Computer Communications 128 (2018) 46–59

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Modeling data backups as a batch-service queue with vacations and exhaustive policy

Apoorv Saxena[*,a], Dieter Claeys[1,a,b,c], Herwig Bruneel[a], Bo Zhang[d], Joris Walraevens[a]

[a] SMACS Research Group, Department TELIN, Ghent University, Sint-Pietersnieuwstraat 41, Gent B-9000, Belgium
[b] Department of Industrial Systems Engineering and Product Design, Ghent University, Technologiepark 903, Zwijnaarde B-9052, Belgium
[c] Industrial Systems Engineering (ISyE), Flanders Make, Belgium
[d] IBM Research AI, Yorktown Heights, NY 10598, USA

ARTICLE INFO

Keywords:
Data backup
Batch service
Queueing model
Vacation
Recovery point objective

ABSTRACT

For data backup processes to cloud infrastructure, there is a clean trade off between backing up frequently (improving data safety) and reducing resource usage (power consumption and communication cost). With rapid growth of data storage requirements in recent years, we need to find the right balance between both objectives. To explicitly address this trade off, we model a wide set of exhaustive data backup processes as a general batch service queueing model with multiple vacations and probabilistic restarts.

We study this queueing model and establish expressions for its performance measures such as system content and queue content distributions. This analysis aids in computing Quality of Service (QoS) measures of the data backup process such as the fraction of time the backup server is busy, the frequency of new connections and the age of the data at the beginning of a backup period. This enables us to quickly examine the dependence of QoS on the model parameters as well as to compute the optimal parameters in the backup process. We illustrate the latter by defining a particular cost function of a user and by framing an optimization problem.

1. Introduction

Processes which involve movement of data, including backup processes, consume much electricity and bandwidth. A significant part of this power is utilized to keep the system on in an idle state. Nguyen et al. [1] state, from the estimates of the US Environment Protection Agency, that 1.5% of the total power produced in US was used for Data Center computing. Further, they note that the ICT industry causes 2% of global CO_2 emissions. Therefore, in addition to higher electricity cost, under-utilization of computing resources leads to higher carbon footprint. Hence, new models are being proposed, such as in Guan et al. [2], to minimize the energy usage in data centers. Studies such as Chen et al. [3] have analyzed that the requirement of data storage technologies will grow exponentially and may grow to as high as 40 ZB by 2020 from 1.8 ZB in 2011. With this rapid increase machines need to schedule data backup processes efficiently and cost effectively.

Organizations have started moving from local disks towards cloud systems as primary storage nodes to run operations. Amazon [4] provides a list of major companies using its cloud services which includes some very well-known companies such as Netflix and Thomson Reuters. This move is driven by many remarkable benefits of the cloud infrastructure such as ease of setup, higher reliability, availability, security and protection from regional calamities or power failures (see Chang and Wills [5] for more details).

With the rapid increase in data storage requirements and migration towards cloud storage, the right balance between doing backups very frequently and reducing resource consumption has to be found. To help finding this balance, we model a wide set of data backup processes as a generic batch service queueing model with vacations, exhaustive service and probabilistic restarting conditions. As we will explain in Section 2, batch service queueing models with vacations provide a very natural way of modeling of data backups to cloud infrastructure. We are able to precisely compute the Quality of Service (QoS) measures of the data backup processes using this model. Nevertheless, to the best of our knowledge, data backup processes have been rarely modeled and analyzed as queueing processes. Yu et al. [6] propose a queueing model of cloud based streaming services to evaluate service quality. Van de Ven et al. [7] model data backups as a two dimensional Markov chain and

* Corresponding author.
E-mail addresses: saxena.apoorv@ugent.be (A. Saxena), Dieter.Claeys@UGent.be (D. Claeys), Herwig.Bruneel@UGent.be (H. Bruneel), zhangbo@us.ibm.com (B. Zhang), Joris.Walraevens@UGent.be (J. Walraevens).
[1] www.FlandersMake.be

https://doi.org/10.1016/j.comcom.2018.07.014
Received 23 March 2018; Received in revised form 6 June 2018; Accepted 6 July 2018
Available online 18 July 2018
0140-3664/

study optimization of the network bandwidth utilization. Xia et al. [8] present decision algorithms, utilizing Markov decision processes, which use constraints on recovery parameters to decide when to backup and which type of backup to perform.

Queueing models with vacations, in general, have been studied extensively in the literature (see Doshi [9] and Naishuo and George [10] for details about some applications). In particular, some relevant batch service queueing models with vacations have been studied. Sikdar and Gupta [11] study a server which goes into a vacation when the system is empty and restarts service on return if there is at least one customer in the system. Arumuganathan and Jeyakumar [12] study a batch service model with multiple vacations where the server requires a minimum threshold of queue size to keep the server on and another threshold to restart it after the vacation. The model also takes into account a setup and close down time for the server. Other models studied by Sikdar and Gupta [13] and Lee et al. [14] consider a single vacation case with a threshold where the system waits in idle condition after returning from a vacation if the threshold is not matched. The queueing models listed above have not been built and studied from the perspective of data backup processes. Our model, on the other hand, is specifically designed to model a wide range of exhaustive data backup processes.

The purpose of our paper is to model exhaustive data backup processes as a very general batch service queueing model with vacations and probabilistic restarting conditions. It is already well known that discrete-time models are more appropriate to model telecommunication processes (see Bruneel and Kim [15]). We thoroughly analyze the discrete time queueing model and obtain various performance measures. We then explicitly translate performance measures of the queueing model to QoS measures of data backup processes. Our model and the QoS measures are useful to find the right balance between doing cloud backups frequently enough (to increase data safety) and reducing resource usage (power consumption and communication costs). We study the dependence of QoS measures on the model parameters as well as compute the values of these parameters that minimize the user cost function. Note that, as stated in Chen and Trivedi [16], pure threshold policies may not be optimal. Therefore, we define parameters for probabilistic restarts to model the behavior of the server. Moreover, since the cost of connecting to cloud infrastructure is typically independent of the amount of data backed up, our work focuses on exhaustive policies (see eg the cost structure of Amazon Web Services [4] and Microsoft Azure Storage [17]).

This paper presents an extension of our previous work Claeys et al. [18]. The model studied by Claeys et al. [18] assumes single slot vacations and single slot service times. Our model extends the vacation length and service lengths to general distributions as well as introduces probabilistic restarting conditions. Making the model much more general implies that it models a wider set of exhaustive data backup policies. Additionally, this work looks at significantly more performance measures and a more thorough modeling of data backups.

We characterize our model using four types of parameters: l (threshold to begin service with probability 1), c (capacity of backup server), $\alpha_1, \alpha_2, \ldots, \alpha_{l-1}$ (starting probabilities) and T (a random variable which governs the distribution of vacation lengths). Moreover, both the arrival distribution (the number of arrivals in a slot) and service times (which depend on the batch size) can be chosen freely provided that the system remains stable. These flexible parameters empower us to set their values based on the traffic seen by the server and required system performance. Our aim is to answer questions such as: given the service level agreements, what are the right model parameters to maintain a stable and efficient system? Such questions can be answered accurately once we have computed the exact expressions of the main QoS measures in terms of the parameters. The main advantage of our analysis is that we can compute the QoS measures for a general set of parameters.

We start with a description of the model and justify its suitability for data backups in Section 2. We also discuss in detail the utility of each parameter of the model. It is then followed by the analysis of the model in Section 3. We also highlight some observations at the service/vacation epochs which give us some important relations to solve our model. Using the results of Section 3, we deduce the QoS measures of data backups in Section 4. We compute measures such as the age of data at the beginning of a backup period and the probability that the backup server is busy in a slot. We also evaluate the performance of the system for different system parameters in Section 5. Further, we construct an optimization problem to highlight how one can compute the optimal parameters to minimize user cost. In the last section we summarize our observations and results from previous sections and present possible directions for future work.

2. Backup process model

In data backup processes, packets that have not yet been backed up are part of the backlog. The backlog size then corresponds to the number of packets that have not yet been backed up. In case of exhaustive backup service processes, when the backup server initiates a backup, it continues backing up until the backlog size becomes zero.

We model the backlog as a queue where the customers represent data packets and the backup server by a batch server. As a result, packets arriving in the queue correspond to newly generated packets that have not yet been backed up.

The batch server has a capacity of maximum c packets and it employs an exhaustive policy i.e. the server keeps serving until the system content becomes zero. At that moment, the backup server immediately goes into energy saving mode which is modeled as a vacation of T slots with T a random variable. If upon returning from the vacation, the system content (i.e. the total number of packets in the queue and the server) is larger than or equal to l, the server immediately initiates the service (a new backup); otherwise it starts the service with a probability α_i, where i is the number of packets in queue. With probability $1 - \alpha_i$ the server goes into another vacation i.e. stays in the energy saving mode. Lengths of vacation periods are independent and identically distributed.

2.1. Data storage on cloud infrastructures

A cloud data backup process in an organization involves several components as illustrated in Fig. 1. This setup is commonly used across the industry to perform their data backups such as services offered by company host-it [19]. Generally, the cloud infrastructure services are provided by an external organization such as Amazon [4], Microsoft [17]. The namenode drives and monitors all the data processes running on the cloud infrastructure while the data nodes store the data fragments of all the users. All the devices in the organization communicate over LAN and the data generated is kept in sync with the data packets queue (see Fig. 1). The local backup server coordinates the backup of the data packets in the data packets queue to the cloud infrastructure. It communicates with the namenode of the cloud infrastructure and drives the whole backup process. It is exactly the data packets queue and the backup server that are modeled by our queueing model. Fig. 2 highlights the components of our queueing model and connects it with the cloud data backup process of Fig. 1.

When the backup server wants to write data, it sends a request to the namenode. Namenode stores the metadata of the data to be stored and sends back the IP-addresses of the data nodes on which the backup server is allowed to write the data. A data packet completes its service when it has been written to the cloud infrastructure. Our work assumes that the backups performed are incremental in nature, i.e., only the changes in the system from the last copy are saved and backed up.

Now we discuss the model choices and the importance of the model parameters and characteristics.

Why is it a batch service model?

Data packets are segmented and transmitted in batches from the

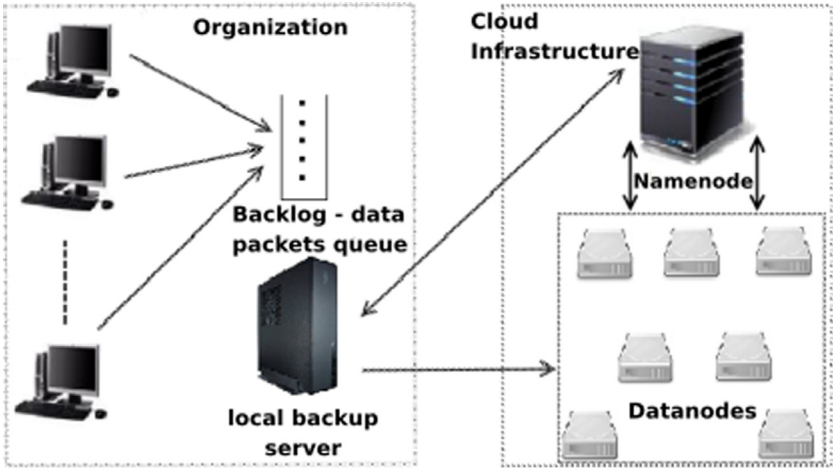


Fig. 1. Components of the backup process.

local backup server to the cloud to improve the efficiency of the transmission. (see Fry [20], Zomaya and Sakr [21] for details)

Why is the backup service exhaustive?

