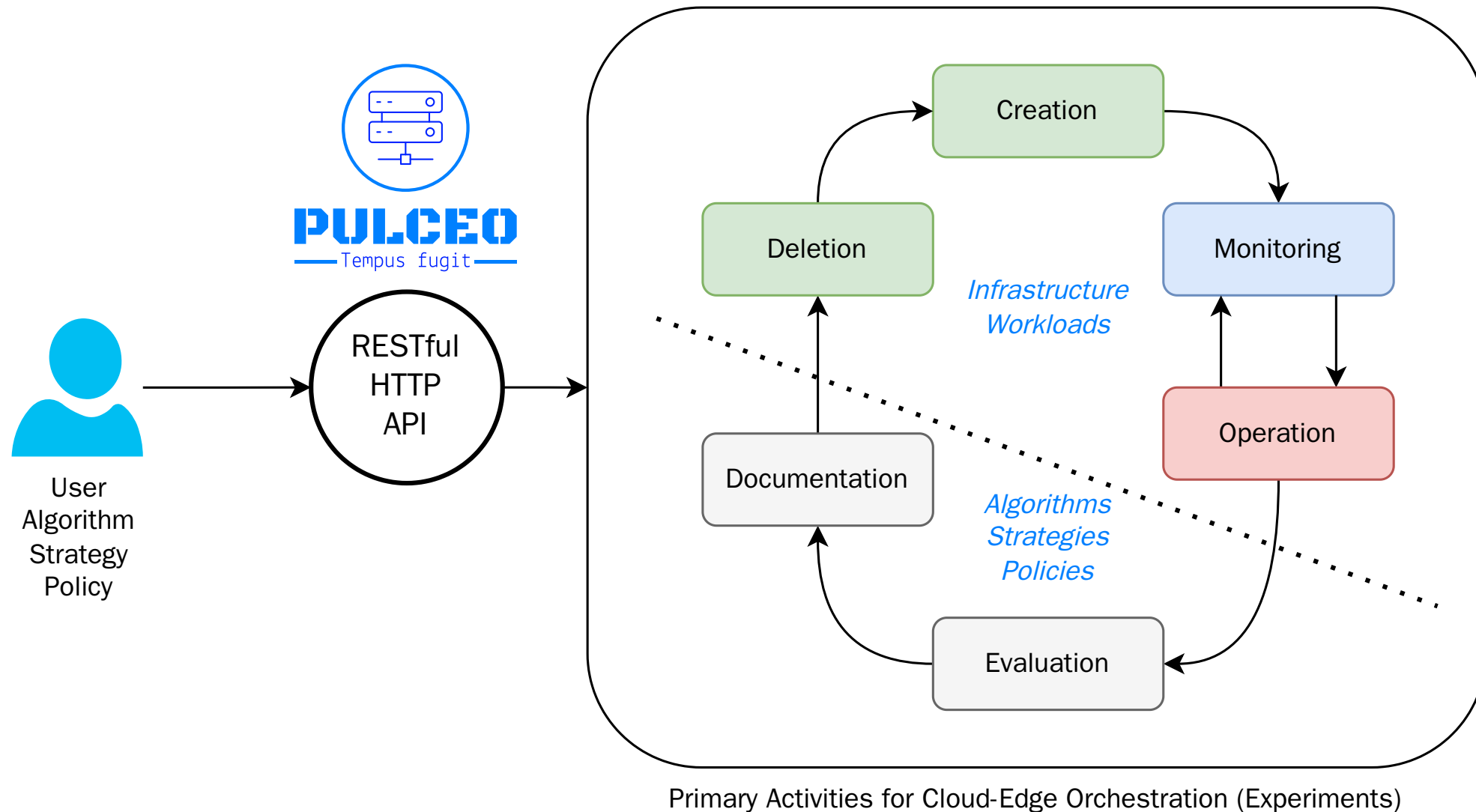
Reduced Example

Topology · Orchestration workflow

Topology

Representational cloud-edge topology with nodes, links, and requests for service placement.

Creation

Providers

Providers supply computational resources, e.g., Compute, Network, Storage, etc.

Two types of providers:

- **On-premises** providers (any virtual machine), built-in

- **Cloud** providers (API availability), e.g., Microsoft Azure

Example: Creation of Microsoft Azure as provider with a service principal

```
 1  curl --request POST \
 2    --url http://localhost:8081/api/v1/providers \
 3    --header 'Accept: application/json' \
 4    --header 'Content-Type: application/json' \
 5    --data '{
 6      "providerName": "azure-provider",
 7      "providerType": "AZURE",
 8      "clientId": "00000000-00000000-00000000-00000000",
 9      "clientSecret": "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA",
10      "tenantId": "00000000-00000000-00000000-00000000",
11      "subscriptionId": "00000000-00000000-00000000-00000000"
12    }'
```
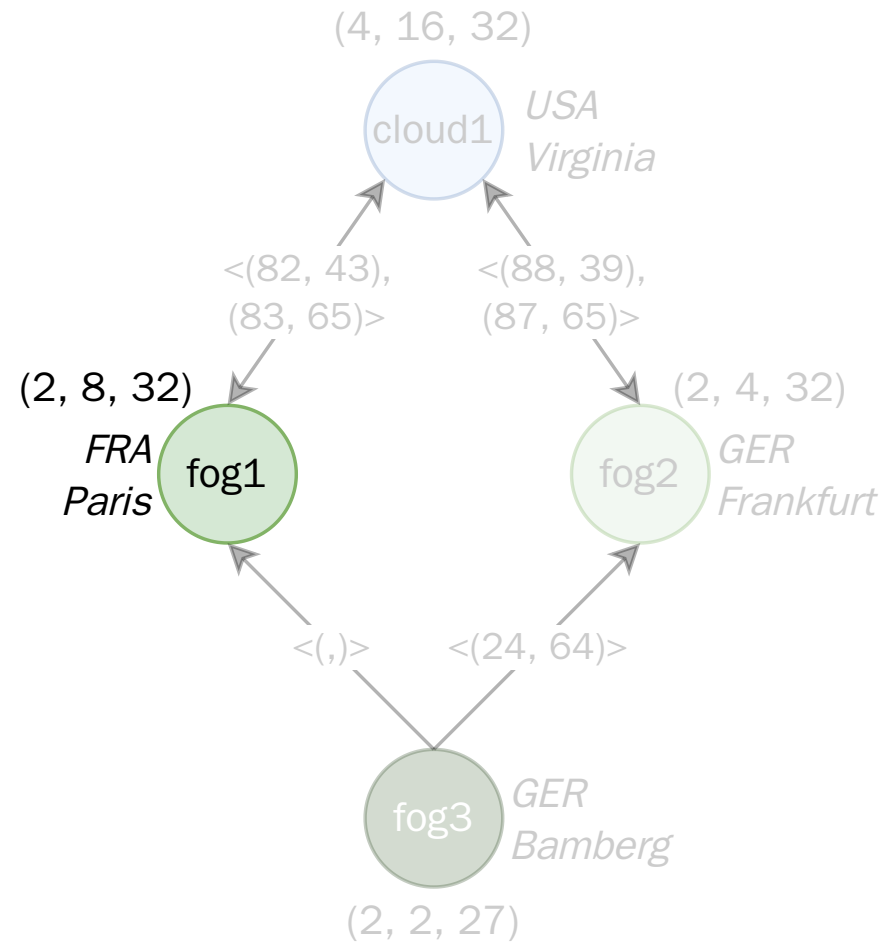
Nodes (fog1)

