The Journal of Supercomputing (2020) 76:6238–6257
https://doi.org/10.1007/s11227-019-02771-3

Response time of a ternary optical computer that is based on queuing systems

Xianchuan Wang[1] · Sulan Zhang[2] · Shan Gao[1] · Mian Zhang[1] · Jie Zhang[1] · Xianchao Wang[1] · Zheng Xu[2]

Published online: 9 February 2019

Abstract

In this paper, a four-stage service model is constructed by combining M/M/1, M^X/M/1 and M/M^B/1 queuing systems. In addition, the immediate scheduling strategy and its algorithm are presented in detail, and the computing accomplished scheduling strategy and its algorithm are proposed. Approaches for computing the receiving time, preprocessing time, operating time and transmission time of operation requests that are based on various queuing systems are discussed, and the response time is calculated by adding these times together. Finally, the response times under the two scheduling strategies are obtained by simulating the models numerically, and the results demonstrate that the proposed computing accomplished scheduling strategy outperforms the immediate scheduling strategy.

Keywords M^X/M/1 · M/M^B/1 · Immediate scheduling · Computing accomplished scheduling · Performance analysis

✉ Xianchao Wang
13965563690@163.com; wxcdx@126.com

Xianchuan Wang
xch_wang@126.com

Sulan Zhang
zhangsl000111@163.com

Shan Gao
gaoshan990504@163.com

Mian Zhang
zhmlqf@163.com

Jie Zhang
zjp562@126.com

Zheng Xu
zhengxu@shu.edu.cn

[1] Fuyang Normal University, Fuyang, Anhui, China

[2] Shanghai University, Shanghai, China

1 Introduction

Optical computing is a novel paradigm that is attracting increasing attention [1–5] due to the multidimensional parallel nature of light and the bottlenecks of the electronic computer, such as power, bandwidth and computing speed. Among these optical computing platforms, the ternary optical computer (TOC) was proposed by Jin et al. [4].

Various achievements have been realized in the theory and application of TOC. For example, the principle [4] and architecture [5] of TOC were presented and the decrease-radix design principle [6] made the construction of the optical processor normative. Carry-free addition [7–10] and vector–matrix multiplication [7] experiments inaugurated the application of TOC in high-performance computing, and the adder of TOC was designed and implemented based on the modified signed-digit (MSD) number system [8–10]. Moreover, the task management system was explored preliminarily [11–14]. These achievements have effectively promoted TOC from theory to experiment and application step by step.

Another crucial issue, namely the quality of service (QoS), of which the service performance is an important aspect, has not been adequately considered, except in our previous research [15]. In this paper, we shall conduct detailed research on the performance evaluation of TOC in terms of the response time for services. The response time is defined as the time for an operation request to be serviced, namely the elapsed time from a customer's submission of a request to TOC until he receives the final output from it.

The main contributions of this paper are the design of a four-stage TOC service model, the proposal of the immediate scheduling strategy, the computing accomplished scheduling strategy and the construction of an analytical model of the response time. The response time was obtained via numerical simulation, and the results demonstrated that the computing accomplished scheduling strategy outperformed the immediate scheduling strategy.

The remainder of this paper is organized as follows: Section 2 briefly introduces preliminaries of the work. Section 3 builds the service model of a TOC, which is based on a queuing system. Section 4 proposes the immediate scheduling strategy and the computing accomplished scheduling strategy and discusses task scheduling algorithms, processor allocation algorithms and proportional allocation algorithms under the two scheduling strategies. In Sect. 5, we present the model of the response time in detail. Section 6 presents the numerical results that are obtained by simulating the model. Finally, Sect. 7 discusses the conclusions of this work and potential future directions.

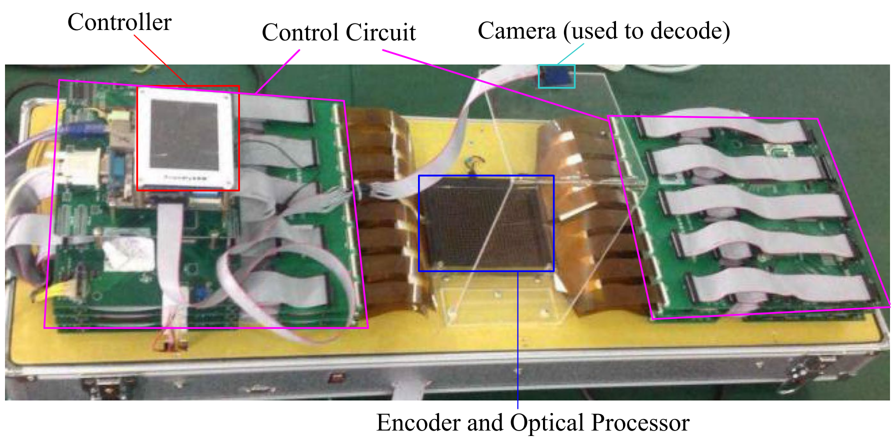

Fig. 1 Real TOC that was constructed at Shanghai University

2 Related work

Figure 1 shows the hardware components of a TOC that was constructed at Shanghai University at the beginning of 2017. It is similar to cloud computing and is a service-oriented architecture. Its service is also classified into three categories: Infrastructure as a Service (IaaS) is made up of various hardware components in TOC, such as the servers, controller, encoder, reconfigurable optical processor, decoder and network resource. Platform as a Service (PaaS) provides a computing platform such as a task management system, a database, a calculation routine and development tools. In Software as a Service (SaaS), the TOC provider offers a software delivery model in which customers can obtain expediently the application programs.

However, TOC differs from other parallel computing platforms in the following aspects: First, TOC principally and directly processes two-input tri-valued logic operations. Second, its optical processor is flexible. In other words, TOC can reconfigure the optical processor according to customers' requests in runtime. Third, there are many data bits in the optical processor, and they are fully parallel. The optical processor that is shown in Fig. 1 has 192 data bits, and it can be easily expanded to ten thousand or more. Thus, TOC can concurrently process many operation requests. Fourth, the diversity of the operation requests, the dynamical reconfigurability of the optical processor and the time dependence of the load enable TOC to provide the QoS that is expected by customers for a wider range of applications. However, we cannot directly apply the approaches for the performance analysis of cloud computing to TOC. Consequently, this paper will investigate the service performance of TOC based on queuing system and determine whether it can offer higher QoS.

