Performance Analysis of Cloud DVR for Energy Efficiency Clustering Strategy

Zhen Zhao(✉)

Comcast Interative Media, Comcast, 1701 JFK Blvd, Comcast Center, Philadelphia, PA 19103, USA
Zhen_Zhao@Comcast.com

Abstract. Cloud digital video recorder (cDVR) is a new service that Comcast provides to its subscribers. The primary current legal interpretation approving cloud DVR relies on a single copy in the Cablevision decision. This makes the cDVR data center running cost very high. An asynchronous service system with categorizing users by cDVR usage is employed to reduce the energy consumption. In this system, cDVRs with similar usage schedule are constructed in one cluster. The cloud recording service on this cluster goes to sleep if there has been a period with no cDVR requests. When there are one or more cDVR request arrivals, those requests are buffered in queues while the cDVR service wakes up. In this paper, a 2-class Markov Geo/G/1/K vacation model is presented to analyze the performance of this system. Different scheduling policies are compared in the simulations and experiments.

Keywords: Cloud tv · Cloud dvr · Markov process

1 Introduction

Nowadays, cloud computing is becoming popular. Comcast is providing cloud TV service to its subscribers, which gives the customers two major flexibilities: (1) watching TV on any device anytime anywhere; (2) recording the TV/Movie programs in cloud digital video record (cDVR) so they could access to cDVR to play the videos from any device later. It is straightforward to store the videos in content delivery network, hence the cDVR only need store the urls of recorded videos. However, due to the requirement of US laws, each cDVR has to provide a physical copy for any program that users record. This makes the cDVR service have much heavier load. Besides the cost of hardware, the daily energy consumption also increases a lot. To reduce the energy consumption, we deploy an asynchronous system of cDVR recording service. Clustered by the usage histogram. In this system, users with similar cDVR usage habits are categorized within one group. CDVRs of this group are put in one server cluster. If there have been a while without any requests, the cDVR recording service of the cluster goes to sleep. If some new requests arrive, the cDVR recording service is waked up. During the wake-up period, the recording requests are buffered in the two queues. Those of recording QAM videos are put into the QAM queue and those of recording IP Streams are put into IP queue.

Our investigation employs vacation modeling results from queueing theory, to obtain blocking probabilities, and queue lengths incurred by sleeping policies for the cDVR recording service responding to the QAM and IP streaming recording requests with a finite buffer. To the best of my knowledge, this is the first work on a 2-class M/G/1/K vacation model, where two classes of requests, each with its own buffer, are processed by a server following a sleeping policy. The sleeping policy is usually characterized by three aspects: (*i*) *How does the sleeping process start?* The exhaustive policy is widely used, in which the cDVR service won't go to sleep until the buffer is empty. (*ii*) *How does the sleeping process end?* Two approaches are the most popular: termination policy and threshold policy. In the former policy, the server checks its buffer occupancy only at the time instant of sleep termination. If the buffer is empty, it goes back to sleep again. Otherwise, it starts processing requests. The latter policy requires the server to check its buffer state whenever the buffer occupancy changes. If the occupancy exceeds the threshold, the server starts processing requests. (*iii*) *What is the distribution of the sleeping process?* Usually, the sleeping process is assumed as a general distribution with an independent and identically distributed (i.i.d.) random variable (r.v.). Our work focuses on the exhaustive, termination policy, and a process with i.i.d. r.v.

The paper is organized as follows. In Sect. 2, a 2-class vacation model is described. In Sect. 3 the marginal occupancy distributions at an arbitrary time instant are derived. Section 4 presents numerical, simulation and experimental results. Conclusions are discussed in Sect. 5.

2 2-Class Vacation Model

In this section, we present a model of a server receiving and processing 2-class heterogeneous requests (see Fig. 1). Our analytical approach is to embed a Markov chain at the time slot immediately after processing completion slots. Next, we build equilibrium equations for the embedded Markov chain to obtain the marginal occupancy distributions at processing completion slots.

We consider a server with two classes of requests (class $1, 2$, e.g., QAM and IP streaming), each with its own finite buffer. The request arrival of each class is a i.i.d. geometric process. The requests of a certain class are queued into the corresponding buffer if the buffer is not full and blocked otherwise. With a general scheduling function, requests from the two buffers are selected to be processed. The service order for requests of each class is scheduled by the selection function $\alpha(i, j)$, the probability that a class 1 request is chosen to be processed, while i and j requests are in class 1 and 2's buffer/queue, respectively. The processing process has a general distribution. The recording service keeps processing requests until both buffers are empty at a process completion time instant, in which case

