SMTaaS: Serving problem solving workloads over the computing continuum

Pantelis Frangoudis
Distributed Systems Group, TU Wien

Joint work with Stefan Holzer, Christos Tsigkanos, Schahram Dustdar

Motivation

Emerging CPS/IoT systems

- Strict functional & non-functional specifications
  - E.g., safety-related
- Formalisms prescribe application behavior
- Continuous monitoring on service & environment status
- Often in practice:

  Problems can be formulated as Satisfiability Modulo Theories (SMT)

Many examples in the IoT space…

- Motion planning for robots (Imeson & Smith)
- Verifying correct operation of IoT services & edge computing systems (Avasalcai et al.)
- Detecting threats in rule-based smart home systems (Wang et al.)

F. Imeson, S.L. Smith, "An SMT-Based Approach to Motion Planning for Multiple Robots With Complex Constraints," IEEE Trans. Robotics, 2019.
C. Avasalcai et al., "Resource Management for Latency-Sensitive IoT Applications With Satisfiability," IEEE TSC, 2022.
Q. Wang et al., "Charting the Attack Surface of Trigger-Action IoT Platforms," ACM CCS, 2019.

… but also beyond that

- BGP control plane verification (Tang et al.)

- SDN security (Bringhenti et al.)

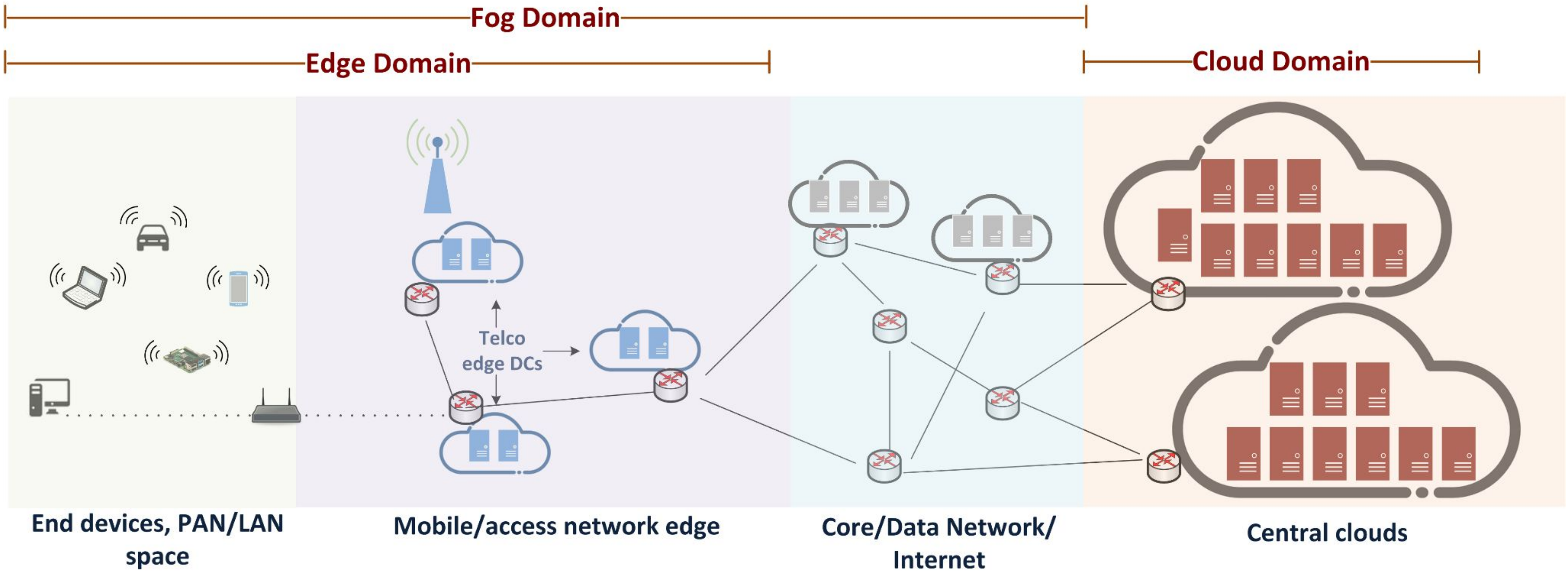
- Function placement & connectivity policy enforcement in NFV (Marchetto et al.)

A. Tang et al., "Lightyear: Using Modularity to Scale BGP Control Plane Verification," ACM SIGCOMM, 2023.
D. Bringhenti et al., "Automatic, verifiable and optimized policy-based security enforcement for SDN-aware IoT networks," Comput. Networks, 2022.
G. Marchetto et al., "A Formal Approach to Verify Connectivity and Optimize VNF Placement in Industrial Networks," IEEE Trans. Ind. Informatics, 2021.