Queueing theory, which is an important branch of stochastic operations research, is widely used in communication, task scheduling, resource allocation and cloud computing. Queuing systems have been used to explore the performance analysis and evaluation of various parallel computing platforms, such as cloud computing.

In Vilaplana et al. [16], a resource optimization model was constructed for a cloud computing platform by combining M/M/1 and M/M/m queuing systems. In Yang et al. [17], the probability distribution of the response time for a cloud service center was deduced based on an M/M/m/m+r queuing system under the assumption that the task arrival time interval and service time follow exponential distributions. In Khazaei et al. [18], an approximate analytical model was constructed for the service performance analysis and evaluation of cloud computing centers using an M/G/m/m+r queuing system and an approximated Markov chain and various performance indicators, such as the mean number of tasks in the system, the task response time and the blocking probability, were obtained by simulating the model numerically. In Jaiganesh et al. [19], a queue model with indicators such as service time and response time was constructed for evaluating a cloud service with the objective of maximizing the profit based on an M/G/m/m+r queuing system and the first-come-first-served scheduling strategy. In Bai et al. [20], a complex queue model was constructed for the performance evaluation of heterogeneous data centers by connecting two queuing systems, in which the indicators were the mean response time and the mean waiting time. In Cao et al. [21], a power allocation and load distribution model was constructed based on an M/M/m queuing system for optimizing the heterogeneous multicore server processor in cloud computing. These explorations yielded a performance analysis model for parallel computing platforms such as cloud computing and numerical simulations or system experiments were conducted. They provided a theoretical foundation for constructing novel and more effective parallel computing systems. In other words, queuing theory can afford an organized, structural and concise framework for analyzing, evaluating and simulating the service process of optical computing. Therefore, this paper investigates the service performance of TOC under two types of scheduling strategies, namely the immediate scheduling strategy and the computing accomplished scheduling strategy, which are based on different queuing systems.

3 Service model of the ternary optical computer

The service model of TOC is illustrated in Fig. 2. The model consists of four stages: In Stage 1, customers submit their operation requests to the receiving server (RS) in high-level language text, which is similar to the other high-performance computing platforms such as cloud computing. Assume that there is an infinite queue in RS and the operation requests are queued on a first-come-first-served (FCFS) basis when they arrive. RS transmits them to the preprocessing server (PPS) after receiving them. In

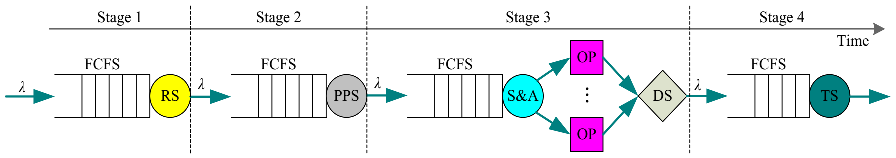

Fig. 2 Service model of the ternary optical computer

Stage 2, PPS transforms the operations into two-input tri-valued logic operations, changes the binary data into tri-valued data as the control internal code, executes each operation in the requests and queues the requests on an FCFS basis. In Stage 3, the scheduler (S) simultaneously schedules several requests on an FCFS basis via various scheduling strategies and the allocator (A) allocates the data bits of the optical processor to the scheduled requests. At the same time, it sends the allocated results and the reconfiguration codes to the optical processor (OP). OP performs the optical computing after its reconfiguring component has completed the reconfiguration of the optical processor. The decoder server (DS) decodes the computing results to generate tri-valued results in the control internal code and sends them to the transmitting server (TS) or the OP as feedback for the optical computing. Finally, in Stage 4, the TS transforms the tri-valued results into operation results that can be understood by the customers and transmits them to the corresponding clients. The four stages constitute a queuing system, namely a four-stage tandem queuing network [22].

The QoS of TOC includes the reliability, security, availability, throughput and many other parameters, along with performance indicators such as the response time, waiting time, task blocking probability, prompt service probability and mean task number in the system, all of which can be obtained via the queuing system. In this paper, we shall focus on the response time for analyzing and evaluating the performance of TOC.

4 Task scheduling and processor allocation

We presented allocation-by-demand algorithm [14] and a dynamic data-bit allocation algorithm [13]. In this paper, we proposed two novel task scheduling strategies, namely the immediate scheduling (IS) strategy and the computing accomplished scheduling (CAS) strategy, and a processor allocation strategy, namely the proportional allocation strategy.

4.1 Immediate scheduling

Under the IS strategy, the data bits of the optical processor are equally divided into n parts and each part corresponds to a small optical processor that can be independently used. The scheduler immediately schedules the currently arriving requests if there is an idle small optical processor. Otherwise, they are queued for scheduling and are not scheduled until there is an idle small optical processor. Denote the task number of the task that is being processed as N_{Proc} and the length of scheduling queue Q as L_Q. The corresponding immediate scheduling algorithm is as follows:

Algorithm 1 Task scheduling algorithm under the IS strategy

```
Step 1 Initialize the parameters to $N_{\mathrm{Proc}} = 0$ and $L_Q = 0$.
Step 2 Increase $L_Q$ by 1 when a task arrives.
Step 3 Judge whether $N_{\mathrm{Proc}}$ is equal to $n$. If so, jump to Step 5. Otherwise, judge whether $L_Q$ is 0. If so, jump to Step 5. Otherwise, schedule a task in $Q$, decrease $L_Q$ by 1 and increase $N_{\mathrm{Proc}}$ by 1.
```

Step 4 Decrease N_{Proc} by 1 and jump to Step 2 when the scheduler receives a semaphore that a task has been accomplished.
Step 5 End.

A task is scheduled in every iteration; hence, under the IS strategy, the scheduling number attains its maximum value for the specified task set. At the same time, the allocation number and the reconfiguration number also attain their maximum values.

