Peer-to-Peer Networking and Applications (2020) 13:1830–1838
https://doi.org/10.1007/s12083-020-00934-1

Performance analysis of edge-PLCs enabled industrial Internet of things

Yanjun Peng[1] · Peng Liu[1] · Tingting Fu[1]

Received: 23 November 2019 / Accepted: 14 May 2020 / Published online: 19 June 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract
With the recent advancement in Industrial Internet of Things (IIoT), general programmable logic controllers (PLCs) have been playing more and more critical roles in industrial control systems (ICSs), such as providing local data processing, decentralized control and fault diagnosis. These so called edge-PLCs, directly receive the raw data from sensors embedded in factory equipments, put them into predefined memory space and perform analysis using programs such as the ladder logic. The challenge is how to allocate blocks in the fixed-size memory to different sensors so as to match irregular data flows. In this paper, we try to conduct performance analysis of different partition instances of the memory in the edge-PLC by modeling this problem as a multiple single-server queueing systems. We assume every sensing flow is independent of each other and has its dedicated processer. Changes can be made to partition instances to adapt to the external environment, such as the rising of order numbers or product category switching. Each state of the environment is defined by the finite state Markov chain and arrival of sensing data flows follow the stationary Poisson process. The data in the queue will expire after staying in the memory for a while. The duration of availability and service is modeled as the exponential distribution. The performance measured under different system states are analyzed in the simulation.

Keywords Industrial Internet of things · Edge-PLC · Performance analysis · Queuing system

1 Introduction

By integrating various sensors, mobile communications, intelligent analysis and other technologies into every link of industrial production process, industrial Internet of things (IIoT) [3] has enabled industrial control systems with vast sensing and monitoring capabilities, greatly improved manufacturing efficiency and product quality while reducing resource consumptions and manufacturing costs. In the meantime, intelligence also means more sensing data, deep calculation and fast response, which pushes computational-intensive tasks close to end devices and puts heavy burden on the computation and communication requirements.

This article belongs to the Topical Collection: *Special Issue on Emerging Trends on Data Analytics at the Network Edge*
Guest Editors: Deyu Zhang, Geyong Min, and Mianxiong Dong

✉ Peng Liu
perryliu@hdu.edu.cn

[1] Institute of Industrial Internet, School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China

Edge computing (EC) [2, 16, 19] has emerged as a new paradigm in **IIoT** by placing computing services close to the physical location of either the user or the source of the data, which provides faster, more reliable services and enables the design of high-performance and adaptive system. **In such case**, computational tasks and data traffic no longer need to be uploaded to the cloud server and travel through the unpredictable Internet. In the meantime, some privacy and security issues can be avoided, if the cloud is not trustworthy [1, 12, 17]. Therefore, EC enabled IIoT will potentially push the traditional industry to usher into a new stage of intelligence.

A PLC is an electronic device designed for digital operations and is specially designed for industrial production. It contains a programmable memory, which is used to store the internal logic program. The logic program performs logical operation, sequence control, timing, counting, arithmetic operations and other user-oriented instructions. It also controls various types of machinery or production processes through its built-in digital or analog input/output. Therefore, PLC is the essential part of the entire industrial control. The enhancement of communication abilities and the development of man-machine interface technologies with IoT,

www.manaraa.com

turn PLCs into a new form, edge-PLCs, which are **easier** to adapt to various control systems. At the meantime, they also bring new requirements and challenges for the technical implementations.

In this paper, we consider a scenario of edge-PLCs enabled industrial control system in a smart factory. These edge-PLCs are distributed **close to** factory equipments, reading data from sensors and putting them into pre-allocated memory space for processing. Once the allocation is determined, it is difficult to expand or re-allocate **during** operation. Another fact is that most of the smart factories support Flexible Manufacturing System, which means the data flow from sensors may vary along the time scale. Therefore, **there exists** the challenge **of** allocating blocks in the fixed-size memory to different sensors in order to match irregular data flows and maximize the system performance. To achieve this goal, we are going to conduct performance analysis **on** memory partition instances in the edge-PLC by modeling this problem as a multiple single-server queueing systems. We can find the optimal solution under different system settings for the corresponding case. Finally, we test our work on the synthetic data set with two kinds of sensors under dual system parameters. The simulation shows the feasibility of the proposed scheme. The key contributions of this work are:

1) According to our best knowledge, we are the first to identify the problem of memory allocation for edge-PLCs in smart IIoT, **which aims to** maximize the system performance under memory space constraints.
2) To accurately restore the practical scenario, we model each state of the environment as the finite state Markov chain and assume that arrival of data flows follow the stationary Poisson process.
3) We carry out the performance analysis and the synthetic simulation, regarding the system as a multiple single-server queue.

The rest of the paper is organized as follows. Section 2 discusses data analysis, edge computing and queuing theory related to IIoT. Section 3 proposes the mathematical model of the system and targeted problem. In Section 4, the system state is defined as a Markov chain, along with the calculation of its transition matrix and stationary distribution. The performance measures and synthetic data-set based simulation are conducted and presented in Section 5. Section 6 finally concludes the paper.

2 Related work

In IIoT, various sensors are employed in smart factories [11], collecting large amounts of data from factory machinery in a real-time manner. Besides data forwarding [18], data and service analysis [23] is another core means to obtain important information. By doing so, we can optimize the operation process of the factory, such as determining the maintenance plan of the equipment in advance, or repairing the malfunctioning equipment in time. In addition, other useful information such as the demand for a certain product, can be obtained.