Key observations

- Input data/problem instances originate at the edge
- Solving SMT problems can be **computationally expensive**
  - Problem if latency-critical operations depend on the outcome
- Solving in the cloud cannot always help (though sometimes it does)
  - Network latency may offset offloading gains
  - Intermittent connectivity? Confidentiality?

On the positive side…



The Computing Continuum

Key question

How to efficiently serve SMT workloads over distributed infrastructure along the computing continuum?

- **Architecture:** Transparent evaluation of SMT problems
- **Offloading decisions:** Where to solve (device, edge/fog, cloud)?

SMT-as-a-Service: System Design

Node abstraction

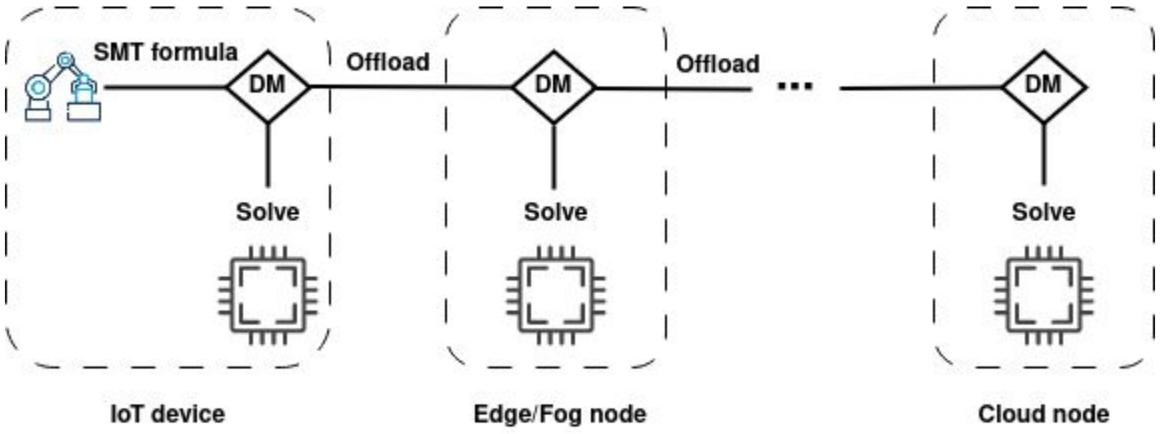
- **Workload:** SMT formulas, originating at IoT/edge devices

- **Solver** node:
  - Exposes API endpoints to accept properly encoded problem instances
  - Abstracts solver internals: any compatible SMT solver works