The allocator selects an unoccupied small optical processor and proportionally allocates its data bits for each operation in the just-scheduled task to guarantee that the operations can be synchronously accomplished. Denote the optical processor as N_{tot}. Then, the number of data bits of each small optical processor is expressed as $N_{\text{sop}}=N_{\text{tot}}/n$. Denote the number of two-input tri-valued logic operations in the scheduled task as N_{Log}. The allocation algorithm under the IS strategy is as follows:

Algorithm 2 Allocation algorithm under the IS strategy

Step 1 Initialize the relevant parameters, set $i=1$ and initialize the computation C of the task to 0.
Step 2 Judge whether i is greater than N_{Log}. If so, jump to Step 4. Otherwise, $C=C+C_i$ (C_i is the computation of the ith operation, which is calculated by PPS).
Step 3 Increase i by 1 and jump to Step 2.
Step 4 $i=1$.
Step 5 Judge whether i is greater than N_{Log}. If so, jump to Step 7. Otherwise, proportionally allocate the data bits for the request as follows:

$$N_i = \left\lfloor \frac{c_i}{c} \times N_{\text{sop}} \right\rfloor.$$

Step 6 Increase i by 1 and jump to Step 5.
Step 7 End.

Steps 1–3 perform the computations of the scheduled task, and Steps 4–6 allocate the data bits of the selected small optical processor. A low arrival rate and small computations may cause small optical processors to be unoccupied, which will reduce the utilization rate of the complete optical processor.

4.2 Computing accomplished scheduling

For improving the utilization rate, we propose the computing accomplished scheduling strategy. Under this strategy, the scheduler schedules the requests when the optical computing of the requests has been accomplished. We denote the maximum number of requests that the processor can simultaneously compute as n, the number of requests to be scheduled as N_{Sched} and the number of two-input tri-valued logic operations in the ith request as $N_{i\text{Log}}$. The scheduling algorithm under the CAS strategy is as follows:

Algorithm 3 Task scheduling algorithm under the CAS strategy

```
Step 1 Initialize the parameters to $N_{\mathrm{Proc}}=0$, $L_Q=0$ and $N_{\mathrm{Sched}}=0$.
Step 2 Schedule the first request according to the IS strategy after it has entered the scheduling queue and $L_Q$ has been increased by 1. Increase both $N_{\mathrm{Proc}}$ and $N_{\mathrm{Sched}}$ by 1 and decrease $L_Q$ by 1. Jump to Step 4.
Step 3 $N_{\mathrm{Sched}}=n-N_{\mathrm{Proc}}$ when the scheduler receives a task-accomplished semaphore.
Step 4 Send $N_{\mathrm{Sched}}$ to the allocator.
Step 5 Judge whether $L_Q$ is equal to 0. If so, jump to Step 7. Otherwise, schedule a request in $Q$, decrease $N_{\mathrm{Sched}}$ by 1 and increase $N_{\mathrm{Proc}}$ by 1.
Step 6 Judge whether $N_{\mathrm{Sched}}$ is equal to 0. If so, jump to Step 7. Otherwise, jump to Step 5.
Step 7 End.
```

The scheduler schedules a request and multiple requests in every iteration under the IS strategy and the CAS strategy, respectively. To improve the utilization rate of the optical processor and guarantee that the requests will be accomplished simultaneously, the allocator similarly allocates the data bits of the whole optical processor proportionally. The corresponding allocation algorithm is as follows:

Algorithm 4 Allocation algorithm under the CAS strategy

```
Step 1 Initialize the parameters. Set $i=1$ and $j=1$ and initialize the total computation $C$ of $N_{\mathrm{Sched}}$ requests to 0.
Step 2 Judge whether $i$ is greater than $N_{\mathrm{Sched}}$. If so, jump to Step 6. Otherwise, proceed to Step 3.
Step 3 Judge whether $j$ is greater than $N_{i\mathrm{Log}}$. If so, proceed to Step 4. Otherwise, $C=C_{ij}+C$ (where $C_{ij}$ is the computation of the $j$th operation in the $i$th request, which is calculated by PPS).
Step 4 Increase $j$ by 1 and return to Step 3.
Step 5 Increase $i$ by 1 and return Step 2.
Step 6 $i=1, j=1$.
Step 7 Judge whether $i$ is greater than $N_{\mathrm{Sched}}$. If so, jump to Step 11. Otherwise, proceed to Step 8.
Step 8 Judge whether $j$ is greater than $N_{\mathrm{iLog}}$. If so, proceed to Step 9. Otherwise, allocate the data bits for the $j$th operation in the $i$th task as follows:

        $$P_{ij} = \left\lfloor \frac{c_i}{c} \times N \right\rfloor.$$

Step 9 Increase $j$ by 1 and return to Step 8.
Step 10 Increase $i$ by 1 and return to Step 7.
Step 11 End.
```

Steps 1–5 are used to calculate the total computation C of the N_{Sched} requests, and Steps 6–10 are used to allocate the data-bit resource. We call Algorithm 2 and Algorithm 4 proportional allocation algorithms. The main difference between them is that the former allocates the data bits of a small optical processor in each iteration, while the latter allocates all data bits of the whole optical processor in each iteration.

5 Model of the response time

The response time, which is denoted by T, can be obtained via the following formula when the system is in equilibrium.

$$T = T_{\mathrm{R}} + T_{\mathrm{P}} + T_{\mathrm{C}} + T_{\mathrm{T}} \tag{1}$$

where T_{R} represents the receiving time, T_{P} the preprocessing time, T_{C} the computing time that is spent in Stage 3 and T_{T} the transmission time. Assume that they are statistically independent from one another.

5.1 Receiving time

Assume that the arrival intervals of the requests follow a negative exponential distribution with parameter λ, and the service times of RS for the requests are independent and identically distributed random variables that follow a negative exponential distribution with parameter μ_{R}, which denotes the service rate. Parameter μ_{R} is closely related to the network transmission speed, which is denoted as ω, and the mean transmission time of the requests, which is denoted as η: $\mu_{\mathrm{R}} = \omega/\eta$. Therefore, we model Stage 1 as an M/M/1 queuing system with a request buffer of infinite capacity and single request arrival. The state transition diagram for the continuous-time Markov chain (CTMC) of RS is shown in Fig. 3, where state m refers to the request number in RS and $m-1$ requests are waiting to be received.

Let $\rho_{\mathrm{R}} = \lambda/\mu_{\mathrm{R}}$. The system is stationary when $\rho_{\mathrm{R}} < 1$ [23–25]. The following equations can be obtained according to the universal law of K-algebraic equations, in which the probability of the mth state is denoted as P_m ($m = 0, 1, 2, \ldots$):

$$\begin{cases} \lambda P_0 = \mu_{\mathrm{R}} P_1, \\ (\lambda + \mu_{\mathrm{R}}) P_m = \lambda P_{m-1} + \mu_{\mathrm{R}} P_{m+1}, \; m \geq 1. \end{cases}$$

Then, $P_m = \rho_{\mathrm{R}}^m P_0, \quad m \geq 1$.