Recently, many industrial automation applications have involved the use of sensors that transfer raw data to programmable logic controllers (PLCs), where those collected sensor data will be stored in a capacity constrained memory first and then processed [13][21]. Reference [5] introduces a new platform that supports efficient analysis of big data collected in smart factories by using 5G wireless network, fog node and cloud computing. Sensors with low price and good performance are the cornerstone of the application of industrial Internet of things. Reference [14] shows how production machines can be enabled for predictive maintenance by retrofitting with low-cost sensors, with IIoT and machine learning.

To process data, some previous studies have combined IIoT with cloud computing. An enabled framework for health monitoring is proposed in [6]. The paper presented a Health IIoT-enabled monitoring framework, where ECG and other healthcare data are collected by mobile devices and sensors, which are then securely sent to the cloud for seamless access by healthcare professionals. In [8], the authors propose an analytical model of a system with both cloud servers and edge nodes, and use it to show how to reduce the cost for computing resources while ensuring performance constraints for a healthcare monitoring system.

Most systems in IIoT will face the challenge of constrained resources, in which the trade-off between computation and communication costs is often evaluated [10]. Computational task offloading is one of the important research topic for resource management [22, 24], such as in vehicular networks [20]. Some paper has addressed the resource allocation problem in mobile edge computing enabled industrial Internet of things with constrained computing power, which adopts a two double auction schemes with dynamic pricing to determine the matched pairs between IIoT MDs and edge servers [15].

Queuing theory is often used in performance analysis and optimization which is a very important step of resource planning and management. Some literatures have used queuing theory for performance analysis in IIoT. The problem of performance evaluation of a wireless sensor node with energy harvesting is solved in [9]. The paper formulated a single-server queueing system with two finite buffers in which the flow of customers and energy units arrive. Following [4], a novel multi-server queue with heterogeneous customers is formulated and analyzed to model the operation of a cell based mobile

www.manaraa.com

communication network. In our paper, we consider dividing the space of an edge-PLC memory into several memory allocation instances, and conducting performance analysis by modeling the problem as several independent single-server queues where data in the queue may be dropped due to timeout.

3 Mathematical model

Suppose there is a manufacturing system with single edge-PLC as the access point of multiple types of sensors. The edge-PLC is equipped with a load memory and work memory with limited capacity due to cost control. The perceived data from sensors of the same category will be stored in a pre-allocated area of the work memory. The logic to process those data will be put into the pre-allocated area of the load memory. Since the load memory is usually implemented using nonvolatile memory (NVM), hence it is inconvenient to alter the allocation frequently. **The goal of this paper** is to analyze the performance of allocation instances under different system characteristics with the aim to maximize the **global efficiency**.

As shown in Fig. 1, without loss of generality, we abstract two memories into a so called executing storage, which is divided into several memory blocks of the same size. Assume there are m types of sensor groups and each individual sensor group will be assigned with **certain** number of blocks (marked as yellow, blue and red) which can only store and process the corresponding type of data. The sum of all dedicated memory blocks cannot exceed the total storage space of the edge-PLC. Furthermore, it is not necessary to use up the executing storage, but **allow** some space being left unused. It is worth noting that each memory block has a limited capacity and data will be dropped if the amount exceeds the capacity. Once the allocation is determined, the dynamic expansion of storage space is not allowed.

In this paper, we assume that the arrival rate of each type of data varies and the executing processes is also independent from each other. Furthermore, even for the same type of data, its arrival rate would change in different scenario. We also assume that there are sperate processing units (i.e., $\mathbb{S}_1, \mathbb{S}_2, ..., \mathbb{S}_m$) to handle different types of data independently, and their computing capacity is disparate as well. The memory blocks occupied by each type of data will not be released until it has been processed. Therefore, different partitioning instance of the executing storage will result in variant performance outcome facing practical situations.

The basic rule of processing data is First-Come-First-Serve. Upon the arriving of any data, if the corresponding processing unit $\mathbb{S}_i$ is free, it will be assigned to $\mathbb{S}_i$. In the case that the unit $\mathbb{S}_i$ is already occupied by the same kind of data, the system will try to store the data into the memory blocks, providing there is space available. However, the incoming data will be dropped if there is no space in the designated memory blocks.

For simplicity, we consider the scenario where there are two data types, i.e. $m = 2$, named Tp_1 and Tp_2 respectively as shown in Fig. 1. This assumption can be easily extended to the case of multiple data types. The arrival data flows of the two data types behave as the stationary Poisson process with intensities λ_1 and λ_2 respectively. Meanwhile, the service time of the two types of data is exponentially distributed with parameter μ_1, $\mu_1 \geq 0$ and μ_2, $\mu_2 \geq 0$. Furthermore, the data in the memory is assumed to be time sensitive, which means it is only valid within a variational period of time. Data will leave the buffer after expiration. It is public recognition that the process follows the exponentially distribution with the parameter α, $\alpha \geq 0$. It is assumed that this parameter fits all types of data.