- Provider: Microsoft Azure (Cloud)

- Capabilities: 2 vCPU, 8 GB memory, 32 GB storage

- Location: France, Paris

```
1  curl --request POST \
2    --url http://localhost:8081/api/v1/nodes \
3    --header 'Accept: application/json' \
4    --header 'Authorization: XXXXX' \
5    --header 'Content-Type: application/json' \
6    --data '{
7      "nodeType":"AZURE",
8      "providerName":"azure-provider",
9      "name":"fog1",
10     "type":"fog",
11     "cpu":"2",
12     "memory":"8",
13     "region":"francecentral",
14     "tags": []
15  }'
```

(4, 16, 32)

cloud1   *USA*
         Virginia

<(82, 43),   <(88, 39),
(83, 65)>   (87, 65)>

(2, 8, 32)   (2, 4, 32)

FRA   fog1   fog2   *GER*
Paris   *Frankfurt*

<(,)>   <(24, 64)>

                fog3   *GER*
                          Bamberg

(2, 2, 27)

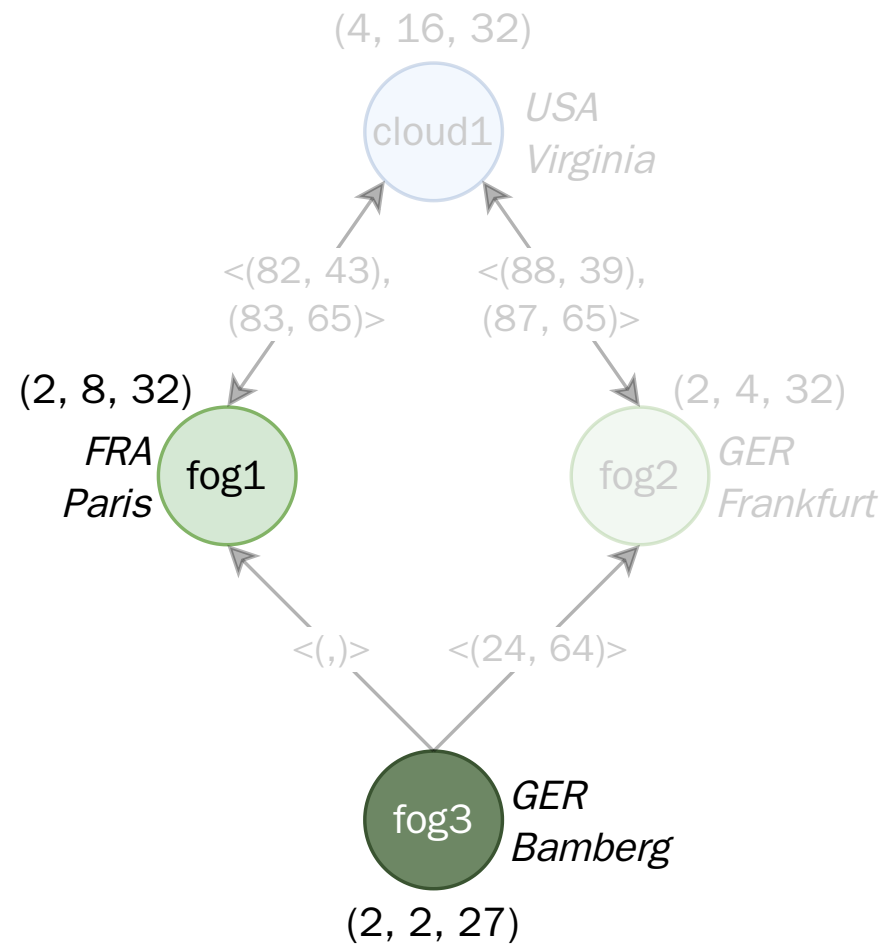https://github.com/spboehm/pulceo-misc

Nodes (fog3)

- Provider: Local data center (On-premises)

- Capabilities: 2 vCPU, 2 GB memory, 27 GB storage

- Location: Bamberg, Germany

```
1   curl --request POST \
2     --url http://localhost:8081/api/v1/nodes \
3     --header 'Accept: application/json' \
4     --header 'Authorization: XXXXX' \
5     --header 'Content-Type: application/json' \
6     --data '{
7       "nodeType":"ONPREM",
8       "type": "fog",
9       "name": "fog3",
10      "providerName":"default",
11      "hostname":"h5138.pi.uni-bamberg.de",
12      "pnaInitToken":"XXXXX",
13      "country": "Germany",
14      "state": "Bavaria",
15      "city": "Bamberg",
16      "latitude": 49.9036,
17      "longitude": 10.8700,
18      "tags": []
19  }'
```