Using the regularity condition $\sum_{m=0}^{\infty} P_m = 1$, we calculate the idle probability P_0 of RS:

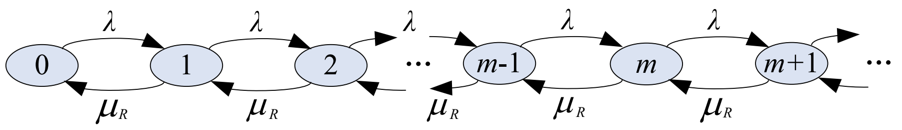

Fig. 3 State transition diagram for the M/M/1 service model of RS

$$P_0 = 1 - \rho_R.$$

We can easily obtain the mean number of requests N_R in RS:

$$N_R = \sum_{m=0}^{\infty} iP_i = \rho_R(1-\rho_R)\sum_{i=0}^{\infty} i\rho_R^{i-1}$$
$$= \rho_R(1-\rho_R)\left(\frac{\rho_R}{1-\rho_R}\right)' = \frac{\rho_R}{1-\rho_R} = \frac{\lambda}{\mu_R - \lambda}.$$

Via Little's formulas, we obtain the mean receiving time, namely T_R, of the requests:

$$T_R = \frac{N_R}{\lambda} = \frac{1}{\mu_R - \lambda}. \tag{2}$$

5.2 Preprocessing time

The requests that are received by RS are immediately transmitted to PPS for preprocessing. Therefore, we similarly represent PPS as an M/M/1 queuing system, and the state transition diagram for CTMC of PPS is identical to that in Fig. 3. According to Kleinrock [24] and Bhat [25], the arrival intervals of requests that arrive to PPS also follow a negative exponential distribution with parameter λ. Let the mean computation of two-input tri-valued logic operations that are transformed by PPS be denoted by and the transformation speed of the data into control internal codes be denoted as v. Then, the service rate of PPS is calculated as follows: $\mu_P = v/C$. Again, let $\rho_P = \lambda/\mu_P$. The stationary probability equations are identical to those of RS when $\rho_P < 1$. Consequently, the conclusion in Sect. 5.1 can be directly applied to the mean preprocessing time, namely T_P:

$$T_P = \frac{1}{\mu_P - \lambda}. \tag{3}$$

5.3 Computing time

The scheduler sends the requests to the optical processor of TOC after scheduling them on an FCFS basis. Then, the allocator allocates the data-bit resource of the optical processor among the just-scheduled requests and sends the results and the reconfiguration codes of the optical processor to TOC. The reconfiguration component of the optical processor implements the reconfiguration fully in parallel. The operator implements the transformation of the optical states, namely the optical computing, after the encoder has transformed the electrical signals, namely the data in control internal codes, into optical signals. Finally, DS transforms the computing results into electrical signals in control internal codes.

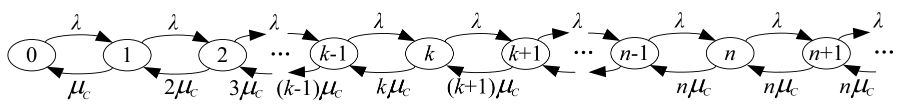

Fig. 4 State transition diagram for the computing time under the IS strategy

Differences in the scheduling strategies of the requests and the allocation strategies of the optical processor may lead to differences in the approaches for calculating the computing time. We focus on exploring the calculating approaches under the IS and CAS strategies, which are proposed in Sect. 4.

5.3.1 Computing time under the immediate scheduling strategy

Stage 3 can be represented with an M/M/n queuing system under the IS strategy, where n denotes the number of independent small optical processors. The state transition diagram for CTMC of the stage under the IS strategy is shown in Fig. 4, where μ_C is the mean service rate of each small optical processor, which is expressed as $\mu_C=\mu/n$, where μ is the mean service rate of the whole optical processor. Hence, $\mu=\tau/C$, where τ is the computing speed of the whole processor. In addition, in state k, k small optical processors are computing a request, respectively, respectively, and the remaining processors are unoccupied if $0\le k<n$. Otherwise, each small optical processor is busy computing and $k-n$ requests are waiting in the queue to be serviced.

The arrival rate of requests that are arriving to Stage 3 is also λ under this strategy. Let $\rho_{C1}=\lambda/\mu_C=n\lambda/\mu$ and $\rho_C=\rho_{C1}/n=\lambda/\mu$. Similarly, the system is stationary when $\rho_C<1$. Moreover, the K-algebraic equations can be obtained easily according to Fig. 4 and the equilibrium distribution is as follows:

$$P_k=\begin{cases}\frac{\rho_{C1}^k}{k!}P_0=\frac{(n\rho_C)^k}{k!}P_0, & 0\le k<n\\ \frac{\rho_{C1}^k}{n!n^{k-n}}P_0=\frac{n^n\rho_C^k}{n!}P_0, & n\le k\end{cases}$$

Using the regularity condition, we can calculate the idle probability P_0 of Stage 3

$$P_0=\left[\sum_{k=0}^{n-1}\frac{\rho_{C1}^k}{k!}+\frac{\rho_{C1}^n}{n!(1-\rho_C)}\right]^{-1},$$

and the mean number of requests

$$N_C=\frac{\rho_C\rho_{C1}^nP_0}{n!(1-\rho_C)^2}+\rho_{C1}.$$

Then, the mean computing time, namely T_C, can be obtained according to the Little's law.