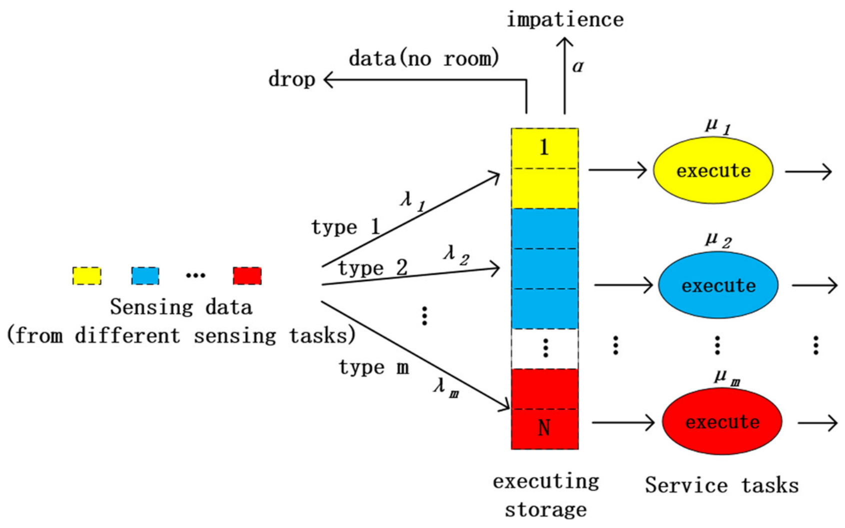


Fig. 1 Demonstration of the system model

المنارة للاستشارات
www.manaraa.com

The values of those parameters could change in different scenario, as the manufacturing requests and merchandise categories change from time to time. As a result, for each type of data, we now model it as the queuing system of the $M/M/1/n$ type, in which n is the amount of memory blocks allocated to this type of data.

According to above facts and assumptions, once the scenario switches to another state, parameters of the queuing model will change as well. For instance, if the data arrival rate λ_1 rises from intensity 1 to 5, the number of memory blocks allocated to the first data type should be also increased in order to reduce the potential data loss. Our goal is to analyze the performance measures of the proposed model under the memory storage constraint, to obtain various possible memory partitioning instances in different scenario, and to consider the relevant optimization problem.

4 Stationary distribution of the system states

Let n_1 and n_2 represent the number of memory blocks allocated to data type Tp_1 and Tp_2 respectively, and N is the total size of the executing storage in the edge-PLC, we have $n_1 + n_2 \leq N$. The system described above is modeled by a regular irreducible continuous time Markov chain ϕ_t as shown below: ϕ_t={i_t, j_t, s_t}, $t \geq 0$, in which, during the epoch t,

- t, stands for the time variable;
- i_t, $i_t \in [0, n_1 + 1]$, is the amount of data of type Tp_1;
- j_t, $j_t \in [0, n_2 + 1]$, is the amount of data of type Tp_2;
- s_t, $s_t \in \{0, 1, 2, 3\}$, is the state of the server: s = 0 means all processing units are idle, s = 1 means only the processing unit $\mathbb{S}_1$ is working, s = 2 means only the processing unit $\mathbb{S}_2$ is working, s = 3 means all processing units are working.

We formulate the Markov chain as the following state space:

$$\left(\{0,0,0\}\right) \bigcup \left(\{i,0,1\}, i \in [1, n_1+1]\right)$$
$$\bigcup \left(\{0,j,2\}, j \in [1, n_2+1]\right)$$
$$\bigcup \left(\{i,j,3\}, i \in [1, n_2+1], j \in [1, n_2+1]\right)$$

Then, we can obtain the stationary probabilities of the system states according to the properties of Markov chains, e.g., the Markov chain ϕ_t is irreducible and has the finite state space : $\pi(i, j, s) = \lim_{t\to\infty} P\{i_t = i, j_t = j, s_t = s\}$, $i \in [0, n_1 + 1]$, $j \in [0, n_2 + 1]$, $s \in \{0, 1, 2, 3\}$.

The arrival flow of each type of data conforms to the distribution of different parameters, and after allocating the storage space, all types of data are processed independently from each other. Therefore, we treat each type of data as a separate queue, and the arrival and processing of each type of data as a separate single-server queuing model problem. Let us first consider service process analysis for one of the two types of data.

We use the value of i to describe the macroscopic state of the Markov chain, and add the values of j and s to these macroscopic states to describe the specific state of the system. Then we form the row vectors π_i as follows:

$$\pi_0 = (\pi(0,0,0), \pi(0,1,2), \pi(0,2,2), ..., \pi(0,n_2,2))$$

$$\pi_i = (\pi(i,0,1), \pi(i,1,3), \pi(i,2,3), ..., \pi(i,n_2,3)),$$
$$i \in [1, n_1+1].$$

The notations used in this paper are listed below:

- I stands for the identity matrix and O represents an empty matrix of appropriate dimension;
- V_c^1 is a column vector consisting of all 1's, and V_r^0 denotes a zero row vector;
- diag $\{A_1, ..., A_l\}$ is the block diagonal matrix with the diagonal blocks $A_1, ..., A_l$;

According to the properties of Markov chains, we know that the stationary probabilities vectors in Markov chains satisfy the following Chapman-Kolmogorov formula. If we know the infinitesimal generator of Markov chains, then we can calculate the stationary probabilities vectors according to this formula.

$$(\pi_0, \pi_1, ..., \pi_{n1})Q = V_r^0$$

$$(\pi_0, \pi_1, ..., \pi_{n1})V_c^1 = 1$$

where Q is the infinitesimal generator of the Markov chain $\phi_t, t \geq 0$.

According to the formula mentioned above, in order to obtain the stationary probabilities vector of the Markov chain, we must first calculate its infinitesimal generator, which is the transition probability matrix of the Markov chain. Also, we write down an expression for the generator of the Markov chain $\phi_t, t \geq 0$.

The transfer matrix of the Markov chain $\phi_t, t \geq 0$ is Q. Q is made up of Q_{i_1,i_2}, which is the transition probability from macrostate i_1 to macrostate i_2. According to the property of transfer matrix of Markov chain, the diagonal entries of the transfer matrices are negative, and the total probability of leaving state i is represented by the value $Q_{i,i}$ of the diagonal elements of the transition matrix.

We analyze the transition matrix of Markov chains and calculate the transition probability between states according to the conditions described above. The results are as follows.