(4, 16, 32)

cloud1   *USA*
          Virginia

<(82, 43),   <(88, 39),
(83, 65)>   (87, 65)>

(2, 8, 32)   (2, 4, 32)

FRA   *GER*
Paris   fog1   fog2   *Frankfurt*

<(,)>   <(24, 64)>

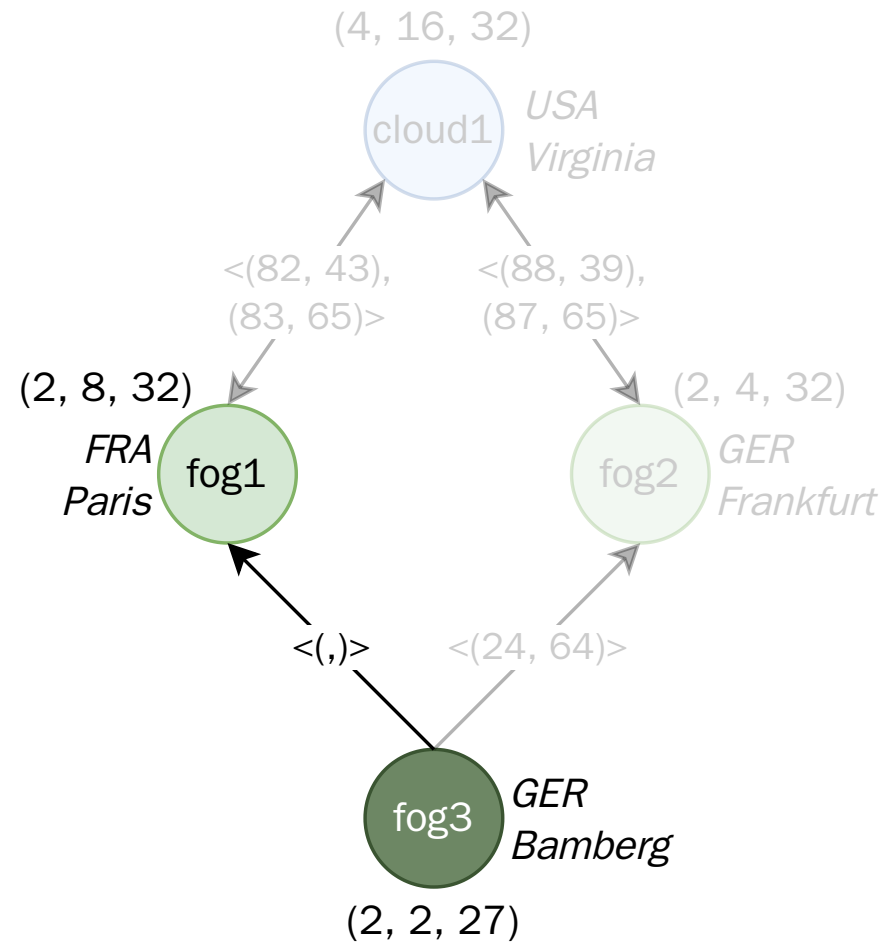fog3   *GER*
        Bamberg

(2, 2, 27)

Links (Example fog3-fog1)

- Link between fog3 and fog1

- Represents a logical connection

- Later used to obtain round-trip time and bandwidth between nodes

- <(,)>: <(*latency*, *bandwidth*), …>

```
1   curl --request POST \
2     --url http://localhost:8081/api/v1/links \
3     --header 'Accept: application/json' \
4     --header 'Authorization: XXXXX' \
5     --header 'Content-Type: application/json' \
6     --data '{
7       "linkType": "NODE_LINK",
8       "name": "fog3-fog1",
9       "srcNodeId": "fog3",
10      "destNodeId": "fog1"
11    }'
```

(4, 16, 32)

cloud1   *USA*
         Virginia

<(82, 43),   <(88, 39),
(83, 65)>   (87, 65)>

(2, 8, 32)   (2, 4, 32)

FRA   fog1   fog2   *GER*
Paris   *Frankfurt*

<(,)>   <(24, 64)>

fog3   *GER*
        Bamberg

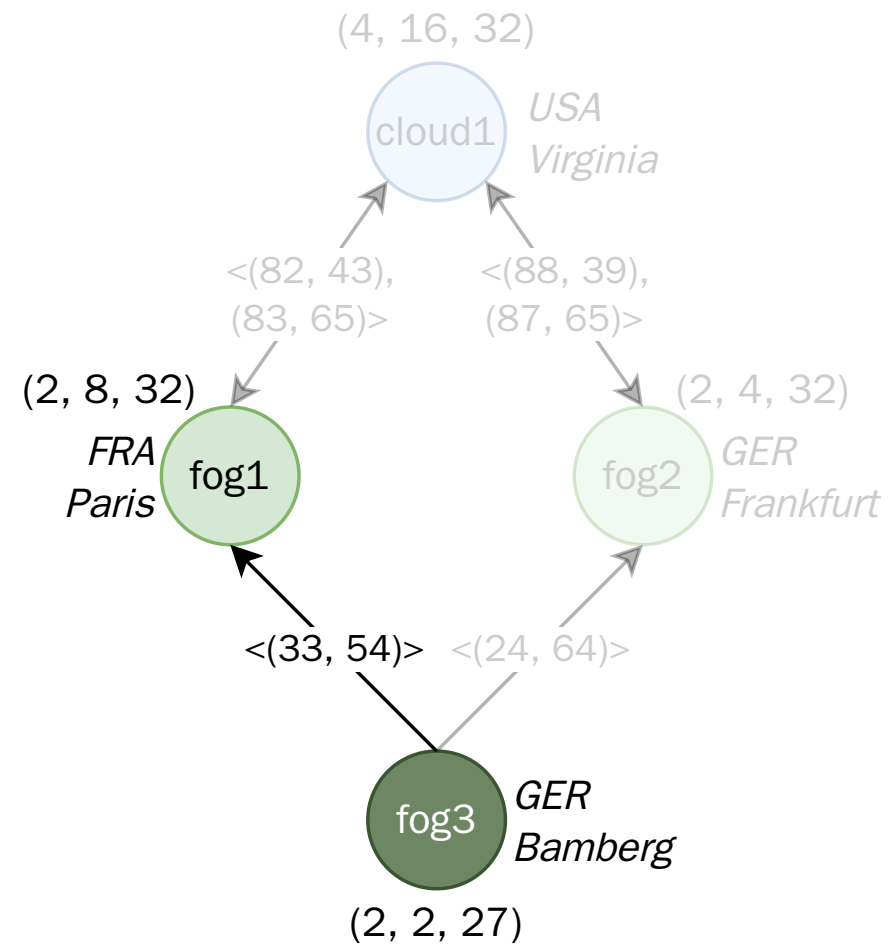(2, 2, 27)

