TSTSS: A Time-Sensitive Task Scheduling System for Multi-modal Industrial Internet of Things

Xikai Sun
Department of Automation,
Tsinghua University
Beijing, China
sxk18@mails.tsinghua.edu.cn

Xu Wang
Global Innovation Exchange,
Tsinghua University
Beijing, China
xu_wang@tsinghua.edu.cn

Fan Dang
Global Innovation Exchange,
Tsinghua University
Beijing, China
dangfan@tsinghua.edu.cn

ABSTRACT

We propose a task scheduling system for Multi-modal Industrial Internet of Things (IIoT). The system is based on the improvement of Kubernetes and the parsing of task. Furthermore, it can dynamically select the appropriate nodes to parallelly process sub-tasks according to theirs latency requirement and real-time communication and computing conditions. It can effectively solve the impact of latency sensitivity differences on task scheduling in IIoT.

CCS CONCEPTS

• **Networks** → **Network management**.

KEYWORDS

Multi-modal Industrial Internet of Things, task scheduling, collaborative edge computing

ACM Reference Format:
Xikai Sun, Xu Wang, and Fan Dang. 2023. TSTSS: A Time-Sensitive Task Scheduling System for Multi-modal Industrial Internet of Things. In *ACM Turing Award Celebration Conference 2023 (ACM TURC '23), July 28–30, 2023, Wuhan, China.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3603165.3607381

1 INTRODUCTION

With the development of technologies such as Cloud Computing and Internet of Things, multi-modal Industrial Internet of Things (IIoT) is widely applied in recent years. Generally, for the computing tasks generated in IIoT applications, some simple tasks are completed locally, while complex tasks are often transmitted to the cloud servers. However, the local computing mode will cause problems such as waste of computing resources and task congestion. The burden of data transmission in cloud computing can significantly increase the latency of computing tasks, and may cause problems like privacy leakage. Computational requirements in the IIoT, such as low latency, computational offloading, make collaborative edge computing (CEC) a new computing trend in IIoT applications. CEC reasonably allocates tasks generated in networks to other local edge devices to reduce network transmission latency, and ensures full utilization of computing resources as much as possible through task scheduling.

In the IIoT, different devices may use different APIs and data transmission protocols, bringing challenges to CEC. Through applying container technology, the computing resources of devices are virtualized, which facilitates the unified arrangement of tasks among heterogeneous devices and the unified scheduling of computing resources. Kubernetes has become the mainstream container system standard, which can effectively guarantee the resource allocation of working nodes. However, Kubernetes is not designed for the IIoT, ignoring the performance requirements such as the latency of tasks and the bandwidth of edge devices.

In this paper, we propose a Time-Sensitive Task Scheduling System (TSTSS) for the IIoT. By modularizing the task program and introducing network condition analysis to the Kubernetes, TSTSS can realize the parallel processing of tasks in the IIoT and the scheduling of tasks with latency constraints, ensuring the deterministic latency of the IIoT and the efficient using of computing resources.

2 THE DESIGN OF TSTSS

We describe the design of TSTSS in this section. As shown in Fig. 1, the entire system is mainly composed of the master server and the IIoT network managed by this server.

Tasks generated by networks are mainly divided into two categories: PLC control tasks and data processing tasks. The former is the tasks generated by PLC virtualization on the CaaS switches [2], and the results will be used to directly control the devices connected to the switches. Such tasks usually require low latency to ensure the managed devices to respond quickly. According to latency requirements, data processing tasks can be divided into time-sensitive data processing tasks and time-insensitive data processing tasks. In order to accelerate to process tasks, we adopt the data flow

ACM TURC '23, July 28–30, 2023, Wuhan, China
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0233-4/23/07.
https://doi.org/10.1145/3603165.3607381

