Poster Abstract: Enabling Elasticity on the Edge using Heterogeneous Gateways

Nabeel Nasir
University of Virginia
Charlottesville, VA, USA
nabeeln@virginia.edu

Bradford Campbell
University of Virginia
Charlottesville, VA, USA
bradjc@virginia.edu

ABSTRACT

Edge computing for the Internet of Things prescribes executing applications on server machines closer to devices rather than depending on the cloud. However, server machines are expensive, are not flexible to adapt to varying application requirements, require gateways to interact with IoT devices, and follow a centralized model which increases traffic and application latency. Special-purpose hardware for the edge is becoming increasingly sophisticated, with support for machine learning, secure enclaves etc., and this work is an attempt to leverage such hardware to cooperatively execute edge applications, rather than relying on expensive edge servers. To do so, our design relies on a distributed middleware which can seamlessly scale up with new hardware, and a task scheduler which best matches application requirements with the hardware capabilities available. We have built a prototype middleware that operates on multiple gateways in our testbed of 250 IoT devices, and we plan to further improve our platform to support more varying use cases.

CCS CONCEPTS

• **Networks → Cyber-physical networks**.

KEYWORDS

Edge Computing, Internet of Things, Cloud Computing, Task Scheduling

ACM Reference Format:
Nabeel Nasir and Bradford Campbell. 2021. Poster Abstract: Enabling Elasticity on the Edge using Heterogeneous Gateways. In *The 19th ACM Conference on Embedded Networked Sensor Systems (SenSys'21), November 15–17, 2021, Coimbra, Portugal.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3485730.3492890

1 INTRODUCTION

The Internet of Things (IoT) is growing rapidly and is expected to have 41 billion connected devices by 2025 [2]. Handling data at such massive scales require solutions like edge computing, which prescribes shifting applications closer to the devices than on the cloud. Current edge computing solutions use server machines deployed in the local network to execute applications which interact with IoT devices on the network. The applications can still utilize the cloud, but only do so if necessary, usually for long term storage, computation heavy tasks etc. This tiered architecture has been utilized for a plethora of applications like AR/VR, traffic monitoring, smart irrigation etc.

However, relying on server machines for edge computing has its downsides. First, edge servers are expensive, usually costing upwards of $1000 even with minimal specifications. Second, provisioning a server to match edge application requirements is hard, since the edge application spectrum is vast, from small-scale sensing and actuation apps to supporting rich interactive streams for VR devices. Unlike the cloud, edge servers are not elastic resources, and a "one size fits all" server leads to resources being underutilized while being expensive, whereas under provisioning necessitates future upgrades leading to wasted resources. Third, they don't have the requisite wireless radios to communicate with IoT devices, and require additional IoT gateways to do so. Fourth, requiring all data to be brought from gateways to a central server increases network traffic and application latency.

Our work investigates the possibility of supporting edge computing using multiple inexpensive IoT gateways (single board computers), instead of depending on expensive servers. This is motivated by the rise in specialized edge hardware, which we believe can support the heterogeneous requirements of edge applications. Boards like the Raspberry Pi [3] are becoming increasingly performant, others offer GPU support on the edge [1], and some provide secure code enclaves [5]. These gateways also have the wireless radios to communicate with devices, and executing applications locally on the gateways would reduce traffic and further improve latency. What is missing is an edge platform that can leverage such heterogeneous gateways to cooperate together to execute applications, instead of limiting their usage to only ad-hoc solutions.

The key piece in building such a platform is a middleware that operates on the gateways to easily scale up when adding new gateways, and also maximizes utilization of the limited resources that are available. The central idea is to enable edge computing on as minimal a hardware as a single gateway, and allow scaling up by adding more gateways to handle the workload, or adding specialized gateways to support more complex requirements. The middleware has to identify the requirements of the applications and match them with the available capabilities of the gateways, while maximizing the overall utilization of the platform. Prior works on task scheduling either use homogeneous hardware [4], or [6] require information on execution times and deadlines of tasks, making it impractical for real world use cases.

SenSys'21, November 15–17, 2021, Coimbra, Portugal

ACM ISBN 978-1-4503-9097-2/21/11.
https://doi.org/10.1145/3485730.3492890

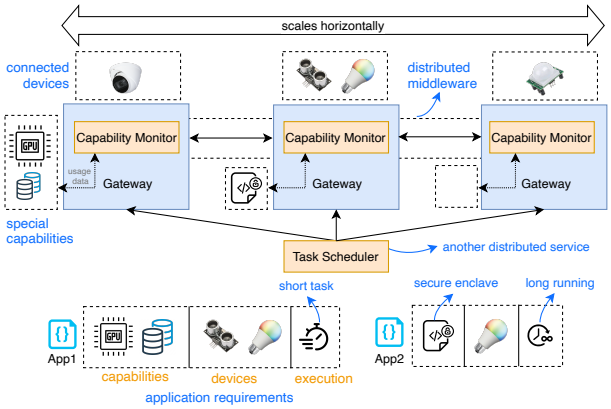

Figure 1: Task scheduler matches apps of diverse requirements with gateways that offer heterogeneous capabilities.