Monitoring

Nodes · Links

Metric Requests

- Collection of monitoring data

  - CPU, memory, storage, and network utilization for nodes and applications

  - ICMP round-trip time, TCP & UDP bandwidth for links

- Individual and batch (*) assignments

- **Example:** Latency all for links, once per hour (recurrence 3600s = 1h)

```
1  curl --request POST \
2    --url http://localhost:8081/api/v1/metric-reques
3    --header 'Accept: application/json' \
4    --header 'Authorization: XXXX' \
5    --header 'Content-Type: application/json' \
6    --data '{
7      "type":"icmp-rtt",
8      "linkId":"*",
9      "recurrence":"3600"
10  }'
```

(4, 16, 32)

cloud1   *USA*
           Virginia

<(82, 43),   <(88, 39),
(83, 65)>   (87, 65)>

(2, 8, 32)   (2, 4, 32)

FRA   fog1   fog2   *GER*
Paris   *Frankfurt*

<(33, 54)>   <(24, 64)>

         fog3   *GER*
                 Bamberg

(2, 2, 27)

Operation

Workload · Resource Management · Service Placement

Workload

- Placement of edge-iot-simulator (eis)[1] on node fog1
- Mimics a typical application:
  - Simulates a temperature sensor (sends temperature readings at regular intervals)
  - Allows to perform HTTP requests to simulate a microservice application
- Example resource assignments:
  - 1 vCPU (1000 shares)
  - 1 GB memory (1000 MB)

1. https://github.com/spboehm/edge-iot-simulator
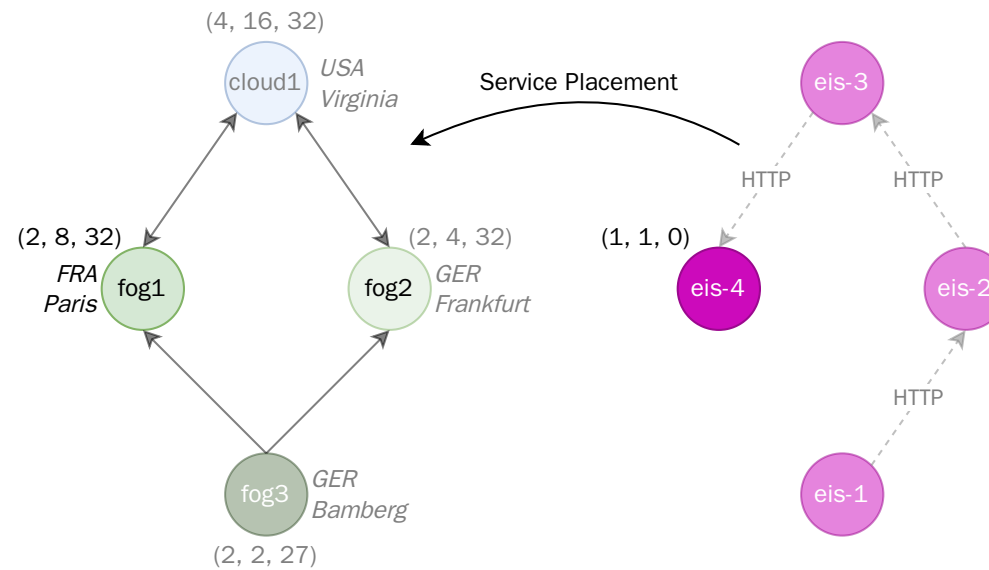
Resource Management (Example CPU)

Read CPU resources of fog1

```
1  curl --request GET \
2    --url http://localhost:8081/api/v1/nodes/fog1/cp
3    --header 'Accept: application/json' \
4    --header 'Authorization: XXXXX' \
```

```
1  {
2      "uuid": "8aeae447-a552-4ea2-86a3-2bd1f79d6117"
3      "nodeUUID": "e1076174-380a-47e4-a468-b9fd1b0ea
4      "nodeName": "fog1",
5      "cpuCapacity": {...},
6      "cpuAllocatable": {
7          "modelName": "Intel(R) Xeon(R) Platinum 82
8          "cores": 2,
9          "threads": 2,
10         "bogoMIPS": 5187.81,
11         "minimalFrequency": 2593.906,
12         "averageFrequency": 2593.906,
13         "maximalFrequency": 2593.906,
14         "shares": 2000,
15         "slots": 0.0,
16         "mips": 5187.81,
17         "gflop": 0.0
18     }
19  }
```

