마감기한이 존재하는 일들에 대한 스케줄링에 관한 연구

Ngoc-Hung Nguyen, Kangyu Gao, 전상운
한양대학교

{ngochungng, gaokangyu, sangwoonjeon}@hanyang.ac.kr

Job Scheduling with Deadline Constraints

Ngoc-Hung Nguyen, Kangyu Gao, and Sang-Woon Jeon
Hanyang University

Abstract

We consider a mobile edge computing (MEC) server system dealing with several tasks. Each task is given with a hard deadline constraint. To achieve high service ratios, previous works have been proposed several solutions such as traditional job scheduling, deep learning, or meta-heuristics. In this paper, we propose an optimal job scheduling algorithm that maximize the service ratio when a set of tasks are given.

Ⅰ. Introduction

Owing to the rapid development of the Internet of Things (IoT) and the sixth generation (6G) cellular networks, a lot of devices are connected to a cloud or center server for offloading their tasks. To satisfy requirements of the quality of services, mobile edge computing (MEC) has been actively studied is an emerging solution [1], [2], [3]. Because of the vast number of tasks and their stringent requirements on MEC servers, the number of served tasks is a crucial consideration. Therefore, it is necessary to design an efficient algorithm that schedules tasks with a high service ratio. Note that the service ratio is the number of served tasks per total number of given tasks on a server [4].

The traditional methods, such as earlier deadline first (EDF), and smallest data size first (SDF), were analyzed in [4]. From the analyzed result, the job scheduling algorithm called D*S was proposed as an improved solution for EDF and SDF. More recently, reinforcement learning has been also considered to provide online adaptive solution for various network scenarios [5], [6].

In this paper, we propose a job scheduling algorithm with deadline constraints. The proposed algorithm improves the efficiency of the previous algorithms and optimal under deadline constraints.

Ⅱ. Model and Method

We consider a job scheduling consisting of a single server and N tasks. Let f [CPU cycles/sec] be the computation capability of the server. Each task i is denoted by T_i = (α_i, β_i), where $i \in [1{:}N]$. Here, α_i [CPU cycles] is the amount of CPU cycles required for T_i, and β_i [sec] is the deadline for completing T_i. Let $\mathbf{s} = [s(1), s(2), \ldots, s(N)]$ be the order set of $[1{:}N]$. We further define the collection of all possible ordered sets of $[1{:}N]$ as $\boldsymbol{\mathcal{S}}$. That is, $|\boldsymbol{\mathcal{S}}| = N!$.

Denote the completion time of task i by t_i. For given $\mathbf{s} \in \boldsymbol{\mathcal{S}}$, the server will compute N tasks based on the order of $[T_{S(1)}, T_{S(2)}, \ldots, T_{S(N)}]$. Hence, the completion time of $T_{S(i)}$ is given by: $t_{s(i)} = \sum_{j=1}^{i} \frac{\alpha_i}{f}$, where $i \in [1{:}N]$. Note that outage occurs for $T_{S(i)}$ if $t_{s(i)} > \beta_{s(i)}$. In this paper, we focus on outage minimization, which is represented by the following optimization:

$$\min_{\mathbf{s}=\{s(1),s(2),\cdots,s(N)\}\in s(N)} \sum_{i=1}^{N} \mathbb{1}(t_{s(i)} > \beta_{s(i)}).$$

The proposed algorithm traverses the tasks from the smallest required cycles and schedules them with the corresponding deadline. Each traversed time, the algorithm checks whether the task satisfies the deadline constraint. If the task satisfies the deadline constraint, it adds the scheduled tasks list. The complexity of the algorithm in the worst case is $O(N^2)$.

III. Numerical Results

To evaluate the performance of the proposed algorithm, we simulate it with the algorithm mentioned in [4]. The measurement metric is the service ratio, the parameter is shown in Table I.

Table I

Simulation Parameters

Parameter	**Value**
Number of required cycles	50-200 (cycles)
Deadline for completing	280-1125 (sec)
Computation capacity	1-8 (cycles/sec)

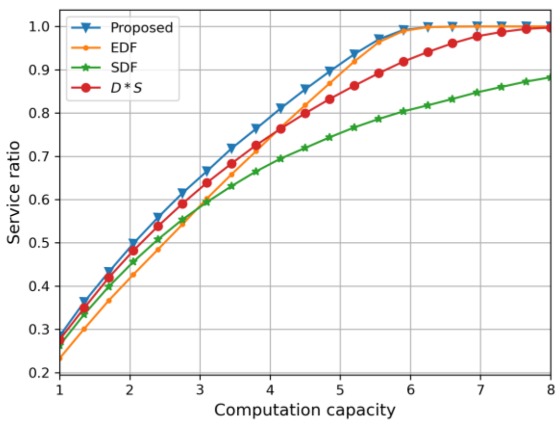

Fig. 1. Service ratios achievable by different algorithms.

As depicted in Fig. 1, the proposed algorithm shows the best performance. When the computation capacity changes from 1-8, the service ratios increase, and the outcome of the proposed algorithm is the upper bound of the other algorithms. These results obtain because the proposed algorithm considers both information on required cycles and deadlines, while the other algorithms focus only on the minimum required cycles, deadlines, or D*S value.

IV. Conclusion

In this paper, we considered the problem of job scheduling with hard deadline constraints. Numerical results showed that the proposed algorithm outperform the conventional algorithms in the considered scenario.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2020R1C1C1013806).

REFERENCES

[1] S. Dang, O. Amin, B. Shihada, and M.-S. Alouini, "What should 6G be?" Nature Electronics, vol. 3, no. 1, pp. 20–29, 2020.

[2] B. Ji, Y. Wang, K. Song, C. Li, H. Wen, V. G. Menon, and S. Mumtaz, "A survey of computational intelligence for 6G: Key technologies, applications and trends," IEEE Transactions on Industrial Informatics, vol. 17, no. 10, pp. 7145-7154, 2021.

[3] M. Sohaib and S. -W. Jeon, "Hybrid online-offline learning for task offloading in mobile edge computing systems," submitted to IEEE International Conference on Communications (ICC), 2023.

[4] Y. Zhang, J. Zhao, and G. Cao, "Service scheduling of vehicle-roadside data access," Mobile Networks and Applications, vol. 15, no. 1, pp. 83-96, 2010.

[5] M. Sohaib, J. Jeong and S. -W. Jeon, "Dynamic Multichannel Access via Multi-Agent Reinforcement Learning: Throughput and Fairness Guarantees," IEEE Transactions on Wireless Communications, vol. 21, no. 6, pp. 3994-4008, June 2022.

[6] J. Jeong, S. H. Lim, Y. Song and S. -W. Jeon, "Online Learning for Joint Beam Tracking and Pattern Optimization in Massive MIMO Systems," IEEE Conference on Computer Communications (Infocom), 2020, pp. 764-773.