A new connection is established when the server communicates with the namenode of the cloud service provider, that is, when a new backup is initiated. Cloud service providers charge cloud users for each new connection. Since the cost of a new connection is independent of the amount of data to be backed up, an exhaustive service policy comes naturally. The details of cost structure of cloud service providers are available at Amazon [4] and Microsoft [17].

Why are starting probabilities included in the model?

In contrast to a fixed threshold policy, probabilistic restarting conditions allow the system to restart service even when the threshold l has not been met. It is important to note that fixed threshold based policies are just a special case of our model parameters, i.e., $\alpha_i = 0, \forall i < l$ where l is the threshold. Therefore, including the restarting probabilities makes the model more generic and encourages a much wider set of exhaustive backup policies.

Further, randomization has often been used to achieve an improvement in the efficiency of the system. For instance Boullery et al. [22] use randomization to ease-off the traffic for data backups and therefore reduce the peak load. Gkantsidis and Rodriguez [23] use randomization to improve the efficiency of distribution of data blocks over a network. Similarly, by including probabilistic restarts in our model, we are able to achieve a trade-off between contrasting QoS measures. This is also illustrated in Section 4.4.

Why are vacations included in the model?

Data processes are very power intensive and a significant amount is spent to keep the system idle while it waits for packets to arrive. As stated in Section 1 from Nguyen et al. [1], 1.5% of total power produced in the US was used for Data center computing. By introducing vacations, the local backup server sleeps during the idle periods and power consumption of the backup server is reduced. Moreover, by modeling vacation length as a random variable, we allow our model to incorporate more randomization which makes it, again more generic. Clearly, deterministic fixed length vacations is just a special case of this model parameter.

2.2. Recovery point objective

Recovery point objective (RPO) is one of the most crucial parameters defined in the literature of data backup processes. Recovery point is defined as the time point in history such that the system is recoverable up to that point in case of an immediate disaster. The RPO gives us a bound on the age of the recovery point i.e. the difference between current time and recovery point (see Druva [24] for more details). Therefore, RPO defines a bound on the waiting time of data packets during a data backup cycle where a data backup cycle is defined as consecutive vacation and service periods. In a stable system, the data packet which arrives first in a data backup cycle suffers maximum wait for restart of backup process. In paragraph 4.4, we therefore compute the waiting time of this data packet as ensuring bounds on this waiting

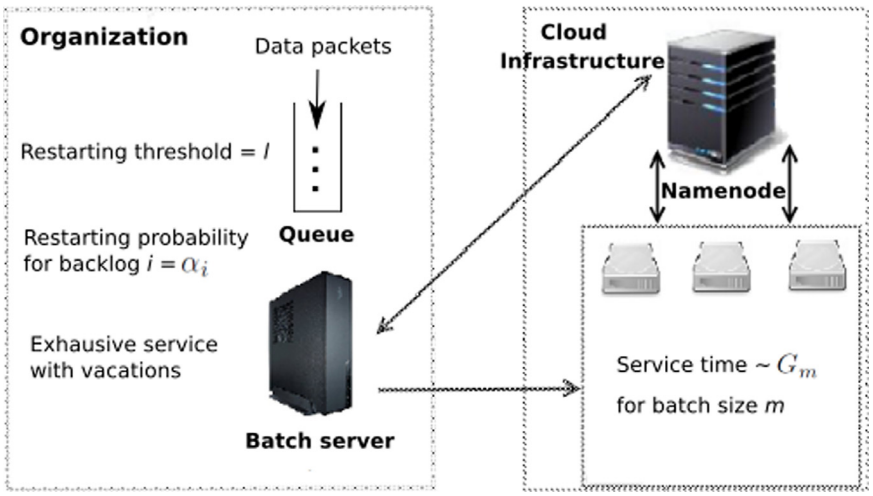


Fig. 2. Queueing model of Cloud data backup process.

time is crucial to satisfy the RPO.

Now we introduce the variables used for the analysis of the model.

2.3. Definitions and assumptions

- c : capacity of the backup server; defines the maximum number of packets that can be served in a single batch.
- k : this subscript defines the slot number.
- l : threshold for service start. When the system wakes up and finds $\geq l$ packets the server resumes service.
- α_i : probability that a backup process is started in presence of exactly i packets with $\alpha_0 = 0$, $0 \leq \alpha_i \leq 1$, $\forall\ 0 < i < l$ and $\alpha_i = 1$, $\forall\ i \geq l$.
- Q_k : queue content (the number of packets in the queue, excluding the ones receiving service) during the kth slot.
- S_k : server content (the number of packets in service) during the kth slot. Since $S_k = 0$ when the backup service is in vacation, we use this event as a proxy to check if the backup is running.
- G_m : service time required for a batch of m packets. This stochastic variable has a (general) generating function $G_m(z)$.
- R_k : remaining time in slot k of the current service/vacation period. When the server is serving packets it is the remaining serving time of those packets; otherwise, it is the remaining vacation time of the current vacation.
- T : vacation length. This variable has a generating function $T(z)$.
- A_k : number of arrivals in slot k. It has a generating function $A(z)$. Hence, we assume independent and identically distributed (i.i.d.) arrivals in each slot. However, this can be any general distribution.
- ρ : load of the system defined by $\frac{A'(1)G_c'(1)}{c}$.

All the terms defined above are discrete variables. Without loss of generality, in the analysis we have assumed that $l \geq c$. This is because a model with $l = l_0$, $l_0 < c$ can be rewritten as another equivalent model with $l = c$ and $\alpha_i = 1$, $\forall\ l_0 \leq i < c$ while keeping all the remaining parameters the same. Further, we assume that the probability of zero arrivals in a slot $A(0) > 0$. Otherwise, the backlog would never be empty and the system would never enter a vacation.

3. Analysis of the model

We study the discrete time model of the system defined in Section 2 using the definitions from paragraph 2.3. We calculate the joint generating function of the queue content, server content and remaining service time at a random slot boundary. From this generating function we deduce some of the performance measures of the model such as the system content at a random slot boundary and the number of customers served in a random batch.

This analysis is critical to be able to compute the QoS measures of the data backup process in Section 4. It also helps us to select the starting parameters. We illustrate the selection of parameters using an optimization problem in Section 5.3.

3.1. Transition equations

We relate the random vectors (Q_k, S_k, R_k) and $(Q_{k+1}, S_{k+1}, R_{k+1})$. The relations are governed by following observations

- If the remaining service/vacation time of the current batch is more than one slot ($R_k > 1$), then there is no change in the system in slot $k+1$ other than new arrivals ($Q_{k+1} = Q_k + A_k$).
- If the server is serving during slot k and the remaining service time is equal to 1 slot ($R_k = 1$) and there are a positive number of packets in queue at the end of the slot ($Q_k + A_k > 0$), then the server starts service of a new batch in slot $k+1$ due to the exhaustive service policy.
- If the remaining service/vacation time is equal to 1 slot ($R_k = 1$) and there are no packets in the queue at the end of the slot ($Q_k + A_k = 0$), the server goes for a vacation starting in slot $k+1$.
- If the server is in vacation and is about to wake up in the next slot ($R_k = 1$), the server begins service with probability α_i in slot $k+1$, where i is the number of packets in queue at the end of the slot k ($i = Q_k + A_k$).

This leads to the following relations

$$\begin{pmatrix} Q_{k+1} \\ S_{k+1} \\ R_{k+1} \end{pmatrix} = \begin{cases} (m,\ S_k,\ R_k - 1) & \text{if } R_k > 1,\ Q_k + A_k = m \\ (0,\ m,\ G_m) & \text{if } R_k = 1,\ S_k > 0,\ Q_k + A_k = m,\ 0 < m < c \\ (m - c,\ c,\ G_c) & \text{if } R_k = 1,\ S_k > 0,\ Q_k + A_k = m,\ c \leq m < l \\ (0,\ 0,\ T) & \text{if } R_k = 1,\ Q_k + A_k = m,\ m = 0 \\ (m - c,\ c,\ G_c) & \text{if } R_k = 1,\ Q_k + A_k = m,\ m \geq l \\ & \text{if } R_k = 1,\ S_k = 0,\ Q_k + A_k = m,\ 0 < m < c \\ (m,\ 0,\ T) & \text{with probability } (1 - \alpha_m) \\ (0,\ m,\ G_m) & \text{with probability } \alpha_m \\ & \text{if } R_k = 1,\ S_k = 0,\ Q_k + A_k = m,\ c \leq m < l \\ (m,\ 0,\ T) & \text{with probability } (1 - \alpha_m) \\ (m - c,\ c,\ G_c) & \text{with probability } \alpha_m \end{cases}$$

Note that $R_k \geq 1$, because as soon as it becomes one, it either changes to the service time of next batch or a multiple slot length vacation in the next slot. For this reason we will work with $R_k - 1$ rather than R_k in the generating function.

3.2. Limiting generating function

Define,

$$V_{k+1}(z, y, x) = E\left(z^{Q_{k+1}} y^{S_{k+1}} x^{R_{k+1}-1}\right)$$

and

$$E(z^X\{\text{condition}\}) = E[z^X|\text{condition}]Pr(\text{condition}).$$

Using the transition equations defined in Section 3.1, one can write

$$\begin{aligned} V_{k+1}(z, y, x) \;=\; & E\left(z^{Q_k + A_k} y^{S_k} x^{R_k - 1 - 1}\{R_k > 1\}\right) + E\left(x^{T-1}\{R_k = 1,\ Q_k + A_k = 0\}\right) \\ & + \sum_{m=1}^{c-1} E\left(y^m x^{G_m - 1}\{R_k = 1,\ S_k > 0,\ Q_k + A_k = m\}\right) \\ & + \sum_{m=c}^{l-1} E\left(z^{m-c} y^c x^{G_c - 1}\{R_k = 1,\ S_k > 0,\ Q_k + A_k = m\}\right) \\ & + \sum_{m=l}^{\infty} E\left(z^{m-c} y^c x^{G_c - 1}\{R_k = 1,\ Q_k + A_k = m\}\right) + \sum_{m=1}^{c-1} E\Bigg(\Big(z^m x^{T-1}(1 - \alpha_m) \\ & + y^m x^{G_m - 1}\alpha_m\Big)\{R_k = 1,\ S_k = 0,\ Q_k + A_k = m\}\Bigg) + \sum_{m=c}^{l-1} E\Bigg(\Big(z^m x^{T-1}(1 - \alpha_m) \\ & + z^{m-c} y^c x^{G_c - 1}\alpha_m\Big)\{R_k = 1,\ S_k = 0,\ Q_k + A_k = m\}\Bigg) \end{aligned}$$

which can be transformed in

$$\begin{aligned} V_{k+1}(z,y,x) &= \frac{A(z)}{x}E\left(z^{Q_k}y^{S_k}x^{R_k-1}\right) - \frac{A(z)}{x}E\left(z^{Q_k}y^{S_k}x^{R_k-1}\{R_k=1\}\right) \\ &+ E\left(x^{T-1}\{R_k=1,\ Q_k+A_k=0\}\right) \\ &+ \sum_{m=1}^{c-1} E\left(y^m x^{G_m-1}\{R_k=1,\ S_k>0,\ Q_k+A_k=m\}\right) \\ &+ \sum_{m=c}^{l-1} E\left(z^{m-c}y^c x^{G_c-1}\{R_k=1,\ S_k>0,\ Q_k+A_k=m\}\right) \\ &+ \sum_{m=l}^{\infty} E\left(z^{m-c}y^c x^{G_c-1}\{R_k=1,\ Q_k+A_k=m\}\right) \\ &+ \sum_{m=1}^{c-1} E\Bigg(\left(z^m x^{T-1}(1-\alpha_m) + y^m x^{G_m-1}\alpha_m\right)\Bigg\{R_k=1, \\ &S_k=0,\ Q_k+A_k{=}m\}) + \sum_{m=c}^{l-1} E\Bigg(\Bigg(z^m x^{T-1}(1-\alpha_m) \\ &+ z^{m-c}y^c x^{G_c-1}\alpha_m\Bigg)\{R_k=1,\ S_k=0,\ Q_k+A_k=m\}\Bigg). \end{aligned} \tag{1}$$