$$T_C=\frac{N_C}{\lambda}. \tag{4}$$

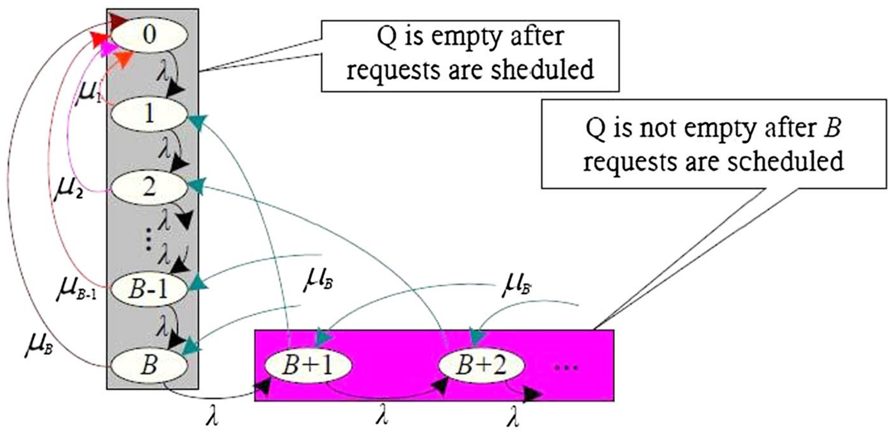
Q is empty after requests are sheduled
Q is not empty after B requests are scheduled
0
1
2
B-1
B
B+1
B+2
...
λ
μ_1
μ_2
μ_{B-1}
μ_B

Fig. 5 State transition diagram for the computing time under the CAS strategy

5.3.2 Computing time under the computing accomplished scheduling strategy

If there are i $(0 \le i \le n)$ requests in the queue, the whole optical processor is divided into i small optical processors for computing them after they have been scheduled according to the CAS strategy. We denote the service rate of each small optical processor as μ_i, which is expressed as $\mu_i = \mu/i$. The state transition diagram for the computing time under the CAS strategy is shown in Fig. 5.

Under this strategy, we model Stage 3 as an M/M^B/1 partial batch service queuing system [23, 25], where B is the maximum number of requests for the batch service, which is equal to the number n of small optical processors, as described in Sect. 5.3.1.

Figure 6 illustrates the progress of scheduling queue Q, where t_1, t_2, t_3 and t_4 are the times at which the optical processor finishes computing and the scheduler S schedules the requests in Q. At t_1 and t_4, S schedules all requests in Q because the number of requests in Q does not exceed the maximum B of the batch

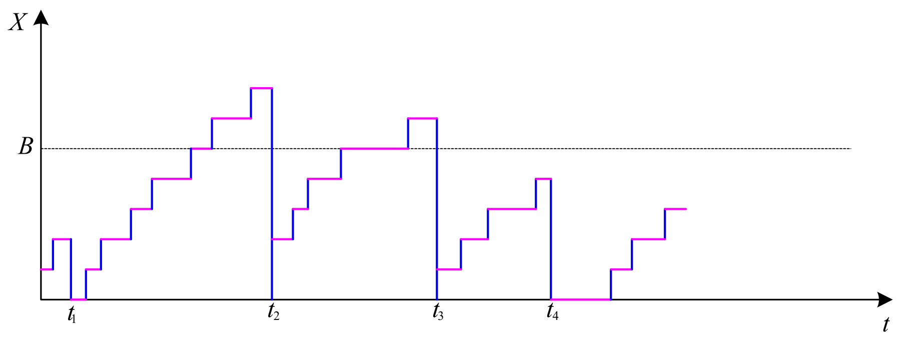

Fig. 6 Scheduling queue changes under the partial batch service model

service. At t_2 and t_3, S schedules B requests in Q because the number of requests has exceeded B.

According to Fig. 5, the K-algebraic equations in equilibrium can be obtained.

$$\begin{aligned} &-\lambda P_0 + \mu(P_1 + P_2 + \cdots + P_B) = 0, \\ &-(\lambda + \mu)P_k + \lambda P_{k-1} + \mu P_{k+B} = 0, \quad k \geq 1 \end{aligned} \tag{5}$$

The second equation of (5) can be rewritten in operator notation as

$$(\mu D^{B+1} - (\lambda + \mu)D + \lambda)P_k = 0, \quad k \geq 0.$$

Consider the characteristic equation: $\mu x^{B+1} - (\lambda + \mu)x + \lambda = 0$. There is one root in (0, 1) for the equation [23, 25]. Denote the root by x_0. Using the regularity condition, we can obtain

$$P_k = (1 - x_0)x_0^k, \quad (k \geq 0).$$

Thus, the computing time is expressed as follows:

$$T_\mathrm{C} = \frac{x_0}{\lambda(1 - x_0)}. \tag{6}$$

5.4 Transmission time

The transmission time is also relevant in the scheduling strategies and algorithms. Denote the mean traffic of the computing results as R. Then, the mean service rate of the transmitting server TS is $\mu_\mathrm{T} = \omega/R$. The calculation of the transmission time is discussed under the IS and CAS strategies.

5.4.1 Transmission time under the immediate scheduling strategy

Under the IS strategy, the computing results also reach TS one by one. Therefore, we model it as an M/M/1 queuing system. Similarly, the transmission time is expressed as follows:

$$T_\mathrm{T} = \frac{1}{\mu_\mathrm{T} - \lambda}. \tag{7}$$

5.4.2 Transmission time under the computing accomplished scheduling strategy

Under the CAS strategy, batch service naturally generates batch arrival. Consequently, the computing results reach TS in bulk. We model Stage 4, namely the transmitting

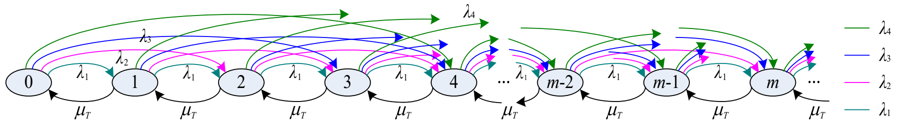

Fig. 7 State transition diagram for the transmission model under the CAS strategy

server, as an M^X/M/1 batch arrival queuing system [23, 25], where X is a random variable; hence, the number of computing results differs among bulks. A sample state transition diagram for the model is shown in Fig. 7, where λ_i ($i=1, 2, \ldots, B$) is the arrival rate of i requests per bulk and $X \in \{1, 2, 3, 4\}$. Let λ be the composite arrival rate of all bulks. Thus, $\lambda = \sum_{i=1}^{B} \lambda_i$; moreover, the probability P_{Bi} of bulk size i is equal to λ_i/λ. The following equilibrium equations can be obtained:

$$\begin{aligned} &-\lambda P_{B0} + \mu_{\mathrm{T}} P_{B1} = 0, \\ &-(\lambda + \mu_{\mathrm{T}}) P_k + \mu_{\mathrm{T}} P_{k+1} + \lambda \sum_{i=1}^{k} P_{k-i} P_{Bi} = 0, \quad (k \geq 1) \end{aligned} \tag{8}$$