Update CPU resources of fog1

```
1  curl --request PATCH \
2    --url http://localhost:8081/api/v1/nodes/fog1/cp
3    --data '{
4      "key": "shares",
5      "value": 1000
6  }'
```

```
1  {
2      "uuid": "8aeae447-a552-4ea2-86a3-2bd1f79d6117"
3      "nodeUUID": "e1076174-380a-47e4-a468-b9fd1b0ea
4      "nodeName": "fog1",
5      "cpuCapacity": {...},
6      "cpuAllocatable": {
7          "cores": 2,
8          "threads": 2,
9          "bogoMIPS": 5187.81,
10         "averageFrequency": 2593.906,
11         "shares": 1000,
12         "slots": 0.0,
13         "mips": 5187.81,
14         "gflop": 0.0,
15     ...
16     }
17  }
```

https://github.com/spboehm/pulceo-misc

Applications (Service Placement)

```
1   curl --request POST \
2     --url http://localhost:8081/api/v1/applications \
3     --header 'Accept: application/json' \
4     --header 'Authorization: XXXXX' \
5     --data '{
6       "nodeId": "fog1",
7       "name": "edge-iot-simulator",
8       "applicationComponents": [
9           {
10              "name": "component-eis",
11              "image": "ghcr.io/spboehm/edge-iot-simulator:v1.1.0",
12              "port": 80,
13              "protocol": "HTTPS",
14              "applicationComponentType": "PUBLIC",
15              "environmentVariables": {
16                  "MQTT_SERVER_NAME":"XXXXX.s1.eu.hivemq.cloud",
17                  "MQTT_PORT":"8883",
18                  "MQTT_TLS":"True",
19                  "MQTT_USERNAME": "XXXXX",
20                  "MQTT_PASSWORD": "XXXXX",
21                  "MQTT_CLIENT_ID":"fog1-edge-iot-simulator",
22                  "WEB_PORT": 80
23              }
24          }
25      ]
26  }'
```
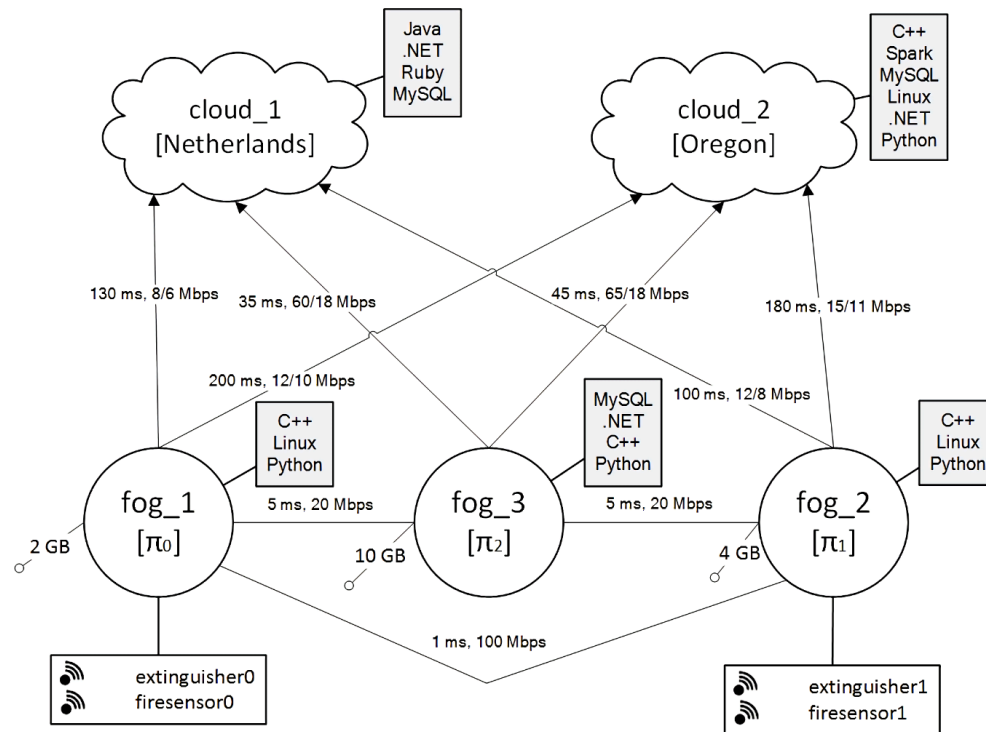
If needed, further Metric Requests to monitor the placed Applications can be issued.

https://github.com/spboehm/pulceo-misc

Use Case

Reproducing the service placement strategy by **Brogi and Forti 2017**[1]

Proposed Cloud-Edge Orchestration Topology.

- Service placement (Greedy)

- Input:

  - Services with requirements (hardware resources, latency and bandwidth)

  - Node and link capabilities

- Operations: PREPROCESS, CHECKHARDWARE, CHECKLINKS, DEPLOY, UNDEPLOY

- Output: Service placement decisions

1. A. Brogi and S. Forti, "QoS-Aware Deployment of IoT Applications Through the Fog," IEEE Internet Things J., vol. 4, no. 5, pp. 1185–1192, Oct. 2017, doi: 10.1109/JIOT.2017.2701408.

https://github.com/spboehm/pulceo-misc

Evaluation

API Requests · Idle Resource Utilization · Link Quality Metrics · Application Resource Utilization · Application Response Time