In heavy traffic conditions, the system never takes vacations and the batch size is always the maximum c. Therefore, for the system to remain stable, the average amount of work arriving during a service period should be less than the capacity of the batch server. The stability condition of this model is therefore given by $A'(1)G_c'(1) < c$, i.e. $\rho < 1$. The vacations do not play a role in the stability condition which is a standard results in queueing systems with vacation. Powell and Humblet [25] prove the stability condition of such queueing systems by proving that the underlying Markov chain of the system content is ergodic under the condition $\rho < 1$. Under the assumption that $\rho < 1$, we look at the state of the system variables in steady state.

Define

$$d_1(m) = \lim_{k\to\infty} Pr(Q_k + A_k = m,\ R_k = 1,\ S_k > 0), \tag{2}$$

$$d_0(m) = \lim_{k\to\infty} Pr(Q_k + A_k = m,\ R_k = 1,\ S_k = 0), \tag{3}$$

$$\begin{aligned} d(m) &= \lim_{k\to\infty} Pr(Q_k + A_k = m,\ R_k = 1) = d_0(m) + d_1(m), \\ V(z,y,x) &= \lim_{k\to\infty} V_k(z,y,x). \end{aligned} \tag{4}$$

Our first aim is to write all the expressions in terms of the unknown boundary probabilities $\mathbb{U} = \{d(0), d(1), d(2), \ldots, d(c-1), d_0(1), d_0(2), \ldots, d_0(l-1)\}$ and then formulate a linear system to solve for these unknowns. From (1), by letting $k \to \infty$ and replacing the terms by their generating functions, we can write

$$\begin{aligned} V(z,y,x) &= \frac{A(z)}{x}V(z,y,x) - \frac{A(z)}{x}V(z,y,0) + \frac{T(x)}{x}d(0) + \sum_{m=1}^{c-1} y^m \frac{G_m(x)}{x}d(m) \\ &+ \left(\frac{y}{z}\right)^c \frac{G_c(x)}{x} \lim_{k\to\infty} \sum_{m=0}^{\infty} z^m Pr(Q_k + A_k = m,\ R_k = 1) \\ &- \left(\frac{y}{z}\right)^c \frac{G_c(x)}{x} \sum_{m=0}^{c-1} z^m d(m) + \sum_{m=1}^{c-1} \frac{(z^m T(x) - y^m G_m(x))}{x}(1-\alpha_m)d_0(m) \\ &+ \sum_{m=c}^{l-1} \left(z^m \frac{T(x)}{x} - z^{m-c}y^c \frac{G_c(x)}{x}\right)(1-\alpha_m)d_0(m). \end{aligned} \tag{5}$$

Since $\sum_{m=0}^{\infty} z^m Pr(Q_k + A_k = m,\ R_k = 1)$ is easily computed, we have

$$\begin{aligned} \lim_{k\to\infty} \sum_{m=0}^{\infty} z^m Pr(Q_k + A_k = m,\ R_k = 1) &= \lim_{k\to\infty} A(z)V_{k+1}(z,1,0) \\ &= A(z)V(z,1,0). \end{aligned} \tag{6}$$

Using the relation (6) we can rewrite (5) as

$$\begin{aligned} V(z,y,x)(x - A(z)) &= -A(z)V(z,y,0) + \sum_{m=1}^{c-1} (y^m G_m(x) - z^{m-c}y^c G_c(x))d(m) \\ &+ \left(\frac{y}{z}\right)^c A(z)V(z,1,0)G_c(x) + \left(T(x) - \left(\frac{y}{z}\right)^c G_c(x)\right)d(0) \\ &+ \sum_{m=1}^{c-1} (z^m T(x) - y^m G_m(x))(1-\alpha_m)d_0(m) \\ &+ \sum_{m=c}^{l-1} (z^m T(x) - z^{m-c}y^c G_c(x))(1-\alpha_m)d_0(m). \end{aligned} \tag{7}$$

From this equation, we can find expressions for $V(z, y, 0)$ and $V(z, 1, 0)$ by different substitutions of x, y and z. Substituting $x = A(z)$ followed by $y \to 1$ gives us

$$\begin{aligned} A(z)V(z,1,0)(z^c - G_c(A(z))) &= (z^c T(A(z)) - G_c(A(z))d(0) \\ &+ \sum_{m=1}^{c-1} (z^c G_m(A(z)) - z^m G_c(A(z)))d(m) \\ &+ \sum_{m=1}^{c-1} (z^m T(A(z)) - G_m(A(z)))z^c(1-\alpha_m)d_0(m) \\ &+ \sum_{m=c}^{l-1} (z^c T(A(z)) - G_c(A(z)))z^m(1-\alpha_m)d_0(m). \end{aligned} \tag{8}$$

Similarly, using the substitution $x = A(z)$ in (7) yields

$$\begin{aligned} A(z)V(z,y,0) &= A(z)V(z,1,0)G_c(A(z))\left(\frac{y}{z}\right)^c - \Bigg(\left(\frac{y}{z}\right)^c G_c(A(z)) \\ &- T(A(z))\Bigg)d(0) + \sum_{m=1}^{c-1} (y^m G_m(A(z)) - z^{m-c}y^c G_c(A(z)))d(m) \\ &+ \sum_{m=1}^{c-1} (z^m T(A(z)) - y^m G_m(A(z)))(1-\alpha_m)d_0(m) \\ &+ \sum_{m=c}^{l-1} (z^m T(A(z)) - z^{m-c}y^c G_c(A(z)))(1-\alpha_m)d_0(m). \end{aligned} \tag{9}$$

Clearly, one can use (8) in the latter expression and get an expression for $V(z, y, 0)$ in terms of the unknown boundary probabilities $\mathbb{U}$. This expression would be useful to compute an expression of system content in Section 3.3. In the remaining part of this section, our primary aim is to form a solvable linear system of equations in $\mathbb{U}$. As a step towards this objective, we look at the generating function of the system content.

3.3. Generating function of system content

The system content in any time slot is defined as the number of packets either in queue waiting for service or currently in service. Therefore, its generating function is given by $V(z, z, 1)$. In (7), replacing $x = 1$, and $y = z$ gives us,

$$(1 - A(z))V(z,z,1) = -A(z)V(z,z,0) + A(z)V(z,1,0). \tag{10}$$

To further solve (10), replace $x = A(z)$ and $y = z$ in (7) which gives us

$$\begin{aligned} A(z)V(z,z,0) &= G_c(A(z))A(z)V(z,1,0) + (T(A(z)) - G_c(A(z)))d(0) \\ &+ \sum_{m=1}^{c-1} z^m (G_m(A(z)) - G_c(A(z)))d(m) \\ &+ \sum_{m=1}^{c-1} z^m (T(A(z)) - G_m(A(z)))(1-\alpha_m)d_0(m) \\ &+ \sum_{m=c}^{l-1} z^m (T(A(z)) - G_c(A(z)))(1-\alpha_m)d_0(m). \end{aligned} \tag{11}$$

Using Eqs. (8)–(11) we get an expression for $V(z, z, 1)$ in terms of the unknowns $\mathbb{U}$, which after some rewriting yields

$$(1 - A(z))(z^c - G_c(A(z)))V(z, z, 1) = B_0(z)d(0) + \sum_{m=1}^{c-1} B_m(z)d(m) + \sum_{m=1}^{c-1} W_m(z)d_0(m) + \sum_{m=c}^{l-1} R_m(z)d_0(m), \quad (12)$$

where

$$\begin{aligned} B_0(z) &= G_c(A(z))(1 - z^c)(T(A(z)) - 1), \\ B_m(z) &= G_m(A(z))G_c(A(z))(z^m - z^c) + z^m(z^c - 1)G_c(A(z)) \\ &\quad + z^c(1 - z^m)G_m(A(z)), \\ W_m(z) &= (G_m(A(z))G_c(A(z))(z^c - z^m) + z^c(z^m - 1)G_m(A(z)) \\ &\quad + G_c(A(z))T(A(z))z^m(1 - z^c))(1 - \alpha_m), \\ R_m(z) &= z^m G_c(A(z))(1 - z^c)(T(A(z)) - 1)(1 - \alpha_m). \end{aligned}$$

For a given value of z, the values $A(z)$, $B_m(z)$, $W_m(z)$ and $R_m(z)$ are known quantities. We now use a result from Adan et al. [26] to prove that $z^c - G_c(A(z))$ has c zeros inside the complex unit circle $|z| \le 1$. They have proved that if $F(z)$ is a probability generating function with $F(0) > 0$ and $F'(1) < c$ with $c \in \mathbb{N}$, then the function $z^c - F(z)$ has exactly c zeros inside the unit circle $|z| \le 1$.

Clearly, $G_c(A(z))$ is the probability generating function of the number of arrivals in a service period of c packets with $G_c(A(0)) > 0$ since $A(0) > 0$. Moreover, from the stability condition $F'(1) = G_c'(1)A'(1) < c$. Therefore, from Theorem 3.2 in Adan et al. [26], $z^c - G_c(A(z))$ has c zeros in the complex unit disk $|z| \le 1$. This includes the trivial solution $z = 1$.

Therefore, Eq. (12) has $c - 1$ zeros in the complex unit disk with $z = 1$ excluded i.e. $\{z: |z| \le 1, z \ne 1\}$. This gives us $c - 1$ linear equations in unknowns $\mathbb{U}$, however the cardinality of $\mathbb{U}$ is $c + l - 1$. Therefore, we need l more relations to form a solvable linear system of equations. For more relations, we look at *service initiation opportunities*. These opportunities are defined as the epochs of start/ end of service/ vacation times. These observations enable us to find more linear relations between the unknowns $\mathbb{U}$ which is a step further towards forming a solvable linear system in these unknowns.