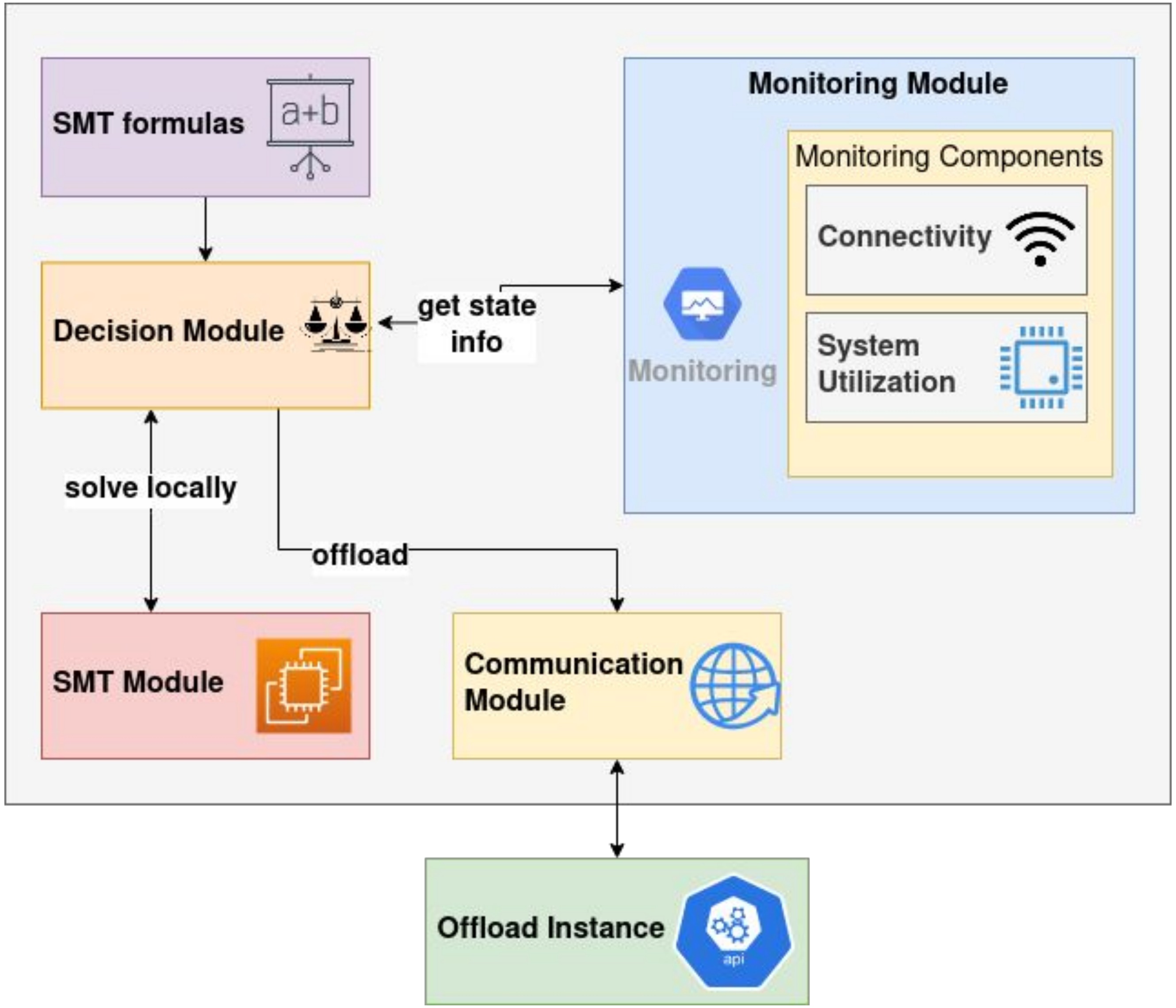
- **Interoperability**: SMT-LIB as the encoding format

Recursive architecture

- Invocations passed on along solver path edge-to-cloud
- Each node **independently** decides: solve locally or forward?
- Transparent to client/application



Solver node: bird's eye view



Where to solve an SMT problem instance?

Offloading decision making module

- Decides whether to solve a received formula **locally** or **offload it further**

- Each node maintains set of candidate offload targets

- Independent decision based on:
  - Information about the *formula* at hand
  - *Node capabilities* and *local view of system state* (e.g., latency, available battery)

- Plug-in framework for custom decision making
  - Should consider node capabilities – edge/IoT resource limitations
  - Different criteria possible – e.g., response time, energy cost

Decisions via reinforcement learning

- Examples investigated: Q-Learning, DQN
- **State:** formula + fog node conditions (e.g., latency to target)
- **Actions:** set of candidate offload targets
- **Reward:** depends on what we optimize for - latency, energy cost, weighted combination

Which DM module to run in a node?

- **Operator’s decision** – think of host capabilities
- **Q-Learning** with reduced state representation for low-end IoT nodes
  - E.g., resource-constrained robot
- **Deep Q-Network** on more capable fog nodes
  - E.g., on power supply, with GPU

Does it work?

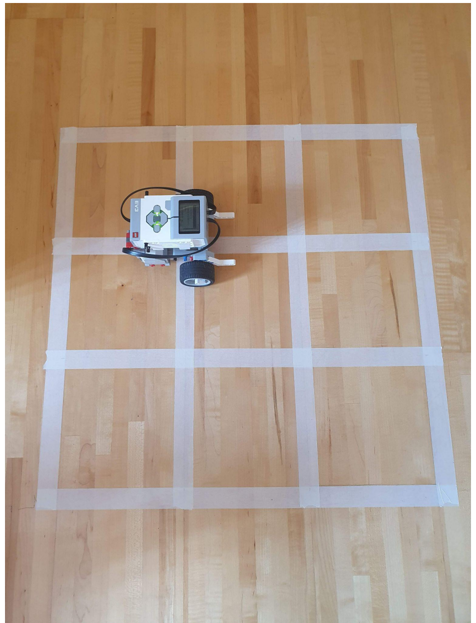
Implementation & experiments

- Open-source, runs on diverse platforms: low-end robots, RPi, …
  - https://github.com/Stefan2911/SMTaaS