المنارة للاستشارات
www.manaraa.com

Theorem 1 *The infinitesimal generator Q of the Markov chain $\phi_t, t \geq 0$, has the following block-tridiagonal structure:*

$$\begin{pmatrix} Q_{0,0} & Q_{0,1} & O & \cdots & O & O \\ Q_{1,0} & Q_{1,1} & Q_{1,2} & \cdots & O & O \\ O & Q_{2,1} & Q_{2,2} & \cdots & O & O \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ O & O & O & \cdots & Q_{n_1,n_1} & Q_{n_1,n_1+1} \\ O & O & O & \cdots & Q_{n_1+1,n_1} & Q_{n_1+1,n_1+1} \end{pmatrix}$$

The non-zero blocks $Q_{i_1,i_2}, i_1, i_2 \in [0, n_1 + 1]$, have the following form: $Q_{0,0} = diag\{-\lambda_1 I_{n_2+1}\}$, $Q_{i,i} = diag\{-(\lambda_1 + \mu_1 + (i-1)\alpha) I_{n2+1}\}$, $i \in [1, n_1]$, $Q_{n_1+1,n_1+1} = diag\{-(\mu_1 + n_1\alpha) I_{n_2+1}\}$, $Q_{i,i+1} = diag\{\lambda_1 I_{n_2+1}\}, i \in [0, n_1]$, $Q_{1,0} = diag\{\mu_1 I_{n_2+1}\}$, $Q_{i,i-1} = diag\{((i-1)\alpha + \mu_1) I_{n_2+1}\}, i \in [2, n_1 + 1]$.

Proof As mentioned above, $Q_{i,i}$ refers to the total probability that the state of the system changes from i. $Q_{0,0}$ refers to the total probability that the system leaves the state where the amount of data is 0. This change may only happen upon one data enters the system. From the previous analysis we can see that each vector π_i has $n_2 + 1$ elements, so we need to multiply the transition probability by I_{n_2+1}. And because the diagonal elements of the transition matrix are all negative, we have $Q_{0,0} = diag\{-\lambda_1 I_{n_2+1}\}$.

$Q_{i,i}, i \in [1, n_1]$ represents the probability that the system leaves the state i, i.e., the probability that the amount of data in the system will change from i. There are two possible reasons for this. One is that there is data arriving at the system, and the probability is λ_1. The other is that one data leaves the system. There are two reasons for one data to leave the system. One is that the data has been processed and actively releases the storage space. The other is that the data has been waiting too long and before getting processed, and is forced to leave the system due to impatience. When there are i data in the system, it means that there is one data being processed and $i-1$ data is waiting, so the probability of impatience caused leaving of data can be calculated as $(i-1)\alpha$, we have $Q_{i,i} = diag\{-(\lambda_1 + \mu_1 + (i-1)\alpha) I_{n2+1}\}$.

Similarly, Q_{n_1+1,n_1+1} defines the probability that the amount of data in the system will change from $n_1 + 1$. $n_1 + 1$ is the upper limit of the amount of data that the system can store, so it is impossible to accept new data to enter the system. Therefore, the change can only be caused by data leaving the system, this is similar to the case of data leaving in $Q_{i,i}$, so $Q_{n_1+1,n_1+1} = diag\{-(\mu_1 + n_1\alpha) I_{n_2+1}\}$

$Q_{i,i+1}, i \in [0, n_1]$ indicates the probability that the amount of data in the system increases by one, and we can see that this situation is caused by one data entering the system, so $Q_{i,i+1} = diag\{\lambda_1 I_{n_2+1}\}$.

$Q_{i,i-1}, i \in [2, n_1 + 1]$ indicates the probability that the amount of data in the system is reduced by one. There are two reasons for the data leaving the system, which have been explained above, so $Q_{i,i-1} = diag\{((i-1)\alpha + \mu_1) I_{n_2+1}\}$.

We find that the situation mentioned above does not include $Q_{1,0}$, which defines the probability that the amount of data in the system changes from 1 to 0. Because when there is only one data in the system, it must have been processed. In this case, the data does not need to wait for processing, then the only reason the data leaves the system is to be processed and release the storage space, so $Q_{1,0} = diag\{\mu_1 I_{n_2+1}\}$.

The transition matrix we are considering is the state transition in an infinitesimal time interval, the server can only process one data and at most one data can arrive the system in each infinitesimal time interval, the matrices $Q_{i_1,i_2}, i_1, i_2 \in [0, n_1 + 1]$, are zero matrices when $|i_1 - i_2| > 1$.

Since we treat the service processes of two types of data as two independent queues, queuing analysis for the other type of data is the same as above. □

5 Numerical and simulation results

5.1 Simulation parameters and performance metrics

In order to evaluate the validity of the proposed analysis method built on Markov chain, the simulation is carried out using synthetic data set derived from practical scenario in a smart factory. The simulation is run for 50 times with same parameters and the average of the outcome is adopted to reduce errors. To be accordance with the mathematic model in Section 3, we consider two data types in the simulation. In the rest of this section, the performance measures of the first type of data Tp_1 will be shown as an example. Meanwhile, the analysis of the second type of data Tp_2 is similar.

Before we move to the result analysis, some measurement quantities are listed below according to the calculated vectors $\pi_i, i \in [0, n_1 + 1]$.