3.4. Observation at service initiation opportunities

We observe the system content at the epoch points of service initiation opportunities. We start by defining a few terms which are also illustrated in Fig. 3:

- j : this subscript defines the epoch number being looked at.
- U_j = the system content at the *end* of jth epoch, i.e., the number of packets waiting in the queue or in service at the end of the epoch.
- $\tau_j = 1$, if the jth period is a service period, 0 if vacation period.

At the end of a vacation (v1), if the system has an empty backlog it will enter a new vacation (v2). This can happen iff the following conditions are true

- at the beginning of the vacation (v1), the system had an empty backlog, and
- there were no arrivals during this vacation (v1).

This gives us (13).

$$Pr(U_j = 0, \tau_j = 0) = Pr(U_{j-1} = 0)T(A(0)) \quad (13)$$

Similarly, the system wakes up with n packets ($n > 0$) in the backlog iff

- the system entered the vacation with empty backlog and there were n arrivals during the vacation, or
- the system entered the vacation with i packets present ($i > 0$) and there were $n - i$ arrivals during the vacation.

Note that, for the system to enter a vacation with i packets where $0 < i \le n$, the previous epoch also has to be a vacation epoch because of the exhaustive service policy. So for $0 < n < l$, where $t_A(n)$ denotes the probability of n arrivals in a vacation period, we have

$$Pr(U_j = n, \tau_j = 0) = \sum_{i=1}^{n} Pr(U_{j-1} = i, \tau_{j-1} = 0)(1 - \alpha_i)t_A(n - i) + Pr(U_{j-1} = 0)t_A(n). \quad (14)$$

Moreover, $t_A(n)$ is a known constant because it is the coefficient of z^n in $T(A(z))$. Under the assumption of $\rho < 1$, the system would reach a steady state and under this steady state, the limiting probabilities of these terms are defined as

$$\pi_i(m) = \lim_{j\to\infty} Pr(U_j = m, \tau_j = i),$$
$$\pi(m) = \lim_{j\to\infty} Pr(U_j = m).$$

Hence, we can write

$$\pi_0(0) = \pi(0)T(A(0)), \quad (15)$$

$$\pi_0(n) = \sum_{i=1}^{n} \pi_0(i)(1 - \alpha_i)t_A(n - i) + \pi(0)t_A(n), \ 0 < n < l, \quad (16)$$

$$\pi(n) = \pi_0(n) + \pi_1(n). \quad (17)$$

3.5. Relating π, $d_0(m)$ and $d(0)$

Now we relate the terms defined in (15)–(17) and (2)–(4). In earlier sections we defined,

$$\begin{aligned} d_0(m) &= \lim_{k\to\infty} Pr(Q_k + A_k = m, R_k = 1, S_k = 0) \\ &= \lim_{k\to\infty} Pr(Q_k + A_k = m, S_k = 0 | R_k = 1)Pr(R_k = 1). \end{aligned}$$

Observe that when $R_k = 1$, the starting boundary of slot $k + 1$ is the beginning of a service initiation opportunity epoch which has been analyzed in Section 3.4. Therefore, $\lim_{k\to\infty} Pr(Q_k + A_k = m, S_k = 0 | R_k = 1) = \lim_{j\to\infty} Pr(U_j = m, \tau_j = 0)$ and $\lim_{k\to\infty} Pr(Q_k + A_k = m | R_k = 1) = \lim_{j\to\infty} Pr(U_j = m)$. These can be used to come up with the following relations,

$$d_0(m) = \pi_0(m) \lim_{k\to\infty} Pr(R_k = 1),$$

$$d(m) = \pi(m) \lim_{k\to\infty} Pr(R_k = 1).$$

Therefore, we can translate (15)–(17) to equations in $d_0(m)$ and $d(0)$ and we obtain

$$d_0(0) = d(0)T(A(0)), \quad (18)$$

$$d_0(n) = \sum_{i=1}^{n} d_0(i)(1 - \alpha_i)t_A(n - i) + d(0)t_A(n), \ 0 < n < l. \quad (19)$$

(19) can be rewritten as,

$$B \cdot \begin{bmatrix} d(0) \\ d_0(1) \\ d_0(2) \\ \vdots \\ d_0(l-1) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad (20)$$

where B is defined as

$$B = \begin{bmatrix} -t_A(1) & 1-(1-\alpha_1)t_A(0) & 0 & \dots & 0 \\ -t_A(2) & -(1-\alpha_1)t_A(1) & 1-(1-\alpha_2)t_A(0) & \dots & 0 \\ -t_A(3) & -(1-\alpha_1)t_A(2) & -(1-\alpha_2)t_A(1) & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -t_A(l-1) & -(1-\alpha_1)t_A(l-2) & -(1-\alpha_2)t_A(l-3) & \dots & 1-(1-\alpha_{l-1})t_A(0) \end{bmatrix}. \quad (21)$$

We do not include (18) in the matrix equations because $d_0(0) \notin \mathbb{U}$.

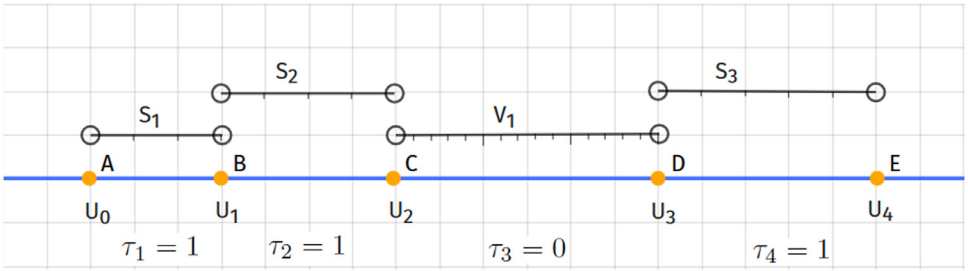


Fig. 3. Illustration of service initiation opportunities, service periods and vacation periods.

However, this equation will be useful while looking at the performance measures in Section 3.8. Summarizing, (20) gives us $(l-1)$ linearly independent equations between the unknowns in $\mathbb{U}$ which are crucial to create a solvable system.

Combining the results of Section 3.3 and 3.5, we have in total $c+l-2$ equations which is still less than the number of unknowns $(c+l-1)$. However, by the normalization property of generating functions, we know that $V(z,z,1)|_{z=1}=1$. This normalization condition, discussed in paragraph 3.6, gives us an additional relation in the unknowns. Combining all these equations together, we get a solvable system of $l+c-1$ linearly independent equations and $l+c-1$ unknowns $\mathbb{U}$.

3.6. Normalization condition

As stated above, the normality condition $V(1,1,1)=1$ plays a critical role to get a solvable linear system. This equation is obtained by applying L'Hospital's rule twice to (12) which gives us

$$\left(\frac{cT'(1)}{c-G_c'(1)A'(1)}\right)d(0)+\sum_{m=1}^{l-1}\left(\frac{cT'(1)}{c-G_c'(1)A'(1)}\right)(1-\alpha_m)d_0(m) + \sum_{m=1}^{c-1}\left(\frac{cG_m'(1)-mG_c'(1)}{c-G_c'(1)A'(1)}\right)(d(m)-(1-\alpha_m)d_0(m))=1. \tag{22}$$

To easily write the system of equations in matrix form, Eq. (22) can be rewritten as

$$\left(\frac{cT'(1)}{c-G_c'(1)A'(1)}\right)d(0)+\sum_{m=1}^{c-1}\left(\frac{mG_c'(1)+cT'(1)-cG_m'(1)}{c-G_c'(1)A'(1)}\right)(1 - \alpha_m)d_0(m)+\sum_{m=c}^{l-1}\left(\frac{cT'(1)}{c-G_c'(1)A'(1)}\right)(1-\alpha_m)d_0(m) + \sum_{m=1}^{c-1}\left(\frac{cG_m'(1)-mG_c'(1)}{c-G_c'(1)A'(1)}\right)d(m)=1. \tag{23}$$

3.7. Original generating function

In this section we show that just like the generating function of system content, it is possible to express the original generating function defined in (2) in terms of the unknowns $\mathbb{U}$. We have already written $V(z,y,x)$ in terms of $V(z,y,0)$, $V(z,1,0)$ and the unknowns $d_0(n)$s and $d(n)$s in (7). We have also shown that we can write $V(z,y,0)$ and $V(z,1,0)$ in terms of the unknowns in (8) and (9). Therefore, substituting (8) and (9) into (7) we get

$$\begin{aligned}(x-A(z))V(z,y,x) &= B_0(z,y,x)d(0)+\sum_{m=1}^{c-1}B_m(z,y,x)d(m)\\ &\quad+\sum_{m=1}^{c-1}W_m(z,y,x)(1-\alpha_m)d_0(m)\\ &\quad+\sum_{m=c}^{l-1}R_m(z,y,x)(1-\alpha_m)d_0(m).\end{aligned}$$

where

$$\begin{aligned}B_0(z,y,x) &= T(x)-T(A(z))+\frac{y^c(G_c(x)-G_c(A(z)))(T(A(z))-1)}{z^c-G_c(A(z))},\\ B_m(z,y,x) &= y^m(G_m(x)-G_m(A(z)))\\ &\quad+\frac{y^c(G_c(x)-G_c(A(z)))(G_m(A(z))-z^m)}{z^c-G_c(A(z))},\\ W_m(z,y,x) &= z^m(T(x)-T(A(z)))-y^m(G_m(x)-G_m(A(z)))\\ &\quad+\left(\frac{z^mT(A(z))-G_m(A(z))}{z^c-G_c(A(z))}\right)y^c\left(G_c(x)-G_c(A(z)),\right.\\ R_m(z,y,x) &= z^m\left(T(x)-T(A(z))+\frac{y^c(G_c(x)-G_c(A(z)))(T(A(z))-1)}{z^c-G_c(A(z))}\right).\end{aligned}$$

3.8. General performance measures of the model

In this section we write expressions of generating functions of some common stochastic variables of this queueing model in terms of $V(z,y,x)$ and input functions. Since we have shown that $V(z,y,x)$ can be expressed in terms of the boundary probabilities in the set $\mathbb{U}$, the generating functions below can also be easily written as functions of these boundary probabilities.

3.8.1. System content at random slot boundaries

As the system content includes both the customers in queue and those in service, the pgf of the system content is given by the generating function $V(z,z,1)$. Exact expression of this generating function was computed in (12).