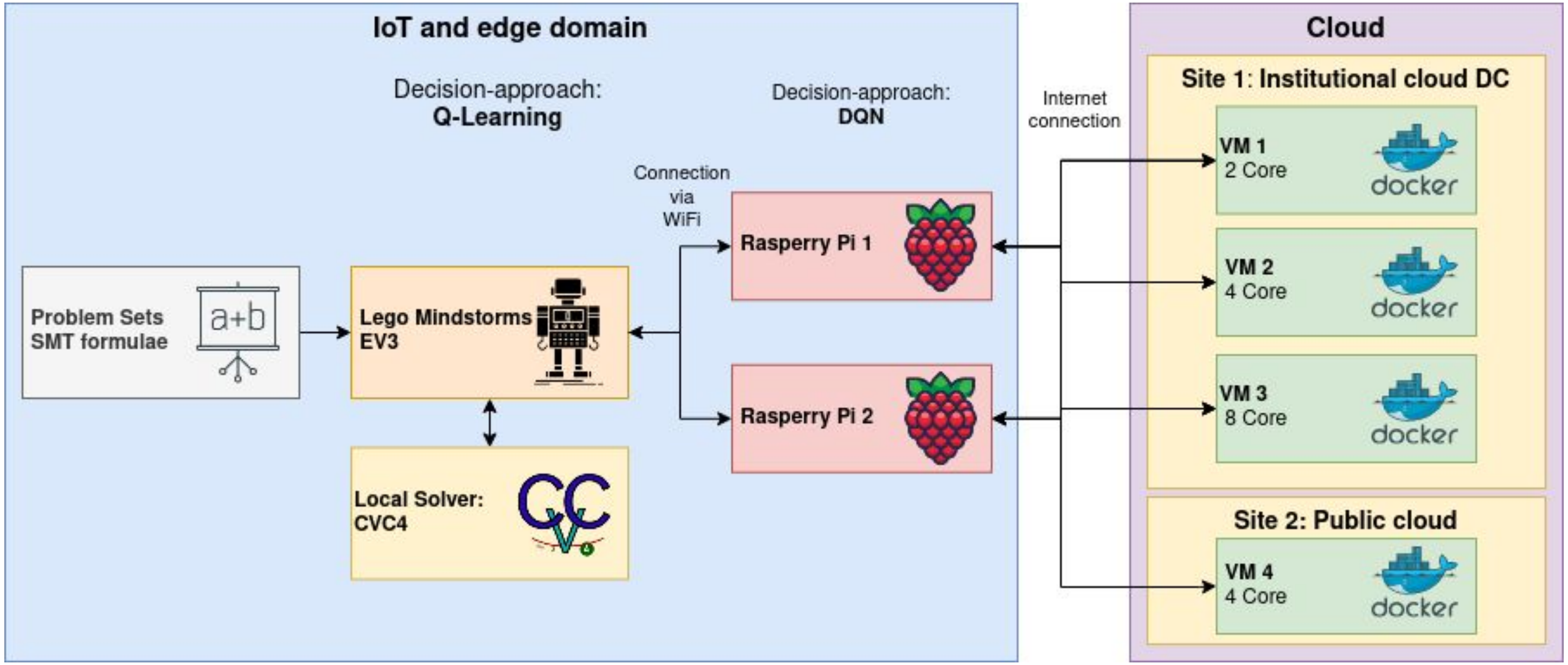
- Solver back-end depends on host capabilities
  - CVC4 @ IoT/edge, Z3 or MathSAT5 @ Cloud

- Testbed experiments w. SMT workloads
  from official SMT-LIB benchmark dataset

- 360º-view example from spec to evaluation
  - Path planning for fog-supported robots



Deployment scenario



- Workload: Simple, medium, hard SMT problems (following measurements)
- Introduce varying latency in edge-cloud path
- Use simplified/abstract energy cost model

Summary of results

- **Learns to balance** among local-edge-cloud execution for lowest latency
- **Saves >40% energy** for mixed workloads vs. device-only or offloading-only
- **Feasible & practical** to offer SMTaaS
- Offloading capability may be critical for CPS use cases
  - Robot path planning on 16-vertex grid: **171s (on-device)** vs. **2.6s (w. Q-learning based offloading)**

The way forward

Open problems

- More sophisticated offloading strategies
  - Capturing latency/cost constraints, more accurate state representation
- Scenarios beyond CPS
  - Network verification, integration with ETSI MEC for lower latency
- System aspects
  - Resource management & dynamic deployment of solver nodes, workload balancing, etc.

Thank you!

Pantelis Frangoudis

pantelis.frangoudis@dsg.tuwien.ac.at

More details: S. Holzer, P. Frangoudis, C. Tsigkanos, S. Dustdar, "SMT-as-a-Service for Fog-Supported Cyber-Physical Systems," Proc. ICDCN, 2024.