We use a generating function approach to solve this system of equations (8). To obtain the solution, we define

$$\begin{cases} P_B(z) = \sum\limits_{i=1}^{\infty} P_{Bi} z^i, \ (|z| \leq 1), \\ P(z) = \sum\limits_{i=0}^{\infty} P_i z^i, \quad (|z| \leq 1) \end{cases} \tag{9}$$

as the generating functions of the batch-size probabilities $\{P_{Bi}\}$ and the steady-state probabilities $\{P_i\}$, respectively. $P_B(z)$ can be viewed an input of the system because the batch-size probabilities are typically known. The objective is to determine $P(z)$ from $P_B(z)$ and to obtain from this the unknown steady-state probabilities $\{P_i\}$. Multiplying each equation of (8) by z_i and summing them yields

$$-\lambda \sum_{i=0}^{\infty} P_i z^i - \mu_{\mathrm{T}} \sum_{i=1}^{\infty} P_i z^i + \frac{\mu_{\mathrm{T}}}{z} \sum_{i=1}^{\infty} P_i z^i + \lambda \sum_{i=1}^{\infty} \sum_{k=1}^{i} P_{i-k} P_{Bk} z^i = 0, \tag{10}$$

and thus,

$$P(z) = \frac{\mu_{\mathrm{T}} P_0 (1-z)}{\mu_{\mathrm{T}}(1-z) - \lambda z[1 - P_B(z)]'} \quad (|z| \leq 1). \tag{11}$$

Since $\lim_{z \to 1} P(z) = 1$, take the limit of (11):

$$\lim_{z \to 1} P(z) = \frac{\lim\limits_{z \to 1} \mu_{\mathrm{T}} P_0 (1-z)}{\lim\limits_{z \to 1} \mu_{\mathrm{T}}(1-z) - \lambda z[1 - P_B(z)]}. \tag{12}$$

Now, P_0 can be calculated by applying L'Hopital's rule to (12). Therefore,

$$P_0 = 1 - \frac{\lambda P_B'(1)}{\mu_{\mathrm{T}}},$$

and

$$P'_B(1) = \lim_{z\to 1} \sum_{i=1}^{\infty} iP_{Bi} z^{i-1} = E(X),$$

where $E(X)$ is the mathematical expectation of random variable X, which is the mean number of requests in each bulk. Let $\rho_T = \lambda E(X)/\mu_T$. Taking the derivative of (11) on z and taking the limit of the result yield the mean number of computing results, namely N_T.

$$N_T = \lim_{z\to 1} P'(z) = \frac{2\rho + \frac{\lambda}{\mu_T} P''_B(1)}{2(1-\rho_T)} \tag{13}$$

and

$$P''_B(1) = E(X^2) - E(X).$$

Then,

$$N_T = \frac{\rho + \frac{\lambda}{\mu_T} E(X^2)}{2(1-\rho_T)}.$$

Thus, the transmission time is expressed as follows:

$$T_T = \frac{E(X) + E(X^2)}{2\mu_T(1-\rho_T)} \tag{14}$$

Finally, the mean response time under the IS strategy can be obtained by substituting (2, 3, 4) and (7) into (1) and that under the CAS strategy can be similarly obtained.

6 Numerical evaluation

In this section, we present numerical results for various service processes. In the discussion of the results, we focus on the influence on the response time of the scheduling strategy and the batch size. The parameters that are used in the experiment are described as follows:

- λ *Arrival rate* It is the average number of requests that are submitted to TOC per unit of time. We aim at demonstrating how the response time (T) is affected by the value of parameter λ.
- ω *Network bandwidth* It is the mean network speed at which the requests are submitted to RS, and the final outputs are sent to the customers over the Internet.
- c *Request size* It is the mean traffic of requests that are submitted to RS.
- R *Result size* It is the mean traffic of the final outputs that are sent to customers.

Table 1 Increase probabilities of the scheduling vector

λ	Increase probability			
	P_{b1}	P_{b2}	P_{b3}	P_{b4}
$1 \le \lambda \le 10$	1	0	0	0
$10 < \lambda \le 20$	2/3	1/3	0	0
$20 < \lambda \le 30$	1/2	1/3	1/6	0
$30 < \lambda \le 40$	4/10	3/10	2/10	1/10

Table 2 Influence on the response time of the arrival rate under the two scheduling strategies

λ	Response time (s)		
	IS	CAS	Ratio
1	150.705	75.705	1.991
2	151.429	76.430	1.981
…	…	…	…
59	381.432	302.229	1.262
60	404.826	325.322	1.244

- *v Preprocessing speed* It is the average speed at which PPS transforms the binary data into tri-valued data as control internal code.
- *τ Computing speed* It is the computing speed of TOC.
- *B Batch size* It is the maximum number of requests when TOC conducts batch service and it is equal to the number n of small optical processors. We aim at demonstrating how the response time is affected by the value of parameter B.

6.1 Response times under various scheduling strategies

An optical processor with 1024 data bits was constructed in 2015. For this reason, we have considered a system with a 1024-bit optical processor in the following numerical experiment. The mean arrival rate of requests is changed from $\lambda = 1$ to 60 per hour, and the mean network transmission speed is set to $\omega = 20$ MB/s, the mean size of the requests to $c = 0.01$ MB, the mean traffic of the computing results to $R = 0.05$ MB, the preprocessing speed to $v = 1$ GB/s, the computing speed of TOC to $\tau = 2$ GB/s and the maximum of batch size to $B = 4$. Since TOC is especially suitable for big data, we assume that the mean computation C is equal to 50 G. To compare the two scheduling strategies, we assume that the number n of small optical processors, namely the maximum number of requests that can be processed in parallel under the IS strategy, is equal to B.