3.8.2. System content at service initiation opportunities

We have already looked at the generating function of $d(m)$. From (6), the generating function of the system content at service initiation opportunities is

$$\frac{A(z)V(z,1,0)}{V(1,1,0)}.$$

Moreover, one can write the generating function of the system content at vacation completion times and service completion times respectively as

$$\frac{A(z)V(z,0,0)}{V(1,0,0)},\quad \frac{A(z)(V(z,1,0)-V(z,0,0))}{V(1,1,0)-V(1,0,0)}. \tag{24}$$

3.8.3. Number of customers served in a random batch

This is an important measure which reflects how efficiently the server resources are being utilized. The sample space for the number of customers in server at a slot boundary can be divided into two mutually exclusive and exhaustive events, $S_k=0$ and $S_k>0$. $S_k=0$ implies that the system is in vacation in the kth slot since server is empty. Therefore, the generating function for the server content in a random batch is given by

$$\frac{V(1,y,1)-V(1,0,1)}{1-V(1,0,1)}. \tag{25}$$

3.8.4. Queue content when the server is in vacation

Since $S_k = 0$ implies that the system is in vacation in kth slot, the queue content during such a slot is given by the generating function

$$\frac{V(z, 0, 1)}{V(1, 0, 1)}. \tag{26}$$

4. QoS measures of data backup policy

In this section we focus on the performance measures that are of interest to the backup policy. The analysis done in Section 3 directly gives us the first three QoS measures namely the backlog size, probability that a new connection is made and the probability that the backup server is busy. We also compute the age of the data at the beginning of a backup period, i.e. time spent by first data packet of a backup cycle waiting for backup to restart. In Section 5 we evaluate the performance of the system by looking at the change in the first moment of these stochastic variables versus change of load as well as versus the mean vacation time.

4.1. Backlog size

The queue content of the system is the size of the backlog of data packets which are waiting for service. A bigger backlog size puts more data at the risk of loss if the server crashes. Moreover, we have a finite capacity of buffer size in practice. Therefore, we would want to keep this quantity as small as possible. By the definition of generating function $V(z, y, x)$, the queue content has a generating function $V(z, 1, 1)$. The mean queue content equals $\frac{\partial V(z,1,1)}{\partial z}\Big|_{z=1}$.

4.2. Probability that a new connection is made

As mentioned previously, cloud service providers charge for new connections made to their infrastructure. A new connection is made whenever a new service period starts after a vacation period. Therefore, we can write the probability of a new connection at a random slot as $\lim_{k\to\infty} Pr(S_{k+1} > 0, S_k = 0)$. We can compute this probability in terms of known constants as below.

$$\begin{aligned} Pr(\text{New connection}) &= \lim_{k\to\infty}(Pr(S_k = 0) - Pr(S_{k+1} = 0, S_k = 0)) \\ &= \lim_{k\to\infty}\Big(Pr(S_k = 0, R_k = 1) + Pr(S_k = 0, R_k > 1) \\ &\quad - Pr(S_{k+1} = 0, S_k = 0, R_k = 1) \\ &\quad - Pr(S_{k+1} = 0, S_k = 0, R_k > 1)\Big) \\ &= V(1, 0, 0) - \sum_{m=0}^{l-1} \lim_{k\to\infty} Pr(Q_k + A_k = m, S_{k+1} \\ &\quad = 0, S_k = 0, R_k = 1) \\ &= V(1, 0, 0) - \sum_{m=0}^{l-1} (1 - \alpha_m) d_0(m) \end{aligned} \tag{27}$$

Note that $d_0(0) \notin \mathbb{U}$; however one can use (18) to compute the value of this boundary probability. Using (27), we can also compute the probability of a new connection at a random service initiation opportunity as,

$$\begin{aligned} &Pr(\text{New Service at Service initiation opportunity}) \\ &= \lim_{j\to\infty} Pr(\tau_j = 1 | \tau_{j-1} = 0) = \frac{Pr(\text{New connection})}{V(1, 0, 0)} \\ &= 1 - \sum_{m=0}^{l-1} \frac{(1 - \alpha_m) d_0(m)}{V(1, 0, 0)}. \end{aligned} \tag{28}$$

As for the backlog size, we would want to keep this quantity as small as possible.

4.3. Probability that the backup server is busy

If the backup server is busy for longer periods, more power would be utilized to keep it running. It would also imply that the connections established with the data nodes to serve data packets would have to be maintained for longer period. The duration for which the backup server are kept on is directly proportional to the probability that the backup server is busy. Since the backup server is busy iff it is not in vacation, the probability that it is busy is given by

$$1 - V(1, 0, 1). \tag{29}$$

4.4. Age of data at the beginning of a backup period

A data backup cycle consists of one vacation period followed by one service period. During a data backup cycle, one can observe that the first data packet arrival has to wait for the longest time for backup service to restart (illustrated in Fig. 4). We randomly select a data backup cycle and label the first data packet to arrive in that cycle as ν.

At any slot, we define the age of data based on the earliest packet in the system. It is defined as the time spent by the earliest packet waiting for restart of backup service. Clearly, under this definition if the backup service is ON or if there are no packets in the system, the age of data is 0. Moreover, the age of data attains it maximum just before the start of backup service (also illustrated in Fig. 4). Therefore, the total time spent waiting by ν is the age of data at the beginning of a backup period. We label this age as Age_ν.

In Section 2.2, we introduced the recovery point objective (RPO) which is one of the most crucial parameters defined in literature of data backup processes. It defines the maximum time any data packet can wait in the data packets queue. To meet the QoS objectives defined by RPO, it is necessary to select the backup parameters such that Age_ν is within acceptable limits. By computing this QoS measure, we compute the impact of vacations on the waiting time of data packets.

We first compute the number of vacations ν witnesses, excluding the one in which it arrives, before the service is restarted. This will help us compute the amount of time it spends waiting for the backup service to restart. Note that ν might arrive in any vacation period after the service is stopped (not necessarily the first one) but those earlier vacation periods are not relevant as they are not part of age of ν. We refer to Fig. 5 for an illustration of the delay of ν.

We observe the system at consecutive vacation epochs starting from the vacation period during which ν arrives. We use the queue content at these epochs to define an absorbing Markov Chain for this process. That is, define the state space $\Omega = \{1, 2, ..., l-1, B\}$, where in state i, the system has i packets backlogged during a vacation while ν is in the system. State B corresponds to the absorbing state in which the backup process restarts. Notice that our backup system can transit from state a to state b before being absorbed in state B, where $a \le b$, if there are exactly $b - a$ arrivals during the vacation period and the backup process does not start at the end of the vacation. The probability of this transition to happen is $(1 - \alpha_b) t_A(b - a)$ where $t_A(n)$ is defined in Section 3.4 (probability of n arrivals in a vacation period).

Since our system jumps between states until it finally gets absorbed, we can model the number of vacations seen by ν as a discrete phase-type distribution. This distribution is characterized by $(\boldsymbol{\beta}, M)$ where the state space is Ω. Such a distribution comprises of $l-1$ transient states ($\{1, 2, ..., l-1\}$) and 1 absorbing state (B) . The transition matrix $P = \begin{bmatrix} M & \boldsymbol{M^0} \\ \boldsymbol{0} & 1 \end{bmatrix}$ and $(\boldsymbol{\beta}, \beta_l)$ is the initial probability vector; M is given by

$$M = \begin{bmatrix} (1-\alpha_1)t_A(0) & (1-\alpha_2)t_A(1) & (1-\alpha_3)t_A(2) & \dots & (1-\alpha_{l-1})t_A(l-2) \\ 0 & (1-\alpha_2)t_A(0) & (1-\alpha_3)t_A(1) & \dots & (1-\alpha_{l-1})t_A(l-3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & (1-\alpha_{l-1})t_A(0) \end{bmatrix} \tag{30}$$

and vector $\boldsymbol{M^0}$ can be calculated using the fact that P is a valid

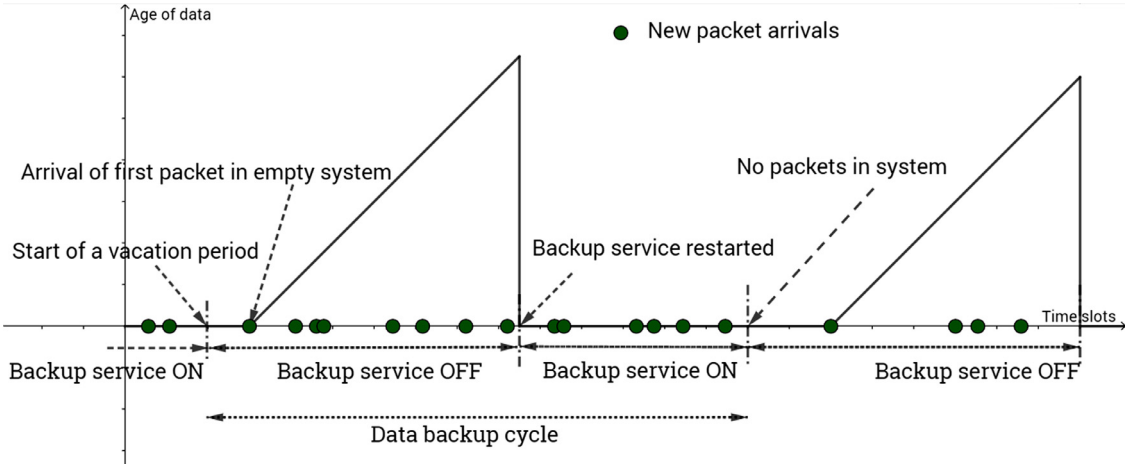


Fig. 4. Illustration of evolution of Age of data in a backup system.

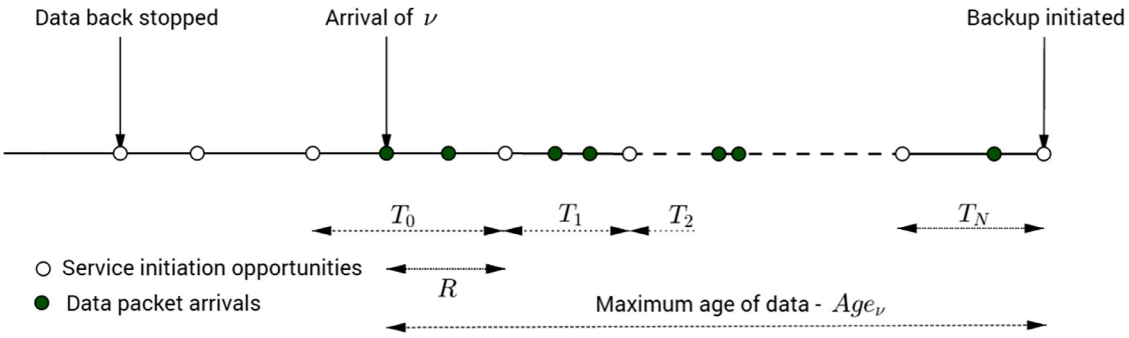


Fig. 5. Illustration of delay of ν.

probability transition matrix, i.e., rows sum to 1.