The probability that server is busy processing the first type of data Tp_1 at an arbitrary moment is:

$$P_{busy} = \sum_{i=1}^{n_1+1} \sum_{j=0}^{n_2+1} (\pi(i, j, 1) + \pi(i, j, 3)) \tag{1}$$

The average **amount** of data being queued at an arbitrary moment is:

$$N_{queue} = (i-1) \times P_{busy} \tag{2}$$

The intensity of output flow of the first type of data Tp_1 is:

$$\lambda_{out1} = \mu_1 \times P_{busy} \tag{3}$$

Below we are going to show the performance metrics of the system according to Eqs. 1, 2 and 3.

www.manaraa.com

The loss probability of an arbitrary first type of data is:

$$P_1^{loss} = 1 - \frac{\lambda_{out1}}{\lambda_1} \tag{4}$$

Regarding the first type of data, because of overflow of memory blocks, the loss probability of an arbitrary data upon arrival is calculated as:

$$P_1^{(ent-loss)} = \sum_{j=0}^{n_2+1} (\pi(n_1+1, j, 1) + \pi(n_1+1, j, 3)) \tag{5}$$

For the first type of data, due to impatience (time out), the loss probability of an arbitrary data is calculated as:

$$P_1^{(imp-loss)} = \frac{1}{\lambda_1}(\alpha \sum_{i=2}^{n_1+1} \sum_{j=0}^{n_2+1} (\pi(i, j, 1) + \pi(i, j, 3))) \tag{6}$$

In order to compare the experimental results under different circumstances, **we consider** a scenario with two states, namely R_1 and R_2, which stand for the demand for products varying at different time. During the peak demand for the product, more machineries and equipments will be involved in the production process. Therefore, the sensors monitoring these devices sequentially collect more data. Then, the data arrival rate is higher, and the corresponding data processing rate has to be increased to match that. At other times, the demand for the product may be lower, so the requiring amount of sensing data will decrease. We set up two different sets of parameters for different states and consider that in state R_2 there is higher volume of data for both types than that in the state R_1.

5.2 Results and discussion

In the state R_1, we define the system parameters as follows: for the first type of data (Tp_1), the data arrival intensity is $\lambda_1 = 0.5$ block per minute, the service intensity is $\mu_1 = 0.8$ block per minute. For the second type of data (Tp_2), the data arrival intensity is $\lambda_2 = 0.2$ block per minute, the service intensity is $\mu_2 = 0.5$ block per minute. Here the impatience intensity is $\alpha = 0.5$ block per minute.

In the state R_2, we define the system parameters as follow: for the first type of data, the data arrival intensity is $\lambda_1 = 1$ block per minute, the service intensity is $\mu_1 = 2$ blocks per minute, for the second type of data, the data arrival intensity is $\lambda_2 = 2$ blocks per minute, the service intensity is $\mu_2 = 2.5$ blocks per minute. Here the impatience intensity is $\alpha = 1$ block per minute.

In order to compare the system performance in different circumstances with the same settings, we fix the total executing storage of the edge-PLC to 50kB, and size of the first type of data to 1 block, size of the second type of data to 2 blocks. In the experiment, we assume that processing units have sufficient processing power. Now, let us take a look at the results from the experiment for the first state R_1.

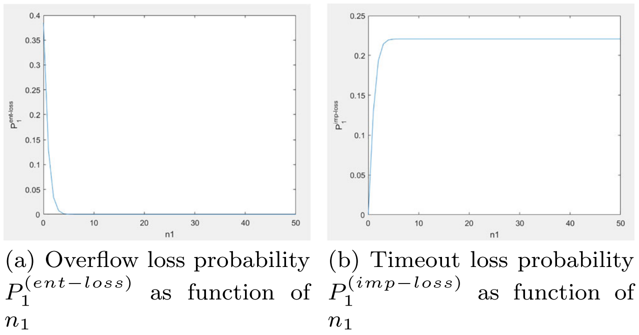

(a) Overflow loss probability $P_1^{(ent-loss)}$ as function of n_1 (b) Timeout loss probability $P_1^{(imp-loss)}$ as function of n_1

Fig. 2 Performance evaluation of the first type of sensor data under state R_1

Figure 2a and b show the loss rate $P_1^{(ent-loss)}$ of the first type of data Tp_1 due to insufficient memory space and the loss rate $P_1^{(imp-loss)}$ due to timeout in the first state R_1, respectively.

According to the figure, $P_1^{(ent-loss)}$ decreases with the increase of n_1, while $P_1^{(imp-loss)}$ increases with the increase of n_1. This is because when the allocated memory is large enough, the arrival data will have little chance to be rejected due to no storage space, and the main reason for data loss is overlength waiting time. Therefore, when n_1 is large enough, $P_1^{(imp-loss)}$ reaches a relatively stable maximal value, because there is a lot of data waiting in the queue at this time. The longer the queue is, the longer the data needs to wait, and the higher the probability of data loss due to longer waiting time. When n_1 is 0, $P_1^{(ent-loss)}$ reaches the largest value and $P_1^{(imp-loss)}$ is 0. At this point, data is lost if the same type of data is already being served when the data arrives. With the absence of waiting area, we can naturally eliminate the overtime factor.

Figure 3a and b show the loss rate $P_2^{(ent-loss)}$ of the second type of data due to insufficient memory space and the loss rate $P_2^{(imp-loss)}$ due to timeout in the first state R_1, respectively.