2 PROPOSED APPROACH

In this section, we describe the design of the middleware and task scheduler used for our edge platform.

2.1 Middleware Design

These are the main characteristics of our middleware:
Distributed: It operates without a central node having to orchestrate the platform. This is achieved by distributing services across the gateways, and also provides useful services like distributed storage.
Scalable: It allows seamless addition or removal of gateways, by allowing gateway discovery using an out-of-band wireless radio.
Supports IoT devices: It supports software modules to obtain sensor data from and send control messages to IoT devices.
Supports applications: Developers use simple abstractions to interact with IoT devices, and can deploy applications on the platform.
Monitors capabilities: It runs a capability monitor which identifies capabilities of gateways, and periodically monitors their resource usages. This allows the scheduler to make more informed decisions.
Resilient: Each application is to be spawned in a lightweight container on whichever gateway it is run on. If a gateway fails, the application can be re-spawned on another gateway.

2.2 Task Scheduler Design

The task scheduler uses application requirements and gateway resources usages to make decisions (Figure 1). When deploying an application, these requirements are collected: 1) *execution time*: whether it's a short-term task or an indefinitely running task, 2) *device requirements*: which IoT devices the application needs to interact with (to schedule the app as close to the devices as possible), 3) *special capabilities required*: GPU, secure enclave, long-term storage, etc. The capability monitor learns capabilities of gateways when they are discovered. It also periodically checks the resource usages (CPU load, available memory, storage, GPU usage, connected devices etc.) of gateways and informs the scheduler. This service also aids in deciding if an application is eating up too much resources, in which case it can be transferred to a different gateway.

The optimization function for the scheduler is still being developed, but is based on the following requirements: i) don't under provision: minimize executing applications on gateways where they will under perform, ii) don't over provision: ensure that gateways with specialized capabilities don't execute applications which don't require those capabilities, iii) ensure minimal transfer of applications from one gateway to another.

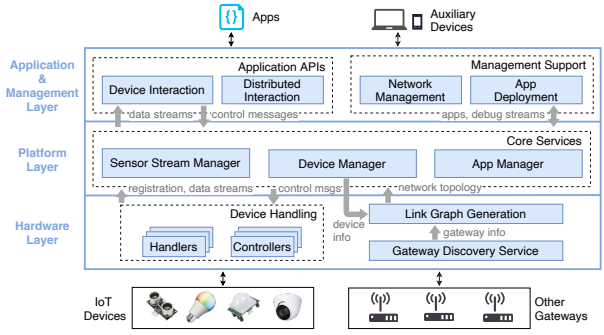

Figure 2: Middleware that runs on each gateway

We plan to evaluate this system using the following experiments: i) performance comparison of some diverse edge applications on gateways versus on an edge server, ii) comparing our task scheduler's performance versus other baselines.

3 PRELIMINARY RESULTS

We have designed a middleware that currently works for homogeneous gateway hardware, but only accommodates limited application requirements (Figure 2). We have successfully deployed this middleware design on multiple Raspberry Pi 4B [3] boards deployed in a floor of our building, supporting around 250 IoT devices. We have also built a simulator to test out different task deployment strategies to design an optimal heuristic for our task scheduler. We plan to converge on a deployment strategy that would satisfy maximum application requirements, ensure gateways are not overloaded, and increase overall utilization of the platform.

4 CONCLUSION

Our work attempts to democratize edge computing by using inexpensive edge hardware which can be scaled up to satisfy application requirements, without requiring server machines. We discussed the design of the middleware and a task scheduler that are key in building an edge platform which can operate independent of the cloud, or offer optimized task deployment for other edge platforms.

REFERENCES

[1] NVIDIA Corporation. 2021. *Jetson Xavier NX Developer Kit.* Retrieved Mar 20, 2021 from https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit
[2] Larry Dignan. 2019. *IDC - IoT devices.* https://www.zdnet.com/article/iot-devices-to-generate-79-4zb-of-data-in-2025-says-idc/
[3] The Raspberry Pi Foundation. 2021. *Raspberry Pi 4.* Retrieved Mar 20, 2021 from https://www.raspberrypi.org/products/raspberry-pi-4-model-b
[4] Karim Habak, Mostafa Ammar, Khaled A Harras, and Ellen Zegura. 2015. Femto clouds: Leveraging mobile devices to provide cloud service at the edge. In *2015 IEEE 8th international conference on cloud computing.* IEEE, 9–16.
[5] NXP Semiconductors. 2021. *EdgeLock Secure Enclave.* https://www.nxp.com/products/product-information/nxp-product-programs/edgelock-secure-enclave
[6] Daniel (Yue) Zhang, Tahmid Rashid, Xukun Li, Nathan Vance, and Dong Wang. 2019. HeteroEdge: Taming the Heterogeneity of Edge Computing System in Social Sensing *(IoTDI '19).* ACM, New York, NY, USA. http://doi.acm.org/10.1145/3302505.3310067