Since the system is non-empty at the beginning of first (relevant) vacation slot, our starting state is

$$\boldsymbol{\beta} = \left(\frac{(1-\alpha_1)t_A(1)}{1-t_A(0)}, \frac{(1-\alpha_2)t_A(2)}{1-t_A(0)}, \frac{(1-\alpha_3)t_A(3)}{1-t_A(0)}, \ldots, \frac{(1-\alpha_{l-1})t_A(l-1)}{1-t_A(0)}\right), \beta_l = 1 - \sum_{i=1}^{l-1}\beta_i.$$

Define N as the number of phases till absorption to B, i.e., the number of vacation epochs seen by ν. The generating function of N is given by $N(z) = z\cdot\boldsymbol{\beta}\cdot(I - zM)^{-1}\cdot\boldsymbol{M^0} + \beta_l$ where I is the identity matrix (see Alfa [27]). Using this, Age_ν can be written as

$$Age_\nu = \sum_{i=1}^{N} T_i + R,$$

where $T_i \sim T$ while R is the remaining vacation time of ν as shown in Fig. 5. Since $E(T_n 1_{\{N\geq n\}}) = E(T_n)P(N \geq n)$, we can use Wald's equation (see Johnson [28]) to get the first moment of Age_ν as

$$E(Age_\nu) = E(N) \times E(T) + E(R). \tag{31}$$

We are left with the computation of $E(R)$. We can do this by analyzing the joint probability mass function(pmf) of the length of vacation the first packet arrives in, T_0, and R. Note that the vacation time in which ν arrives, T_0 (see Fig. 5), does not have the same distribution as T. This is because there is at least one arrival in the vacation in which ν arrives. The joint pmf can therefore be written as, for $r \geq 0$, $t \geq 1$,

$$\begin{aligned} Pr(T_0 = t, R = r) &= Pr\left(T = t, R = r \middle| \sum_{i=1}^{T} A_i \geq 1\right) \\ &= \frac{Pr(T = t, R = r, \sum_{i=1}^{T} A_i \geq 1)}{Pr(\sum_{i=1}^{T} A_i \geq 1)}. \end{aligned} \tag{32}$$

Looking at Fig. 6, we can see that $R = r$ when $T = t$ if there are no arrivals in the first $t - r - 1$ slots and at least one arrival in the slot in which ν arrives. Hence (32) can be written as

$$Pr(T_0 = t, R = r) = \frac{A(0)^{t-r-1}(1 - A(0))P(T = t)}{1 - T(A(0))}. \tag{33}$$

From (33), we can calculate the first moment of the remaining vacation time by computing

$$E(R) = \sum_{r=0}^{\infty}\sum_{t=r+1}^{\infty} \frac{rA(0)^{t-r-1}(1 - A(0))P(T = t)}{1 - T(A(0))}$$

which can be simplified to

$$E(R) = \frac{T'(1)}{1 - T(A(0))} - \frac{1}{1 - A(0)}. \tag{34}$$

5. Numerical evaluation

To demonstrate the sensitivity of the QoS measures on the backup parameters, we evaluate the performance of backup system using realistic parameter values. Therefore, we create a scenario from synthetic data based on our experience working with real, yet proprietary data of an actual data backup system. We translate the backup parameters into model parameters of the batch service queueing system to analyze the performance. These measures are the key elements which can help us select the right parameters based on the performance requirements.

Backup system

We consider an organization which uses cloud services to backup the data on laptops of its employees using a centralized backup server.

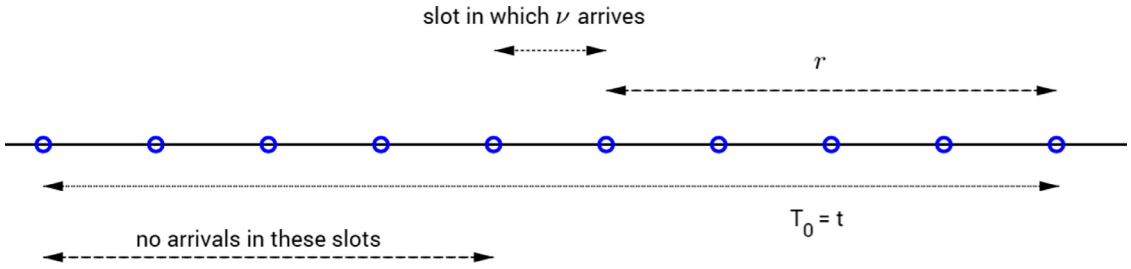


Fig. 6. Illustration of arrival of ν to compute joint distribution of T_0 and R.

This data is generated due to incoming and outgoing emails, documents created or edited etc. From our practical experience and some real data, we define the parameters of the backup process. The average data generate by all users in 1 min is between 20 to 100 MB data. The backup system on the other hand, has a bandwidth that is capable of serving at a maximum speed of 100 MB per min. The data files are zipped into data packets of size 16 MB each and backup server can serve upto 10 packets in a single batch. Moreover, once the backup server enters a vacation period, it has to restart service if there are more than 30 packets in the backlog.

Queueing model

We define our slot length is equal to 10 sec. Using the information of the backup system, the model parameters are defined below:

- It is well known that the arrival distribution has a heavy tail (see eg. Liebeherr et al. [29]) ,i.e, there are some slots which will see a big burst of arrivals of packets. Therefore, we model the arrival distribution to be a mixture of Poisson and Power law distribution. The generating function of the number of arrivals in a slot can be written as

$$\begin{aligned} A(z) &= p \times e^{\lambda(z-1)} + (1-p) \times \frac{\zeta_\gamma(z,a)}{\zeta_\gamma(1,a)} \\ \zeta_\gamma(z,a) &= \sum_{k=a}^{\infty} \frac{z^k}{k^\gamma} \end{aligned} \tag{35}$$

  where λ is the arrival rate of the Poisson process and γ is the order of power law distribution. For this example, we use $a = 25$, $p = .999$, $\gamma = 3.5$ while λ varies to model change in the system load. We also need to ensure that the system remains stable, i.e., the load of the system is less than 1.
- Since the backup server can serve upto 10 packets in a single batch, the batch server capacity $c = 10$.
- Since the backup service has to restart if there are more than 30 packets in the backlog, the restarting threshold $l = 30$.
- Since the writing speed on an average allows the backup server to write about 1 packet in a slot, we model the service time of m packets as discrete uniform distribution U$[m, m + 2]$. Therefore, the $E(G_m) = m + 1$.
- We assume that the backup server goes into vacations of length approximately equal to 5 min. Therefore, the vacation period T is modeled as $U[28, 32]$, i.e., uniformly distributed discrete random variable between 28 and 32 slots. Moreover, while analyzing the impact of vacation lengths on the model, we model T as $U[v - 2, v + 2]$ such that the expected vacation length, $E(T) = v$.

It is important to note that we have used fairly simple distributions for service time and vacation periods. However, the results obtained in the paper are valid for any general distributions of service time, vacation periods and number of arrivals in a slot defined by generating functions $G_m(z)$, $T(z)$ and $A(z)$, respectively.

5.1. Observations

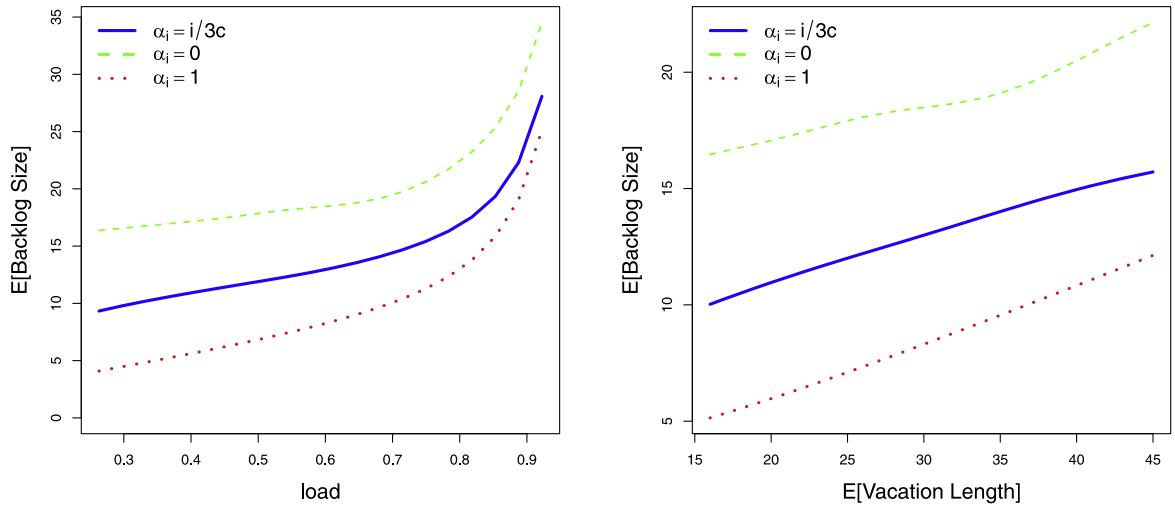
In the graphs in Figs. 7–11 we show both the performance measures of queueing model as well as the QoS measures of the backup process under varying load or vacation length keeping other parameters constant. These graphs represent the first moment of the random variables defined in Sections 3.8 and4. Three policies are compared

1. $\alpha_i = 0, \forall\ 0 < i < l$, this corresponds to the policy where the service does not start until the server threshold is met.
2. $\alpha_i = 1, \forall\ 0 < i < l$, this corresponds to the vacation policy which resumes service if there is at least one packet in the queue when the server wakes up from vacation. Equivalently, it can be defined as a policy with $l = 1$.
3. $\alpha_i = \frac{i}{3c}, \forall\ 0 < i < l$, this corresponds to the policy which starts with a probability that is linear to the number of packets in the queue.

Note that by comparing these policies, we are comparing the threshold policy with a policy which also has probabilistic restarts. Moreover, the policy with $\alpha_i = i/3c$ is an example policy that we have chosen for comparison while the model allows us to substitute any valid value. Our primary aim while doing this comparison is to observe the dependence of QoS measures on the model parameters.

While comparing these policies in Figs. 7–11, the ideal performance for each QoS measure individually would be obtained either by policy (1) or (2). In particular, while policy (1) performs best for QoS measures related to energy consumption and connection cost, policy (2) performs best for QoS measures related to data safety. However, these policies individually are not able to meet all the expectations. We observe that the policy (3) is able to tackle the trade off and gives a performance which lies in between policy (1) and policy (2). Also, the choice of parameters of our model, αs, T, l, c, allows us to get closer to the desired quality of service measures. We illustrate selection of optimal parameters in Section 5.3 by defining the cost function of a user. One can make the following general observations from the graphs

- from Figs. 7 and 8, the values of mean backlog size and the age of data at the beginning of a backup period for policy (3) is in the middle of the two policies (1) and (2). Limiting ourselves to these measures, while the system performs better than policy (1), it performs worse than policy (2).
- from Fig. 9, using policy (3), the probability of a new connection at a service initiation opportunity reduces drastically as compared to policy (2). Although, this probability under policy (3) is higher when compared to policy (1), it is closer to policy (1) than (2).
- from Fig. 10, probability that the server is busy is significantly lowered by using policy (3) as compared to policy (2). This probability is higher than the value obtained using policy (1) but fairly close to it.
- from Fig. 11, the server content in a random batch using policy (3) is higher than policy (2) and smaller than (1).



(a) E[Vacation Length] = 30 (b) Load=0.6

Fig. 7. Mean backlog size.