For the CAS strategy, B is equal to 4; however, when λ is small, namely the arrival interval is large, only one request is computed per bulk in the optical processor. The batch size will increase gradually with the increase of λ. To calculate $E(X)$ and $E(X^2)$ in (14), we denote by $\mathscr{B} = [b_1\ b_2\ b_3\ b_4]$ a scheduling vector of λ, where b_i ($i = 1, 2, 3, 4$) is the frequency of scheduling i requests. Their increase probabilities

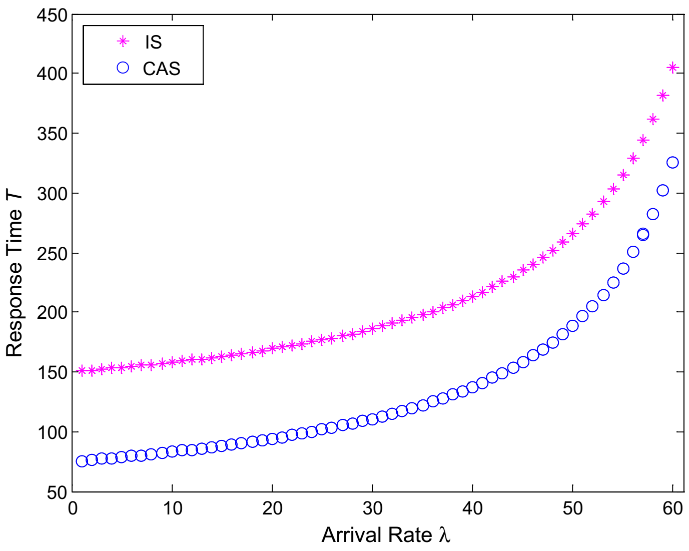

Fig. 8 Response time versus arrival rate under the two scheduling strategies

vary widely: The larger i is, the smaller the increase probability of b_i is. The increase probabilities are listed in Table 1.

Let $\rho = \max\{\rho_R, \rho_P, \rho_C, \rho_T\}$. For the parameters above, $\rho < 1$, that is, the system is in equilibrium, we conduct a numerical simulation of the model of the response time in MATLAB R2010b. The results are presented in Table 2 and Fig. 8.

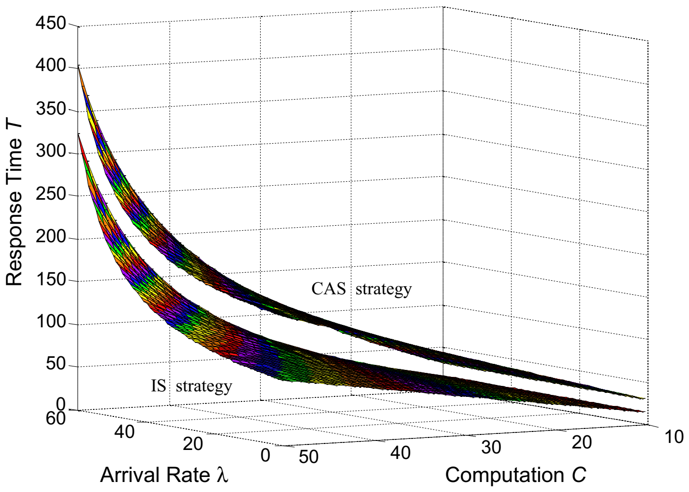

Fig. 9 Response times versus arrival rate and computation

Table 3 Response times under the two scheduling strategies when n and B are varied

Scheduling strategy	n and B	Response time (s)						
		$\lambda=1$	$\lambda=10$	$\lambda=20$	$\lambda=30$	$\lambda=40$	$\lambda=50$	$\lambda=60$
IS	1	75.847	84.682	98.008	117.418	148.239	205.067	349.691
	2	100.707	108.285	120.157	137.957	166.895	221.400	363.008
	3	125.705	133.098	144.487	161.566	189.539	242.773	382.668
	4	150.705	158.070	169.306	186.057	213.507	265.989	404.826
CAS	1	75.880	84.944	98.316	117.410	147.321	202.254	343.315
	2	75.706	83.185	94.626	111.537	138.878	190.691	327.840
	3	75.705	83.079	94.267	110.766	137.574	188.777	325.231
	4	75.705	83.079	94.267	110.752	137.571	188.776	325.209

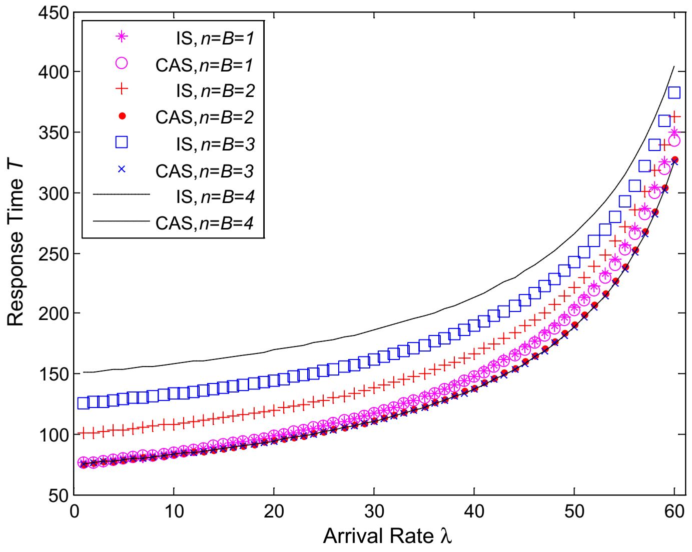

Fig. 10 Response times when n and B are varied

The response times under the two scheduling strategies increase with the arrival rate. This is because the increase in the arrival rate results in the queuing of requests, which causes the response times to increase. Each request undergoes the four stages, as illustrated in Fig. 2. However, they operate in parallel; hence, the response times do not increase in multiples of the increase in the arrival rate. For any arrival rate, the inadequate utilization of the optical processor renders the performance under the IS strategy inferior to that under the CAS strategy. However, the improvement in the utilization rate under the IS strategy with the increase in the arrival rate causes their ratio to decrease overall.

The response times under the two strategies are plotted in Fig. 9 as the computation increases from 10 to 50 G, and the other parameters are unchanged. The response time under the CAS strategy is faster than that under the IS strategy for the same arrival rate and computation.

6.2 Influence on the response time of *n* and *B*

The response times under the two scheduling strategies are presented in Table 3 and Fig. 10 when n and B are varied from 1 to 4. The response times under the two strategies are almost identical when n and B are both equal to 1. Moreover, the IS strategy slightly outperforms the CAS strategy when the arrival rate is less than 30 and the latter slightly outperforms the former when it is greater than or equal to 30 because under the IS strategy, a higher arrival rate leads to constant interruption of the processing of the requests, which degrades the performance. In contrast, the increase of B can reduce the scheduling number under the CAS strategy, which slightly improves the system performance. In addition, under the IS strategy, the increase of n reduces the utilization rate of the optical processor, which causes the response time to increase substantially. Therefore, the increase of n and B will reduce the performance under the IS strategy and improve it under the CAS strategy. In addition, the performance of TOC is approximately optimal when the optical processor with 1024 data bits is divided equally into four smaller processors, and the influence on the response time of the scheduling vector, namely $\mathscr{B}$, is very small because of the low traffic of TS.