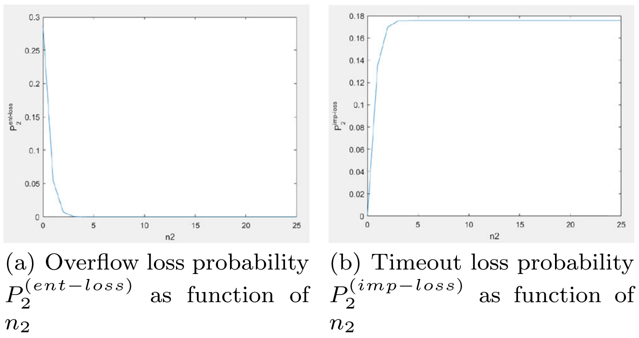
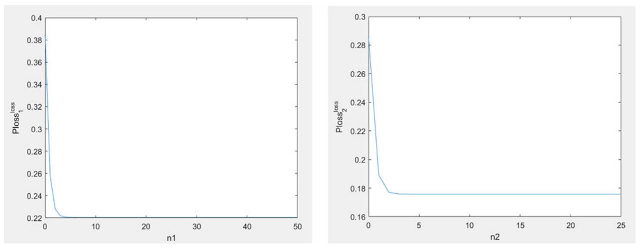

(a) Overflow loss probability $P_2^{(ent-loss)}$ as function of n_2 (b) Timeout loss probability $P_2^{(imp-loss)}$ as function of n_2

Fig. 3 Performance evaluation of the second type of sensor data under state R_1

للاستشارات المنارة
www.manaraa.com



(a) Loss probability P_1^{loss} as function of n_1 (b) Loss probability P_2^{loss} as function of n_2

Fig. 4 Comprehensive performance evaluation of two types of data under state R_1

When the size of allocated memory is zero for all types of data, $P_1^{(ent-loss)}$ is larger than $P_2^{(ent-loss)}$ because the arrival rate of the first type of data is larger than the second type of data. The maximal value of $P_1^{(imp-loss)}$ is larger than that of $P_2^{(imp-loss)}$, because the value of $\frac{\lambda_1}{\mu_1}$ is larger than $\frac{\lambda_2}{\mu_2}$. $\frac{\lambda_1}{\mu_1}$ and $\frac{\lambda_2}{\mu_2}$ are the intensity of the data service rate relative to the data arrival rate.

Figure 4a and b show the integrated loss rate of the two types of data in the first state R_1.

Obviously, the loss rate of both types of data decreases with the increase of the number of the allocated memory blocks. As one can imagine, the data loss rate is the greatest when n_1 or n_2 equals to 0, where $P_1^{loss} = 0.38461538$ and $P_2^{loss} = 0.28571428$, because any arrive data will be immediately rejected if the processing unit is busy. When n_1 and n_2 increase, the data loss rate will gradually reach the minimum value. However, the minimum value is not 0, because before getting the service, some data may reach its maximal time-to-live, TTL. Once $n_1 \geq 9$, P_1^{loss} will reach the minimal value, which is 0.22063176. If $n_2 \geq 7$, the minimal value of P_2^{loss} is 0.17580011. Even if the allocated storage space is large enough to store entire data, some may be dropped because of timeout.

To analyze the influence of different allocation instances, Fig. 5 illustrates the total loss probability of the two types of data in the state R_1. Here, we calculate the total loss probability as follow: $Ploss_{total} = P_1^{loss} + P_2^{loss}$.

We need to consider how to allocate limited memory to different types of data so that more data can be served and the system performance can be optimized. We can see from Fig. 5, if $n_1 = 0, n_2 = 0$, $Ploss_{total}$ is very high, which reaches the maximal value of 0.67032967. This result is easy to understand, once each type of data is assigned with zero memory, the data loss rate is the highest, both individually and collectively. From the figure, we can also see that the total loss rate can be minimized in many multiple combinations of n_1 and n_2, for instance, $n_1 = 15, n_2 = 12$ and $n_1 = 15, n_2 = 13$, where the minimum value is 0.39643187. As we can list all the

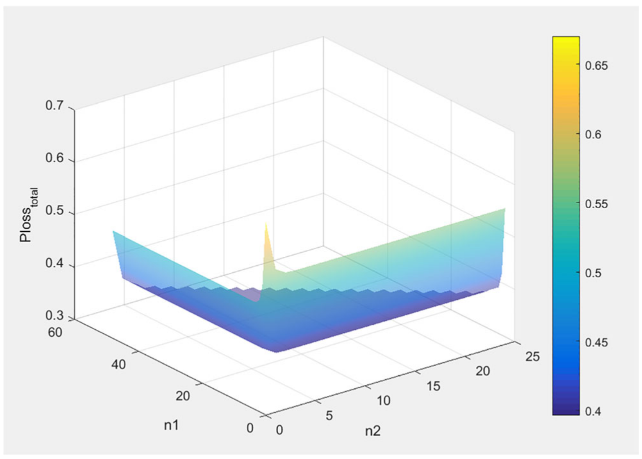


Fig. 5 Total Loss probability $Ploss_{total}$ as function of n_1 and n_2

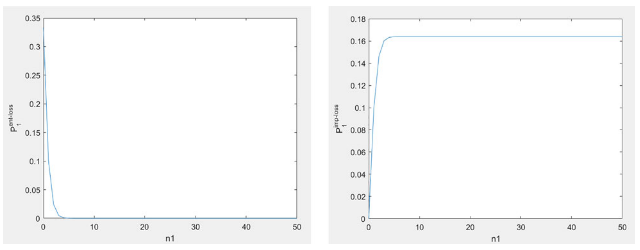
allocation instances that achieve the global optimization of the system perform, we may select the best configuration among them according to the manufacturing goal.

We regard two states as independent cases , each of them should find different allocation strategies. Then, we present the results of the experiment in the state R_2.

Figure 6a and b respectively show the loss rate $P_1^{(ent-loss)}$ due to insufficient memory space and $P_1^{(imp-loss)}$ due to timeout. We omit the analysis here because the results and conclusions of this experiment are similar to those above.

Figure 7a and b illustrate two loss rates of the second type of data in the state R_2, which are similar with the result in Fig. 6.