5.2. Illustrating the trade-off using cost function

In this section we use the QoS measures of the backup service computed in Section 4 to define a cost function for a user. Our primary aim is to address the trade-off between doing frequent backups and reducing resource usage. The frequency of performing backups is dependent on all backup parameters and there are multiple ways of changing this frequency. For instance one can increase the values of restarting probabilities or decrease the restarting threshold l. Here, we change the frequency by changing the restarting threshold l.

We define the cost function as a function of the restarting threshold l as follows

$$Cost(l) = 0.5 \times E(Age_{nu}) + 20 \times e^{Pr(\text{ New connection })} + 0.25 \times E(\text{ Backlog content}) + 15 \times e^{Pr(\text{ Server busy })} \quad (36)$$

Further, we define $\alpha_i = \min\left(1, \frac{i}{100}\right)$ for $i < l$. The remaining parameters c and ρ are kept constant at 10 and 0.6. Fig. 12 shows the change in the cost function with change in the backup frequency. Clearly, the backup frequency has to be chosen optimally which can be done by selecting the backup parameters optimally. We show how we can compute the optimal backup parameters in the next paragraph.

5.3. Cost optimization

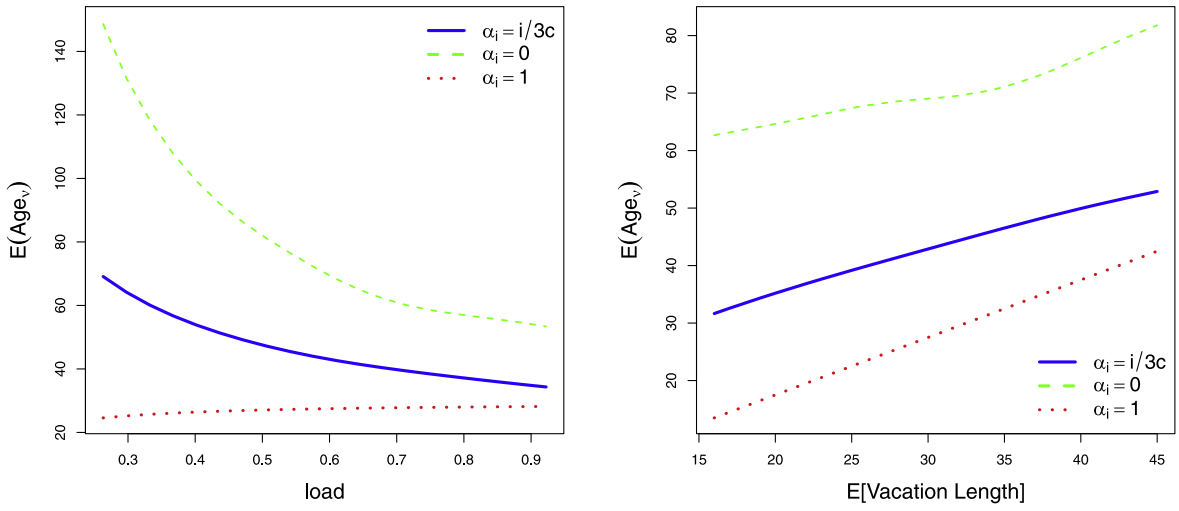
In contrast to the comparisons done in Section 5.1 between three particular policies, in this section we look at computing the optimal parameters for our model. In Section 5.2, we defined a cost function of the user to highlight the need of selecting the optimal backup parameters. However, to compute optimal parameters, it is more appropriate to define the user objective as a combination of cost function and QoS constraints. We use the four QoS measures, $E(Age_v)$, $Pr($ New connection), $E($ Backlog content), $Pr($ Server busy) computed in Section 4 to define the following cost function

$$W(l, c, \alpha_1, ..., \alpha_{l-1}) = 20 \times e^{Pr(\text{ New connection })} + 15 \times e^{Pr(\text{ Server busy })}$$

and QoS constraints of a user as

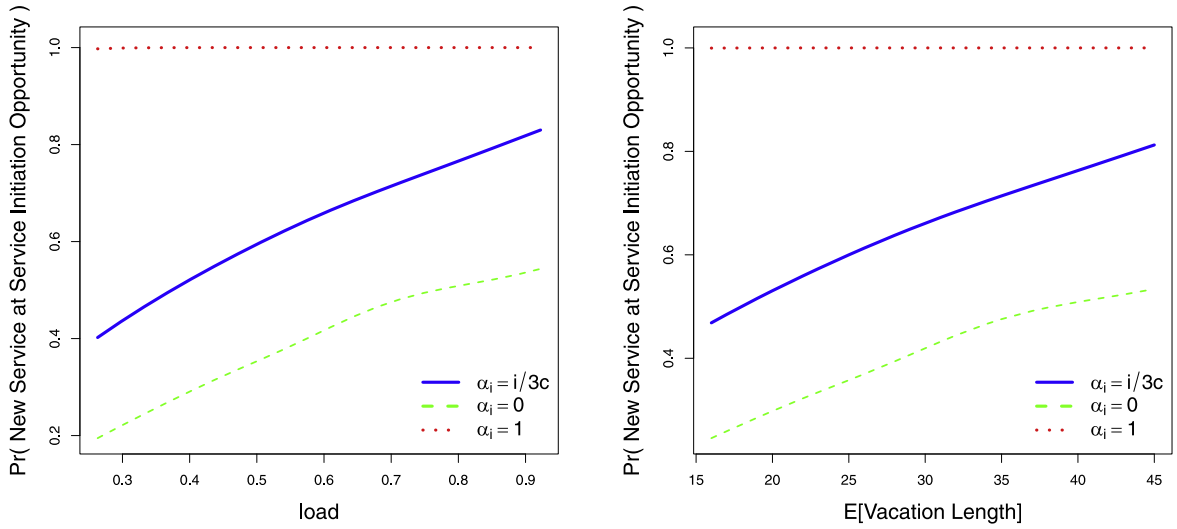
$$E(Age_v) \leq 30, E(\text{ Backlog content }) \leq 20. \quad (37)$$

For illustration purposes, we minimize this function by varying the



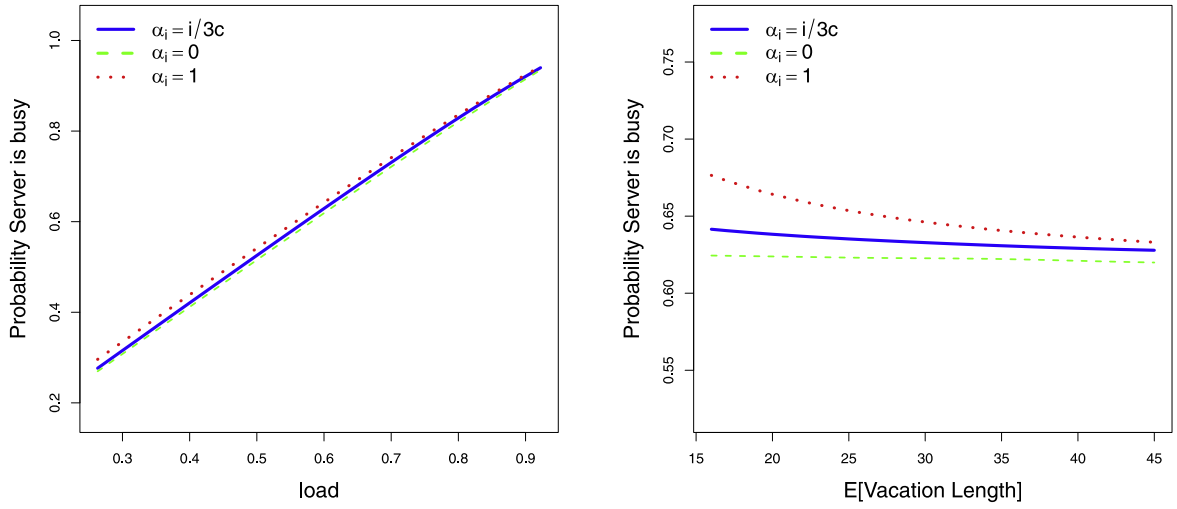
(a) E[Vacation Length] = 30 (b) Load=0.6

Fig. 8. Mean age of data at the beginning of backup cycle.



(a) E[Vacation Length] = 30 (b) Load = 0.6

Fig. 9. Probability of a new connection at a service initiation opportunity.



(a) E[Vacation Length] = 30 (b) Load=0.6

Fig. 10. Probability server busy.

starting probabilities α_is and threshold l while keeping the server capacity c constant. More precisely, we propose a simple mechanism to find the optimal l and $\alpha_1, \alpha_2, ..., \alpha_{l-1}$. Our primary aim is to show that the optimal policy may not be a pure threshold policy.

A pure threshold policy can be represented as $\alpha_i = 0, \forall\ i < m$ and $\alpha_i = 1, \forall\ i \geq m$, where m defines the threshold. Further, a pure threshold policy with threshold which minimizes cost is called the optimal pure threshold policy.

The parameters used here are as defined in the beginning of Section 5. For a fixed load ρ, the task of finding the optimal l and α_is for our model is equivalent to solving the following optimization problem

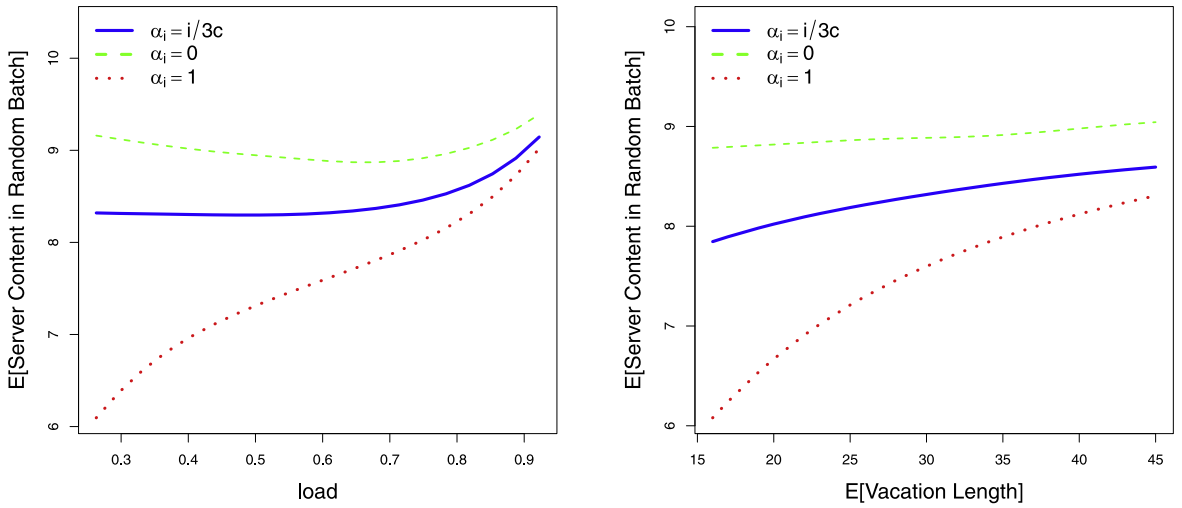
$$\begin{aligned} \underset{\alpha_1,\ldots,\alpha_{l-1},l}{\text{minimize}} \quad & W(l, \alpha_1, \alpha_2, \ldots, \alpha_{l-1}) \\ \text{subject to:} \quad & E(Age_v) \leq 30 \\ & E(\text{ mean backlog content }) \leq 20 \\ & 0 \leq \alpha_i \leq 1,\ \ i = 1, \ldots, (l-1) \\ & \alpha_i = 1,\ i \geq l,\ l \leq l_{\max} \end{aligned}$$

We use $l_{\max} = 15$ to find the solution. For a given load ρ, solution of this problem gives us the optimal parameters l^*, α_i^*s. We use an iterative solver, *auglag*() in **R**, to solve the above non-linear optimization problem. We set the starting point of the solver as the optimal pure threshold policy. Therefore, the solution found by the solver would be better than the optimal pure threshold policy. These may not be globally optimal solutions; however, we have shown that optimal threshold policies with probabilistic restarts perform better than optimal pure threshold policies with threshold l less than $l_{\max}$.