Finally, the consistency between the numerical simulation and analytical results is very high, which demonstrates the validity and feasibility of our analytical model.

7 Conclusions

Performance analysis and evaluation constitute a significant aspect of optical computing, especially TOC, which is of crucial interest to investigators and customers. In this paper, we have proposed an analytical technique that is based on a queuing system for the performance evaluation of TOC. In addition, due to the properties of the TOC optical processor, we have presented IS and CAS strategies and discussed in detail request scheduling algorithms and processor allocation algorithms under the two strategies. Moreover, we have investigated the approaches for computing response times not only under the IS strategy based on M/M/1 and M/M/n queuing systems but also under the CAS strategy based on M/M/1, M^X/M/1 and M/M^B/1 queuing systems.

To evaluate the model, we have conducted numerical simulations and experiments. Simulation and numerical results demonstrate that the proposed model and computing approaches yielded highly accurate results for the response times of TOC. Our results also demonstrate that TOC under the CAS strategy outperforms

TOC under the IS strategy, and the performance is optimal when the optical processor, which has 1024 data bits, is divided equally into four smaller processors.

In the future, we will extend our model to burst arrivals of requests. In addition, we plan to focus on other performance indicators, such as the blocking probability, the probability of immediate service and the mean number of tasks in the system.

Acknowledgements This work was supported by the Project from National Natural Science Foundation of China under Grant 61672006, in part by the Key Project of Natural Science Research in Anhui under Grant KJ2015A182 and KJ2017A340, in part by the Scientific Research Project from Fuyang Normal University under Grant 2017FSKJ12, in part by the innovation team from Fuyang Normal University under Grant XDHXTD201703 and in part by the Horizontal Project under Grant XDHX2016021. And we would like to thank the reviewers for their beneficial comments and suggestions, which improves the paper.

References

1. Shamir J (2013) Half a century of optics in computing—a personal perspective. Appl Opt 52(4):600–612
2. Yue T, Suo JL, Xiao YD, Zhang L, Dai QH (2014) Image quality enhancement using original lens via optical computing. Opt Express 22(24):29515–29534
3. Jiang YS, Devore PT, Jalali B (2016) Analog optical computing primitives in silicon photonics. Opt Lett 41(6):1273–1276
4. Jin Y, He HC, Lü YT (2003) Ternary optical computer principle. Sci China Ser F Inf Sci 46(2):145–150
5. Jin Y, He HC, Lü YT (2005) Ternary optical computer architecture. Phys Scr 118(T118):98–101
6. Yan JY, Jin Y, Zuo KZ (2008) Decrease-radix design principle for carrying/borrowing free multi-valued and application in ternary optical computer. Sci China Ser F Inf Sci 51(10):1415–1426
7. Wang XC, Peng JJ, Li M, Shen ZY, Ouyang S (2010) Carry-free vector-matrix multiplication on a dynamically reconfigurable optical platform. Appl Opt 49(12):2352–2362
8. Song K, Yan LP (2012) Design and implementation of the one-step MSD adder of optical computer. Appl Opt 51(7):917–926
9. Jin Y, Shen YF, Peng JJ, Xu SY, Ding GT, Yue DJ, You HH (2010) Principles and construction of MSD adder in ternary optical computer. Sci China Inf Sci 53(11):2159–2168
10. Peng JJ, Shen R, Jin Y, Shen YF, Luo S (2014) Design and implementation of modified signed-digit adder. IEEE Trans Comput 63(5):1134–1143
11. Wang XC, Peng JJ, Ouyang S (2011) Control method for the optical components of a dynamically reconfigurable optical platform. Appl Opt 50(5):662–670
12. Song K, Jin Y (2011) Overall plan and design of the task management system of ternary optical computer. J Shanghai Univ (Engl Ed) 15(5):467–472
13. Wang XC, Yao YF, Wang CS, Wang KZ (2012) Dynamic data-bit allocation of a ternary optical computer. Appl Mech Mater 109:181–186
14. Wang XC, Yao YF, Wang CS, Sun WW, Wang KZ (2013) Processor allocation of a ternary optical computer. Adv Sci Lett 19(6):1714–1717
15. Wang XC, Zhang SL, Zhang M, Zhao J, Niu XY (2017) Performance analysis of a ternary optical computer based on M/M/1 queueing system. In: International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP 2017), pp 331–344
16. Vilaplana J, Solsona F, Teixidó I, Mateo J, Abella F, Rius J (2014) A queuing theory model for cloud computing. J Supercomput 69(1):492–507
17. Yang B, Tan F, Dai YS (2013) Performance evaluation of cloud service considering fault recovery. J Supercomput 65(1):426–444
18. Khazaei H, Misic J, Misic VB (2012) Performance analysis of cloud computing centers using M/G/m/m+r queuing systems. IEEE Trans Parallel Distrib Syst 23(5):936–943
19. Jaiganesh M, Ramadoss B, Kumar AVA, Mercy S (2015) Performance evaluation of cloud services with profit optimization. Proc Comput Sci 54:24–30

20. Bai WH, Xi JQ, Zhu JX, Huang SW (2015) Performance analysis of heterogeneous data centers in cloud computing using a complex queuing model. Math Probl Eng 2015(1): 15 Article ID 980945
21. Cao JW, Li KQ, Stojmenovic I (2014) Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers. IEEE Trans Comput 63(1):45–58
22. William JS (2009) Probability, Markov chains, queues and simulation: the mathematical basis of performance modeling. Princeton University Press, Princeton, pp 559–610
23. Gross D, Shortie JF, Thompson JM, Harris CM (2008) Fundamentals of queueing theory, 4th edn. Wiley, New York
24. Kleinrock L (1975) Queueing systems: theory, vol 1. Wiley, New York
25. Bhat UN (2008) An introduction to queuing theory: modeling and analysis in applications, 2nd edn. Birkhauser, New York