API Requests

69 requests required to perform the entire orchestration workflow
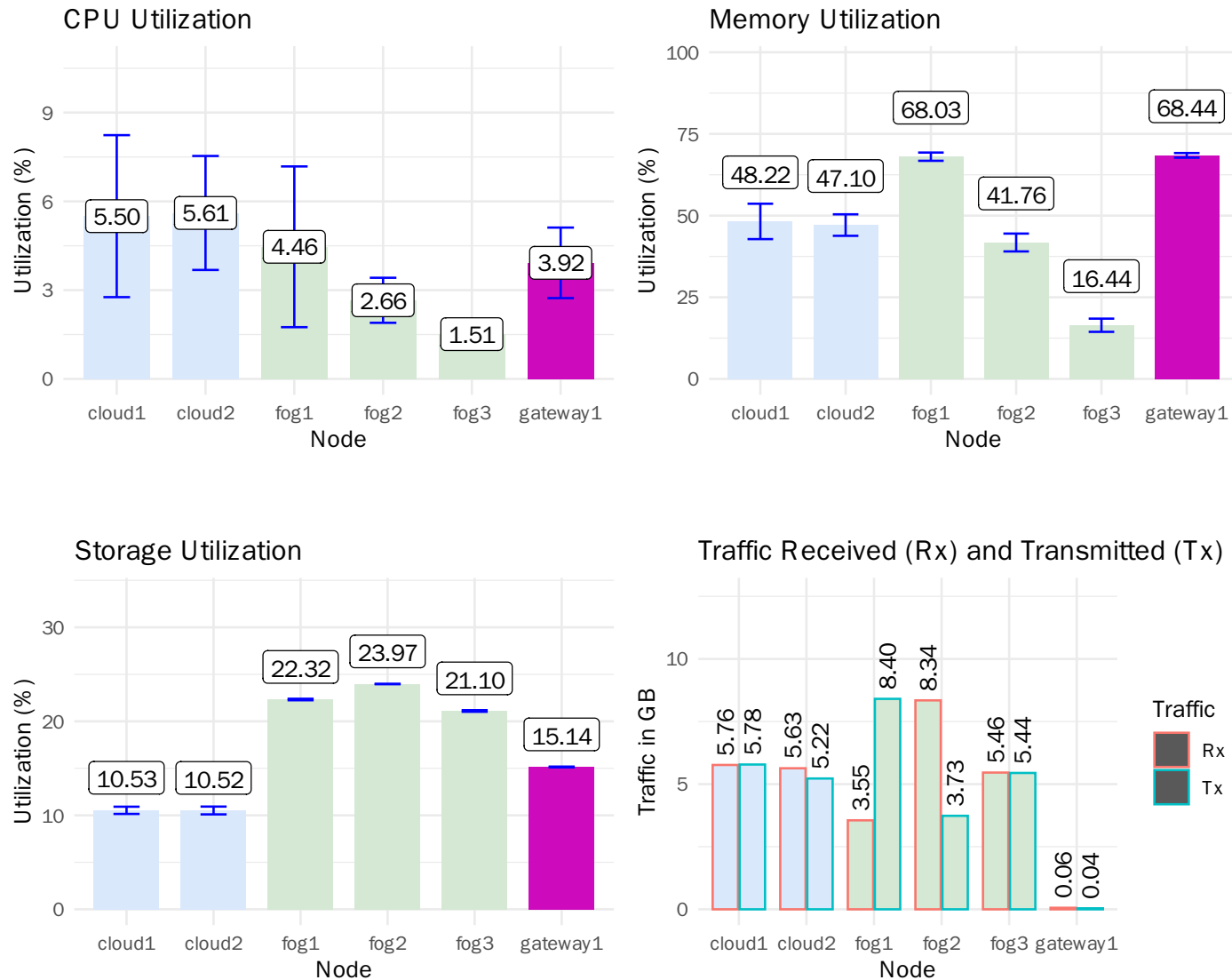
Decoupled orchestration with support for algorithmic steps

#	Service	Endpoint	Method	Count	Algorithmic step [10]	Explanation
1	PRM	/providers	POST	1	–	Register Azure as *Provider*
2	PRM	/nodes	POST	5	–	Create *Nodes*
3	PRM	/links	POST	15	–	Create *Links* between *Nodes*
4	PMS	/metric-requests	POST	10	–	Create *Metric Requests* for *Nodes* (4) and *Links* (6)
5	PRM	/resources/cpu	GET	1	PREPROCESS	Read all allocatable CPU resources
6	PRM	/resources/memory	GET	1	PREPROCESS	Read all allocatable memory resources
7	PRM	/tags?type=node	GET	1	PREPROCESS	Read tags for *Nodes* for reducing the input space
8	PRM	/nodes/{resourceId}/cpu	GET	5	CHECKHARDWARE	Read allocatable shares (CPU) of a particular *Node*
9	PRM	/nodes/{resourceId}/memory	GET	5	CHECKHARDWARE	Read allocatable size (memory) of a particular *Node*
10	PRM	/nodes/{resourceId}/tags	GET	5	CHECKHARDWARE	Read tags of a particular *Node*
11	PMS	/metrics?type=link	GET	1	CHECKLINKS	Read last link quality metrics for all *Links*
12	PRM	/nodes/{resourceId}/cpu	PATCH	5	DEPLOY	Update allocatable shares (CPU) of a particular *Node*
13	PRM	/nodes/{resourceId}/memory	PATCH	5	DEPLOY	Update allocatable size (memory) of a particular *Node*
14	PSM	/applications	POST	5	DEPLOY	Deploy the *Application* edge-iot-simulator on all *Nodes*
15	PSM	/applications/{resourceId}	DELETE	0	UNDEPLOY	Delete an instance of edge-iot-simulator
16	PMS	/metric-requests	POST	4	–	Create *Metric Requests* for *Applications*

https://github.com/spboehm/pulceo-misc

Idle Resource Utilization by Nodes

Including **pulceo-node-agent,** fully configured **monitoring,** and **Kubernetes**

CPU Utilization

Memory Utilization

Storage Utilization

Traffic Received (Rx) and Transmitted (Tx)