In Fig. 13 we compare the cost graph of the user using the optimal parameters with pure optimal threshold based policies. Note that we have considered a fairly simple optimization problem. One could also have constraints as well as utility function dependent on higher moments of QoS measures such as *Var*(mean backlog content) which can be derived using the generating functions computed in paragraph 3.8.

5.4. Final insights

In Section 5.1 we illustrate the dependence of QoS measures on the service parameters. In Section 5.3 we look at an optimization problem to compute the system parameters using the cost function of the user. The key observations are



(a) E[Vacation Length] = 30 (b) Load=0.6

Fig. 11. Average server content in random batch.

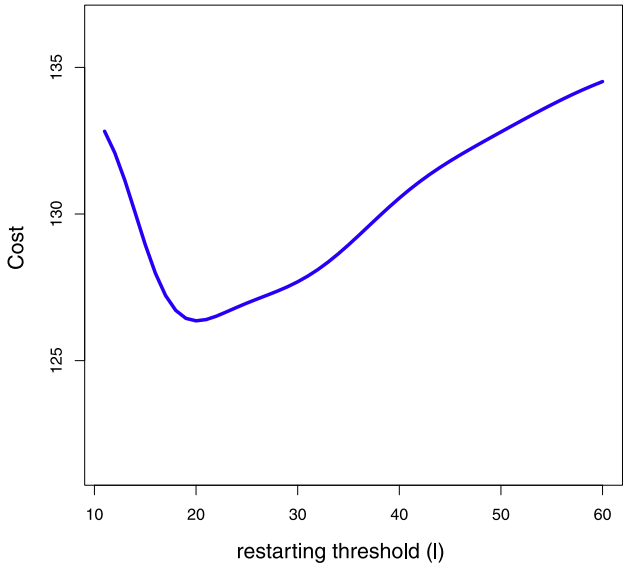


Fig. 12. Cost function for varying frequency of backups.

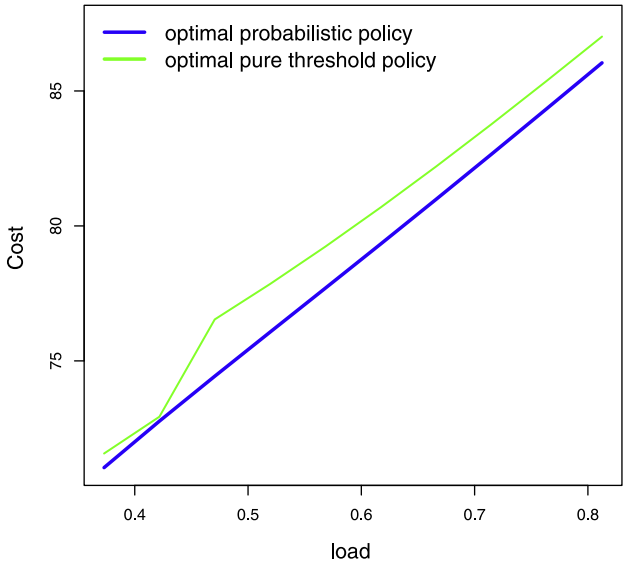


Fig. 13. Cost of doing backup for varying load.

- a policy with probabilistic restarting conditions is able to address the trade off between performing backups too frequently and reducing resource consumption.
- for a give system load ρ, the probabilistic restarting policy with optimal parameters α_i^{ρ} s may outperform the fixed threshold policies.
- the QoS measures can be precisely computed numerically and the parameters can be chosen to meet the desired Quality of Service.
- the performance changes significantly if the model parameters are varied. Therefore, users with different desired QoS need to be treated differently.

6. Conclusions and future work

We have been able to model exhaustive data backup processes as a very general queueing process with batch service, vacations and probabilistic restarting conditions. Moreover, the number of arrivals in a slot and service time can have any general distribution. We have shown it is natural to model data backup processes this way. By computing the QoS measures and analyzing their dependence on the system parameters, we have illustrated how one can strike a balance between doing frequent backups and reducing resource usage for a given Quality of Service. Our observations in Section 5.1 summarize the dependence of performance measures on model parameters. Selecting the optimal parameters allows the system to take longer vacations or support a higher load for the same quality of service. For a given system, such a model empowers us to select the parameters based on the arrival process and desired system performance while keeping the system stable. We have illustrated this using an example of how to use the cost function of the user to select the starting probability parameters.

Our model adopts restrictions that the number of arrivals in the system are assumed to be independent and identically distributed for each slot of the system. Moreover, a data backup process involves replication or erasure coding of stored data which has not been modeled here. The mechanism of replication or coding of the backup process also depends on the topology of the data nodes involved and the exact schedule of the background processes and is therefore challenging to model. We will tackle these challenges in our future work.

Acknowledgment

We would like to thank Research Foundation Flanders (FWO-Vlaanderen), Belgium for their support for this work.

References

[1] T.H. Nguyen, M. Forshaw, N. Thomas, Operating policies for energy efficient dynamic server allocation, Electron. Notes Theor. Comput. Sci. 318 (C) (2015) 159–177.
[2] X. Guan, B.Y. Choi, S. Song, Energy efficient virtual network embedding for green data centers using data center topology and future migration, Comput. Commun. 69 (2015) 50–59.
[3] K. Chen, W.M. Lang, K. Zheng, W.J. Ouyang, Research on the cloud storage security in big data era, Proceedings of the 2015 International Conference on Applied Science and Engineering Innovation, 12 (2015), pp. 659–664.
[4] Amazon, Amazon web service s3, 2017, (https://aws.amazon.com/s3/).
[5] V. Chang, G. Wills, A model to compare cloud and non-cloud storage of big data, Future Gener. Comput. Syst. Int. J. Esci. 57 (2016) 56–76.
[6] F. Yu, Y. Wan, R. Tsaih, Quantitative quality estimation of cloud-based streaming services, Comput. Commun. 125 (2018) 24–37.
[7] P.M. Van de Ven, B. Zhang, A. Schörgendorfer, Distributed backup scheduling: modeling and optimization, 2014 Proceedings of the IEEE INFOCOM, (2014), pp. 1644–1652.
[8] R. Xia, F. Machida, K.S. Trivedi, A Markov decision process approach for optimal data backup scheduling, Proceedings of the Forty-Fourth Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2014, Atlanta, GA, USA, (2014), pp. 660–665.
[9] B.T. Doshi, Queueing systems with vacations — a survey, Queueing Syst. 1 (1) (1986) 29–66.
[10] T. Naishuo, Z.Z. George, Vacation Queueing Models, Springer US, 2006.
[11] K. Sikdar, U.C. Gupta, On the batch arrival batch service queue with finite buffer under server's vacation: $M^X/G^Y/1/N$ queue, Comput. Math. Appl. 56 (11) (2008) 2861–2873.
[12] R. Arumuganathan, S. Jeyakumar, Steady state analysis of a bulk queue with multiple vacations, setup times with n-policy and closedown times, Appl. Math. Model. 29 (10) (2005) 972–986.
[13] K. Sikdar, U.C. Gupta, Analytic and numerical aspects of batch service queues with single vacation, Comput. Oper. Res. 32 (4) (2005) 943–966.
[14] H.W. Lee, S.S. Lee, K.C. Chae, R. Nadarajan, On a batch service queue with single vacation, Appl. Math. Model. 16 (1) (1992) 36–42.
[15] H. Bruneel, B.G. Kim, Discrete-Time models for communication systems including ATM, Kluwer Academic Publishers, Norwell, MA, USA, 1992.
[16] D. Chen, K.S. Trivedi, Optimization for condition-based maintenance with semi-Markov decision process, Reliab. Eng. Syst. Saf. 90 (1) (2005) 25–29.
[17] Microsoft, Microsoft azure storage, 2017, (https://azure.microsoft.com/).
[18] D. Claeys, J.-P. Dorsman, A. Saxena, J. Walraevens, H. Bruneel, A queueing-theoretic analysis of the threshold-based exhaustive data-backup scheduling policy, Proceedings of the International Conference on Numerical Analysis and Applied Mathematics, (2016).
[19] host-it, 2018, (http://www.host-it.ie/online-backup/).
[20] T. Fry, Data processing, Elsevier Science, 2013. URL https://books.google.be/books?id=4fz8BAAAQBAJ .
[21] A. Zomaya, S. Sakr, Handbook of Big Data Technologies, Springer International Publishing, 2017. URL https://books.google.be/books?id=SsQ2DgAAQBAJ .
[22] D. Boullery, A. Schörgendorfer, P. Van de Ven, B. Zhang, Balanced distributed backup scheduling, 2016. US Patent 9,244,777. URL https://www.google.com/patents/US9244777.
[23] C. Gkantsidis, P.R. Rodriguez, Network coding for large scale content distribution, Proceedings of the IEEE Twenty-Fourth Annual Joint Conference of the IEEE Computer and Communications Societies. 4 (2005), pp. 2235–2245 vol.4.
[24] Druva, 2018, (https://www.druva.com/blog/understanding-rpo-and-rto/).
[25] W.B. Powell, P. Humblet, The bulk service queue with a general control strategy: theoretical analysis and a new computational procedure, Oper. Res. 34 (2) (1986) 267–275.
[26] I.J.B.F. Adan, J.S.H. van Leeuwaarden, E.M.M. Winands, On the application of Rouche's theorem in queueing theory, Oper. Res. Lett. 34 (3) (2006) 355–360.
[27] A.S. Alfa, Applied Discrete-Time Queues, second ed., Springer Publishing Company, Incorporated, 2015.
[28] N.L. Johnson, A proof of Wald's theorem on cumulative sums, Ann. Math. Stat. 30 (4) (1959) 1245–1247. URL http://www.jstor.org/stable/2237468 .
[29] J. Liebeherr, A. Burchard, F. Ciucu, Delay bounds in communication networks with heavy-tailed and self-similar traffic, IEEE Trans. Inf. Theory 58 (2) (2012) 1010–1024.