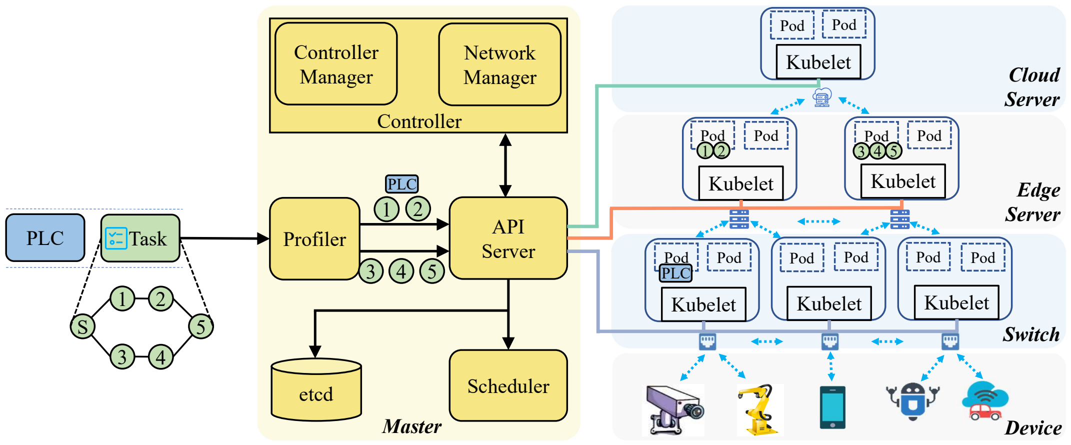

Figure 1: TSTSS overview

programming model [3], so that a task can be parsed into multiple sub-task functional modules, which are conveniently scheduled into different nodes for parallel calculation.

In the master server, we have extended the modules of the Kerbunetes [3], so as to achieve parsing tasks and efficient scheduling of network computing resource under latency constraints. The functional modules included in the master server are as follows. *Profiler* parses the input task into one or more sub-tasks, evaluates their computing requirements, and generates task configuration files to support task scheduling and result aggregation. *API Server* is the entrance of the master server to manage the entire network, and supports the communication between internal modules. *Etcd* is a database that stores nodes' communication and computing resource conditions. *Scheduler* executes the scheduling strategy for each sub-task, determine the node, routing path, and communication port to execute the sub-task. *Controller* consists of controller manager and network manager. The former interacts with Kubelets in the network to determine the computing status and resource allocation of nodes. The latter will periodically measure the communication quality of each node in the network, such as latency and bandwidth.

For each sub-task parsed by the profiler, the task scheduling strategy is executed on the Master: (1) Estimate the time sensitivity of the sub-task. If the sub-task is time-insensitive, it will preferentially assigned to cloud servers or edge servers with normal communication quality but rich computing resources, so as to provide as many choices as possible for time-sensitive sub-tasks. (2) For time-sensitive tasks, its sub-tasks are still time-sensitive. According to the computing and latency requirements of the sub-task, all nodes that meet the requirements are filtered from the etcd as candidate nodes. (3) In order to balance computing resources, the etcd models resource parameters of each candidate node and the sub-task requirement. [1] The scheduler will choose the node with the closest resource parameters to process the sub-task. (4) Regardless of whether the task is time-sensitive or not, after being assigned a sub-task to a node, the node will be recorded in the task configuration file, and its computing resources and network conditions will be updated in the etcd.

In the IIoT network, CaaS switches, edge servers, and cloud servers will be containerized by their Kubelets. After being assigned a sub-task, a corresponding pot is created for calculation. Under the guidance of the task configuration file, if a sub-task has predecessor sub-tasks, the node will continue to monitor until it receives the calculation result of the predecessor sub-tasks; the results of predecessor sub-tasks will be forwarded to nodes with successor sub-tasks.

3 CONCLUSION

In this paper, we propose a time-sensitive task scheduling system for IIoT. It can effectively solve the impact of time sensitivity differences on task scheduling in the IIoT network. We introduced the architecture of TSTSS, which is based on the improvement of Kubernetes. TSTSS can dynamically select the appropriate nodes to parallelly process sub-tasks according to the theirs time sensitivity and current communication and computing conditions. TSTSS provides a scalable and efficient solution for the integration of computing resources in industrial networks.

ACKNOWLEDGMENTS

This work is supported in part by the National Key Research Plan under grant No. 2021YFB2900100, the NSFC under grant No.62202263.

REFERENCES

[1] Meng-Yo Tsai et al. 2019. Crucial-Resource Scheduling Strategy in Edge Computing. In *2019 ICEA*. 146–150.
[2] Zheng Yang et al. [n. d.]. CaaS: Enabling Control-as-a-Service for Time-Sensitive Networking. ([n. d.]).
[3] Mingjin Zhang et al. 2022. ENTS: An Edge-native Task Scheduling System for Collaborative Edge Computing. In *2022 SEC*. 149–161.