https://github.com/spboehm/pulceo-misc

Link Quality Metrics

Using a **high-performance** and **stable** network for on-premises and cloud

| v_1	v_2	ICMP round-trip time (ms)					TCP bandwidth (Mbps)					UDP bandwidth (Mbps)				
		Min	Mean	Max	Med	SD	Min	Mean	Max	Med	SD	Min	Mean	Max	Med	SD
cloud1	fog1	23.03	23.37	23.98	23.32	0.23	65.00	65.00	65.00	65.00	0.00	64.20	64.58	64.70	64.65	0.14
cloud1	fog2	23.15	23.44	23.71	23.43	0.16	65.00	65.00	65.00	65.00	0.00	64.20	64.60	64.70	64.70	0.14
cloud1	fog3	21.68	22.03	22.31	22.04	0.17	65.00	65.00	65.00	65.00	0.00	64.30	64.58	64.70	64.60	0.13
cloud2	fog1	168.68	169.16	172.27	168.99	0.69	31.20	49.80	65.00	46.25	11.46	62.60	62.75	62.90	62.80	0.09
cloud2	fog2	168.70	169.07	169.77	169.06	0.22	24.70	58.50	65.00	65.00	10.65	61.60	62.69	62.90	62.70	0.25
cloud2	fog3	168.80	169.12	170.28	169.07	0.29	41.70	54.05	65.00	52.00	8.22	62.60	62.74	62.90	62.70	0.09
fog1	cloud1	23.12	23.48	26.61	23.32	0.69	64.60	64.98	65.00	65.00	0.08	64.40	64.59	64.70	64.60	0.11
fog1	cloud2	168.79	169.13	169.63	169.04	0.25	46.10	63.51	65.00	65.00	4.70	61.20	62.68	62.90	62.80	0.37
fog1	fog2	0.87	1.01	1.15	1.01	0.07	100.00	100.00	100.00	100.00	0.00	99.80	99.95	100.00	99.95	0.06
fog1	fog3	0.96	1.05	1.16	1.05	0.05	65.00	65.00	65.00	65.00	0.00	65.00	65.00	65.00	65.00	0.00
fog2	cloud1	23.24	23.46	24.03	23.39	0.22	54.30	63.92	65.00	65.00	3.01	64.20	64.60	64.70	64.65	0.14
fog2	cloud2	168.75	169.19	171.87	169.02	0.60	57.70	64.65	65.00	65.00	1.50	62.60	62.78	62.90	62.80	0.06
fog3	cloud1	21.82	22.09	22.57	22.05	0.20	52.20	63.63	65.00	65.00	3.44	64.40	64.58	64.70	64.65	0.13
fog3	cloud2	168.73	169.22	171.53	169.01	0.58	7.97	55.04	65.00	65.00	20.49	62.50	62.73	62.80	62.80	0.09
fog3	fog2	0.90	1.05	1.21	1.05	0.07	65.00	65.00	65.00	65.00	0.00	64.90	65.00	65.00	65.00	0.02

Application Resource Utilization

With deployed edge-iot-simulators (eis)
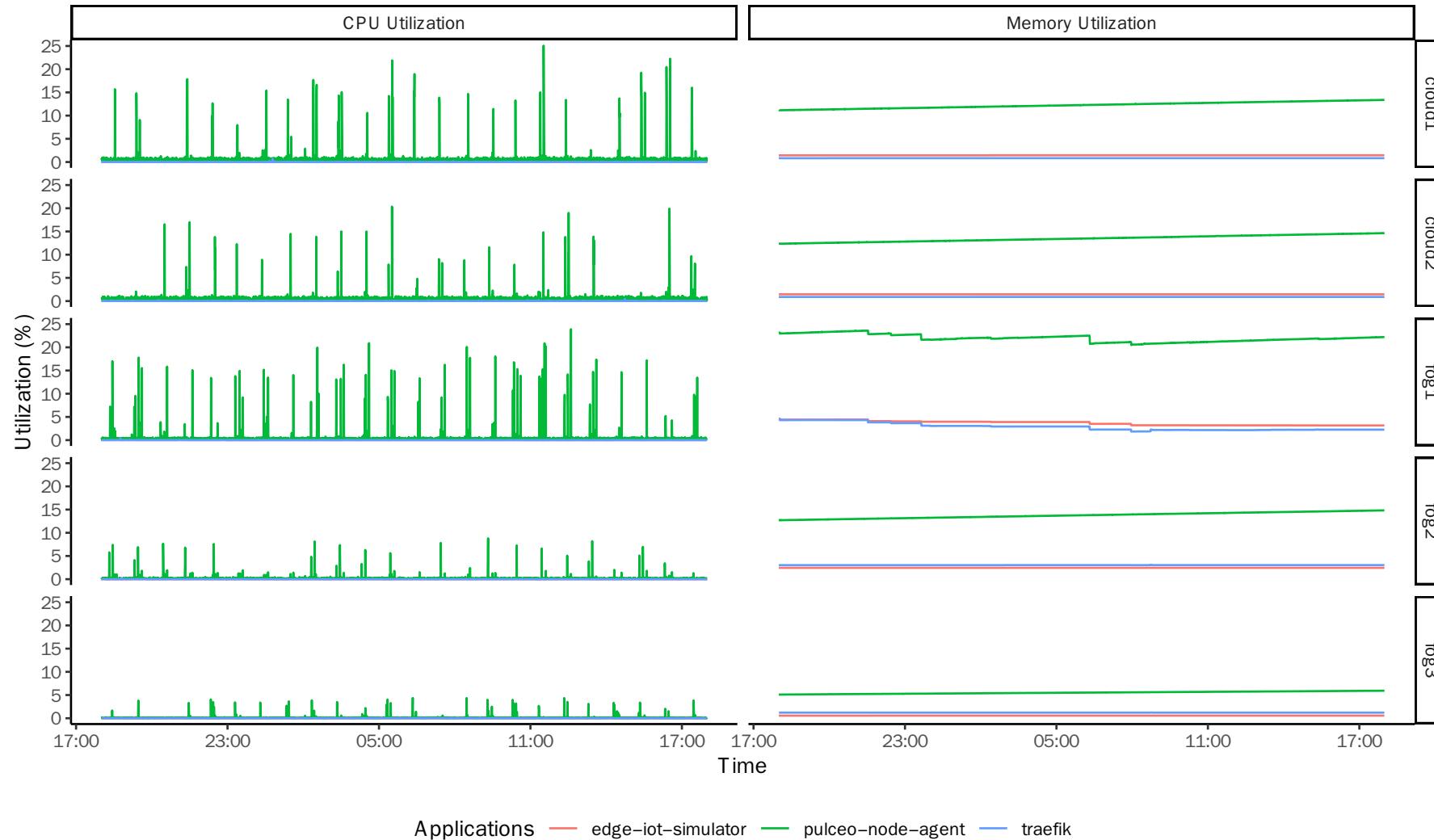
Application Response Time