Table 1. 2-class model notations

p_m	the probability of a request arrives in a slot of class m, where $m = 1, 2$.
μ	requests processing rate; $1/\mu$ is the mean of the general processing time distribution.
θ	service wakeup rate; $1/\theta$ is the mean of the general sleeping time distribution.
K_m	buffer size of class m, where $m = 1, 2$.
$\alpha(i, j)$	probability that a class 1 request is selected to be processed while i and j requests are in class 1 and 2 queues, respectively.
$\pi_{i,j}$	probability that i class 1 and j class 2 requests are in buffers just after processing completion slots.
$\varpi_{i,j}$	probability that i class 1 and j class 2 requests are in buffers just after sleeping termination slots.
$\pi^\star_{i,j}$	probability that i class 1 and j class 2 requests are in buffers at an arbitrary time slot.
X	the processing time random variable measured in slots.
$a(k)$	the probability distribution of X, $a(k) = \mathbb{P}(X = k)$.
Y	the sleeping time random variable measured in slots.
$v(k)$	the probability distribution of Y, $v(k) = \mathbb{P}(Y = k)$.
$\hat{B}$	remaining processing time slots for the request in service.
$\tilde{B}$	elapsed processing time slots for the request in service.
$\hat{V}$	remaining sleeping time for the node in sleep.
$\tilde{V}$	elapsed sleeping time for the node in sleep.
$G(z)$	probability generating function (PGF) of $G(k)$; $G(z) \triangleq \sum_{k=0}^{\infty} G(k)z^k$, where $0 < z \leq 1$
$E(G)$	expectation of G.

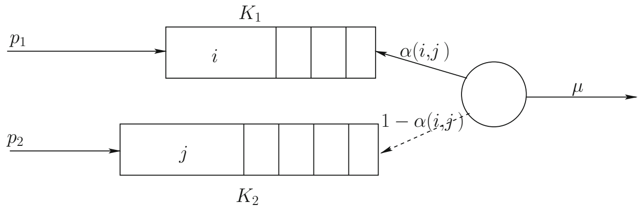

Fig. 1. Queueing model of a recording service processing 2-class requests

the recording service goes to sleep and will continue to sleep if at the sleeping termination time instant there are no requests buffered waiting for processing (Table 1).

As mentioned before, our analytical approach is to embed a Markov chain at the time instant just after processing completion slots, as is typically done for the ordinary Geo/G/1/K system. We define a couple of random variables $(N_1(k^+), N_2(k^+))$ to be the number of requests of each class in their respective buffers immediately following processing completions. Since the arrival process is Markovian, it is evident that the random process $\{N_1(k^+), N_2(k^+), k^+ = 0^+, 1^+, \ldots\}$ is a discrete Markov renewal process [18]. The vector of stationary probability masses for the embedded Markov chain is denoted by $\pi_{i,j}$, and its (i,j)th element is given by

$$\pi_{i,j} = \lim_{k\to\infty} \mathbb{P}\{N_1(k^+) = i, N_2(k^+) = j\}.$$

We denote the probability that i requests of class m $(m = 1, 2)$ arrive during a processing time $X = k$ and during a sleeping time $Y = k$ by $a_m(i)$ and $v_m(i)$, respectively,

$$a_m(i) = \sum_{k=i}^{\infty} \binom{k}{i} p_m^k (1-p_m)^{k-i} b(k),$$

$$v_m(i) = \sum_{k=i}^{\infty} \binom{k}{i} p_m^k (1-p_m)^{k-i} v(k).$$

and the probabilities that no less than i requests of class m $\underline{(m = 1, 2)}$ arrive during a processing time X and during a sleeping time Y by $\overline{a_m(i)}$ and $\overline{v_m(i)}$,

$$\overline{a_m(i)} = \sum_{j=i}^{\infty} a_m(j), \qquad \overline{v_m(i)} = \sum_{j=i}^{\infty} v_m(j).$$

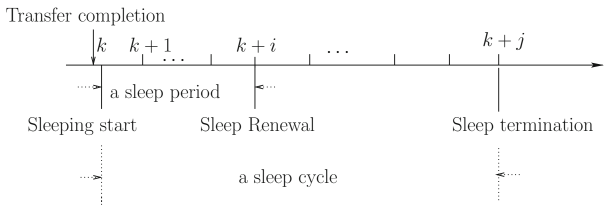

Fig. 2. Sleep cycle starting at processing completion time slots

When a sleep period completes, the server checks the buffer occupancies of the two classes. If they are empty, another sleep period with independent Y will be started. This process repeats unless there are requests waiting in at least one of the two buffers at a sleep completion time instant. We define a sleep cycle as the time interval from the time instant the server or cluster goes to sleep to the time instant it starts to processing requests. Thus, a sleep cycle is composed of one or more sleep periods (see Fig. 2). We denote the probability that i $(i < K_m)$ and K_m requests of class m arrive during a sleep cycle by $\varphi_m(i)$ and $\varphi_m(K_m)$. It is easy to see that they are geometric distributions.

$$\varphi_m(i) = \sum_{j=0}^{\infty} v_m(0)^j v_m(i) = \frac{v_m(i)}{1 - v_m(0)}, \tag{1}$$

$$\varphi_m(i = K_1) = \sum_{j=0}^{\infty} v_m(0)^j \overline{v_m(K_m)} = \frac{\overline{v_m(K_m)}}{1 - v_m(0)}. \tag{2}$$

A generic term to denote the probability transition from (i, j) to (k, l) is $P_{(i,j)(k,l)}$. To find $P_{(i,j)(k,l)}$, we consider four cases: (i) $0 < k < K_1, 0 < l < K_2$; (ii) $0 < k < K_1, l = K_2$; (iii) $k = K_1, 0 < l < K_2$; and (iv) $k = K_1, l = K_2$.