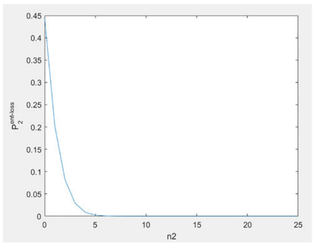
As shown in Fig. 8, the loss rate of two types of data also has extreme values, including the maximal value and the minimal value. When $n_1 = 0$, P_1^{loss} reaches the maximal value of 0.33333333. When $n_1 = 10$, P_1^{loss} reaches the minimum value of 0.16395341. When $n_2 = 0$, P_2^{loss} reaches the maximal value of 0.44444444. When $n_2 = 11$, P_2^{loss} reaches the minimum value of 0.23736939. The figure shows that the overall loss rate for the first data type is



(a) Overflow loss probability $P_1^{(ent-loss)}$ as function of n_1 (b) Timeout loss probability $P_1^{(imp-loss)}$ as function of n_1

Fig. 6 Performance evaluation of the first type of sensor data under state R_2

المنارة للاستشارات
www.manaraa.com


(a) Overflow loss probability $P_2^{(ent-loss)}$ as function of n_2

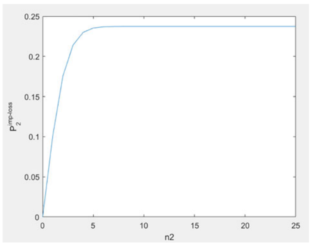
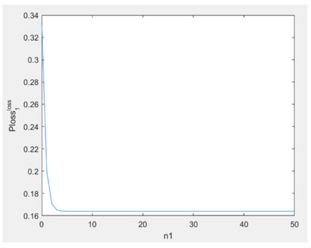

(b) Overflow loss probability $P_2^{(ent-loss)}$ as function of n_2

Fig. 7 Performance evaluation of the second type of sensor data under state R_2

greater in the state R_1 than in the state R_2 because the value of $\frac{\lambda_1}{\mu_1}$ is larger in the state R_1 than in the state R_2. The overall loss rate for the second data type is less in the state R_1 than in the state R_2 while the value of $\frac{\lambda_2}{\mu_2}$ is less in the state R_1 than in the state R_2.

Figure 9 illustrates the total loss rate of the two types of data integrated in the state R_2. The condition to achieve the maximal value of $Ploss_{total}$ is as same as that in the state R_1, i.e., $n_1 = 0, n_2 = 0$. The maximal value is 0.77777777, which is larger than that in the state R_1. There are also many allocation instances to minimize the total loss rate, such as $n_1 = 12, n_2 = 18$, and $n_1 = 12, n_2 = 19$. Then, the value of $Ploss_{total}$ is 0.40132280. We can see that in terms of the overall data loss rate of the system, the state R_2 is higher than the state R_1. Therefore, the data loss rate is not only related to how we allocate memory but also the data arrival rate and service rate. Nevertheless, we can solve the optimal memory allocation problem with system parameters n_1 and n_2 aiming at different scenarios.

According to above simulations, we find that the best allocation of memories for the target scenario is not a point but an area, which means user can incline the allocation to part of the data without affecting the total performance.


(a) Loss probability P_1^{loss} as function of n_1

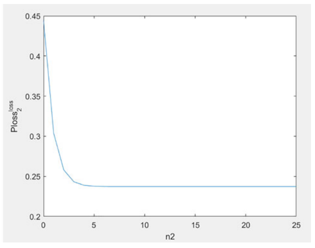

(b) Loss probability P_2^{loss} as function of n_2

Fig. 8 Comprehensive performance evaluation of two types of data under state R_2

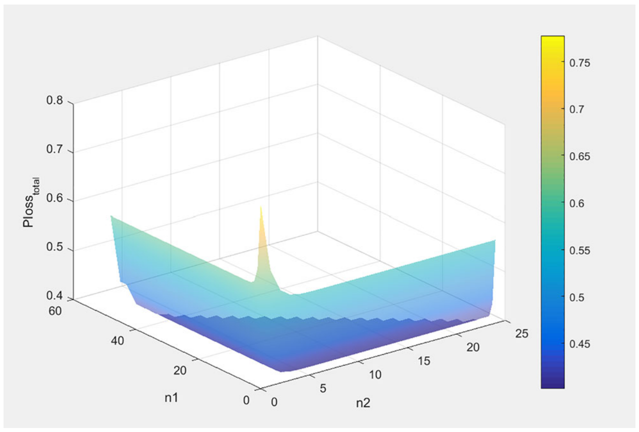


Fig. 9 Total Loss probability $Ploss_{total}$ as function of two types of data

6 Conclusion

In this paper, we consider an edge-PLC enabled manufacturing system, where different types of sensor data is collected and processed by the separated processing unit. All data shares the same space-constrained executing storage which is a bottle neck of the system. At the same time, the system may shift from states to states, where parameters change distinctly. To analyze the system performance, we view it as the one composed of multiple independent single-server queues and model it as the finite state Markov chain of the $M/M/1/n$ type. For both queuing systems, we calculated the stationary distribution of system states and performance measures. Through the results obtained, we can analyze how to allocate memory space for two kinds of data in different states to minimize the total loss rate of data. Our results can easily be extended to cases of more than two types of data. In [7], an interesting idea using Fluid Dynamics to predict time-evolving rating is proposed. Inspired by this work, in the future, we are trying to extend the performance analysis of task flows using Fluid Dynamics.

Acknowledgments This paper is supported by the Natural Science Foundation of China under Grant 61601157. Many thanks to Wasi Wasif for his help in proof-reading.