- Measured by edge-iot-simulators (eis)
- Values have been submitted in a standardized JSON format via MQTT
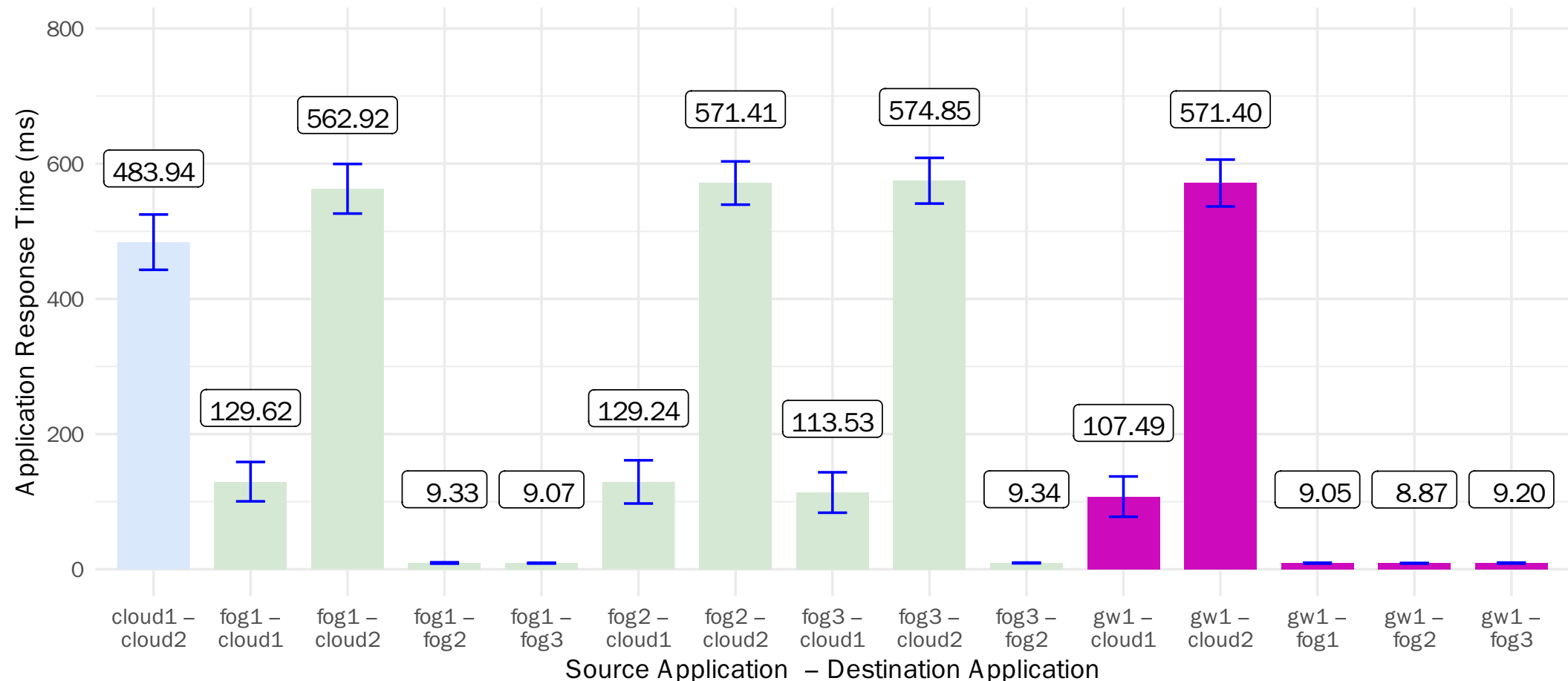
Documentation

Orchestration Data

Orchestration Data

JSON export of all entities of the domain model:
Providers, *Nodes*, *Links*, *Metric-Requests*, *Applications*, *Resources*, *CPUs*, *Memory*, *Events*

Example for nodes:

```
1  curl --request GET \
2    --url http://localhost:8081/api/v1/nodes \
3    --header 'Accept: application/json' \
4    --header 'Authorization: XXXXX' \
```

▶ JSON output for nodes

Report for SOSE2024-prod

https://github.com/spboehm/pulceo-misc

Related Solutions

EU Projects (Horizon Research), like CODECO, FogAtlas, SODALITE@RT, ENACT, etc.

- Latency and bandwidth measurement not fully implemented

- Scheduling for service placement often pre-implemented

- Focus not on scientific experiments

Conceptual and **prototypical research efforts**, like Sophos, Fluidity, ACOA, etc.

- No holistic cloud-edge orchestration (see above)

- Lack of documentation

Out of scope: **Cloud** and **commercial** solutions

https://github.com/spboehm/pulceo-misc

Contributions

- Fully documented **RESTful HTTP API** for universal orchestration

- **Decoupled** and **holistic** cloud-edge orchestration with evaluation and documentation

- Strong **scientific** and **industrial** foundation

  - Platform architecture based on a scientific meta-study[1]

  - Feature engineering based on scientific, peer-reviewed service placement publications[2]

  - Implementation following an industry standard (OpenFog RA)[3]

Limitations: Only a few solutions with stable network conditions have been reproduced

1. B. Costa, J. Bachiega, L. R. de Carvalho, and A. P. F. Araujo, "Orchestration in Fog Computing: A Comprehensive Survey," ACM Comput. Surv., vol. 55, no. 2, pp. 1–34, Feb. 2022, doi: 10.1145/3486221.

2. https://spboehm.github.io/pulceo-misc/

3. S. Böhm and G. Wirtz, "PULCEO - A Novel Architecture for Universal and Lightweight Cloud-Edge Orchestration," in 2023 IEEE International Conference on Service-Oriented System Engineering (SOSE), Athens, Greece: IEEE, Jul. 2023, pp. 37–47.

https://github.com/spboehm/pulceo-misc

Further Resources

Platform-related:

- Source Code

- Pre-built Container Images

- OpenAPI Specifications

- Documentation

Orchestration-specific:

- Orchestration reports

- Example API requests