References

1. Bhuiyan MZA, Tian W, Qi L, Wang G, Wu J, Hayajneh T (2019) Preserving balance between privacy and data integrity in edge-assisted internet of things. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2019.2951687
2. Buyya R, Srirama SN (eds) (2019) Fog and Edge Computing. Wiley Series on Parallel and Distributed Computing. Wiley, New York
3. Duan S, Zhang D, Wang Y, Li L, Zhang Y (2019) Jointrec: A deep learning-based joint cloud video recommendation

المنارة للاستشارات
www.manaraa.com

framework for mobile iots. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2019.2944889
4. Dudin S, Kim C (2017) Analysis of multi-server queue with spatial generation and location-dependent service rate of customers as a cell operation model. IEEE Trans Commun 65(10):4325–4333. https://doi.org/10.1109/TCOMM.2017.2717825
5. Han Y, Park B, Jeong J (2019) Fog based iiot architecture based on big data analytics for 5g-networked smart factory. In: Misra S., Gervasi O., Murgante B., Stankova E., Korkhov V., Torre C., Rocha A. M. A., Taniar D., Apduhan B. O., Tarantino E. (eds) Computational science and its applications – ICCSA 2019. Springer International Publishing, Cham, pp 44–52
6. Hossain MS, Muhammad G (2016) Cloud-assisted industrial internet of things iiot - enabled framework for health monitoring. Comput Netw 101:192–202. https://doi.org/10.1016/j.comnet.2016.01.009. http://www.sciencedirect.com/science/article/pii/S1389128616300019. Industrial Technologies and Applications for the Internet of Things
7. Jiang W, Wu J, Wang G, Zheng H (2016) Forming opinions via trusted friends: Time-evolving rating prediction using fluid dynamics. IEEE Trans Comput 65(4):1211–1224. https://doi.org/10.1109/TC.2015.2444842
8. Kafhali SE, Salah K (2019) Performance modelling and analysis of internet of things enabled healthcare monitoring systems. IET Netw 8(1):48–58
9. Kim C, Dudin A, Dudin S, Dudina O (2017) Performance evaluation of a wireless sensor node with energy harvesting and varying conditions of operation. In: 2017 IEEE International conference on communications (ICC), pp 1–6. https://doi.org/10.1109/ICC.2017.7996994
10. Liu X, Cao J, Yang Y, Qu W, Zhao X, Li K, Yao D (2019) Fast rfid sensory data collection: Trade-off between computation and communication costs. IEEE/ACM Trans Netw 27(3):1179–1191
11. Liu X, Xie X, Wang S, Liu J, Yao D, Cao J, Li K (2019) Efficient range queries for large-scale sensor-augmented rfid systems. IEEE/ACM Trans Netw 27(5):1873–1886
12. Lou P, Yuan L, Hu J, Yan J, Fu J (2018) A comprehensive assessment approach to evaluate the trustworthiness of manufacturing services in cloud manufacturing environment. IEEE ACCESS 6 (9-12):30819–30828
13. Madhuri P, Nagesh AS, Thirumalaikumar M, Varghese Z, Varun AV (2009) Performance analysis of smart camera based distributed control flow logic for machine vision applications In: 2009 IEEE International conference on industrial technology, pp 1–6. https://doi.org/10.1109/ICIT.2009.4939499
14. Strau P, Schmitz M, Wstmann R, Deuse J (2018) Enabling of predictive maintenance in the brownfield through low-cost sensors, an iiot-architecture and machine learning. In: 2018 IEEE International conference on big data (big data), pp 1474–1483. https://doi.org/10.1109/BigData.2018.8622076
15. Sun W, Liu J, Yue Y, Zhang H (2018) Double auction-based resource allocation for mobile edge computing in industrial internet of things. IEEE Trans Ind Inf 14(10):4692–4701
16. Tang W, Ren J, Zhang K, Zhang D, Zhang Y, Shen XS (2019) Efficient and privacy-preserving fog-assisted health data sharing scheme. ACM Transactions on Intelligent Systems and Technology. https://doi.org/10.1145/3341104
17. Tang W, Ren J, Zhang Y (2019) Enabling trusted and privacy-preserving healthcare services in social media health networks. IEEE Trans Multimed 21(3):579–590
18. Wang Q, Yang H, Wang Q, Huang W, Deng B (2019) A deep learning based data forwarding algorithm in mobile social networks. Peer-to-Peer Netw Appl 12(6):1638–1650
19. Wang T, Zhang G, Liu A, Bhuiyan MZA, Jin Q (2019) A secure iot service architecture with an efficient balance dynamics based on cloud and edge computing. IEEE Internet Things J 6(3):4831–4843
20. Wu Y, Qian LP, Mao H, Yang X, Zhou H, Tan X, Tsang DHK (2018) Secrecy-driven resource management for vehicular computation offloading networks. IEEE Netw 32(3): 84–91
21. Yang H, Cheng L, Chuah MC (2018) Detecting payload attacks on programmable logic controllers (plcs). In: 2018 IEEE Conference on communications and network security (CNS), pp 1–9. https://doi.org/10.1109/CNS.2018.8433146
22. Zhang D, Qiao Y, She L, Shen R, Ren J, Zhang Y (2019) Two time-scale resource management for green internet of things networks. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2018.2842766
23. Zhang D, Shen R, Ren J, Zhang Y (2018) Delay-optimal proactive service framework for block-stream as a service. IEEE Wirel Commun Lett 7(4):598–601
24. Zhang D, Tan L, Ren J, Awad MK, Zhang S, Zhang Y (2019) Near-optimal and truthful online auction for computation offloading in green edge-computing systems. IEEE Transactions on Mobile Computing. https://doi.org/10.1109/TMC.2019.2901474

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

المنارة للاستشارات
www.manaraa.com

Reproduced with permission of copyright owner. Further reproduction prohibited without permission.

المنارة للاستشارات

www.manaraa.com