Case (i) $0 < k < K_1, 0 < l < K_2$. All requests are queued in their respective buffers, excluding the processed request. The exact number of class 1 and 2 requests is $k - i$ and $l - j$, respectively.

If $i > 0, j = 0$, there is no sleep period. Only the head of line request of class 1 is processed. The case $i = 0, j > 0$ is similar. Hence, we obtain

$$P_{(i,0)(k,l)} = a_1(k - i + 1)a_2(l),$$
$$P_{(0,j)(k,l)} = a_1(k)a_2(l - j + 1).$$

If $i > 0, j > 0$, the request being proccessed is selected from class 1 with probability $\alpha(i, j)$ and from class 2 with probability $1-\alpha(i, j)$. Thus, we have the transition probabilities $\alpha(i, j)a_1(k-i+1)a_2(l)$ and $[1-\alpha(i, j)]a_1(k)a_2(l-j+1)$.

If $i = 0, j = 0$, there is a sleep cycle between the successive processing completions. When only a single class of requests is arrive during the sleep cycle, the transition probability $P_{(0,0)(k,l)}\{\text{Single}\}$ is given as

$$P_{(0,0)(k,l)}\{\text{Single}\} = \varphi_1(0)a_1(k)\Big(\sum_{j=1}^{l+1} \varphi_2(j)a_2(l - j + 1)\Big) + \varphi_2(0)a_2(l)\Big(\sum_{i=1}^{k+1} \varphi_1(i)a_1(k - i + 1)\Big).$$

When both classes of requests arrive during the sleep cycle, the selection function chooses a request randomly (following the general distribution $\alpha(i, j)$) from the two buffers. So the transition probability $P_{(0,0)(k,l)}\{\text{Two}\}$ is expressed as

$$\begin{aligned} &P_{(0,0)(k,l)}\{\text{Two}\} \\ &= \sum_{i=1}^{k+1}\sum_{j=1}^{l} \alpha(i, j)\varphi_1(i)a_1(k - i + 1)\varphi_2(j)a_2(l - j) \\ &+ \sum_{i=1}^{k}\sum_{j=1}^{l+1}(1 - \alpha(i, j))\varphi_1(i)a_1(k - i)\varphi_2(j)a_2(l - j + 1). \end{aligned}$$

Thus, the transition probability is

$$P_{(0,0)(k,l)} = P_{(0,0)(k,l)}\{\text{Single}\} + P_{(0,0)(k,l)}\{\text{Two}\}.$$

Hence, we have the equilibrium equation for Case i as

$$\pi_{k,l} = \pi_{0,0}P_{(0,0)(k,l)} + \sum_{i=1}^{k+1} \pi_{i,0}P_{(i,0)(k,l)} + \sum_{j=1}^{l+1} \pi_{0,j}P_{(0,j)(k,l)}$$
$$+\sum_{i=1}^{k+1}\sum_{j=1}^{l} \alpha(i,j)\pi_{i,j}a_1(k-i+1)a_2(l-j)$$
$$+\sum_{i=1}^{k}\sum_{j=1}^{l+1}(1-\alpha(i,j))\pi_{i,j}a_1(k-i)a_2(l-j+1). \tag{3}$$

Case (*ii*) $0 < k < K_1, l = K_2$. The analysis is the same as Case (i) for class 1. However, it is different for class 2 when $l = K_2$. It is possible that more than l requests arrive but are blocked due to the finite buffer size K_2. Thus, the probability of $K_2 - j$ or more requests of class 2 arrive is $\overline{a_2(K_2-j+1)}$ or $\overline{a_2(K_2-j)}$. So the transition probability from $(0,0)$ to (k, K_2) is

$$P_{(0,0)(k,K_2)} = \varphi_1(0)a_1(k)\Big(\sum_{j=1}^{K_2} \varphi_2(j)\overline{a_2(K_2-j+1)}\Big)$$
$$+\varphi_2(0)\overline{a_2(K_2)}\Big(\sum_{i=1}^{k+1} \varphi_1(i)a_1(k-i+1)\Big)$$
$$+\sum_{i=1}^{k+1}\sum_{j=1}^{K_2} \alpha(i,j)\varphi_1(i)a_1(k-i+1)\varphi_2(j)\overline{a_2(K_2-j)}$$
$$+\sum_{i=1}^{k}\sum_{j=1}^{K_2}(1-\alpha(i,j))\varphi_i^1 a_1(k-i)\varphi_2(j)\overline{a_2(K_2-j+1)}.$$

And the equilibrium equation is given by

$$\pi_{k,K_2} = \pi_{0,0}P_{(0,0)(k,K_2)} + \sum_{i=1}^{k+1} \pi_{i,0}a_1(k-i+1)\overline{a_2(K_2)}$$
$$+\sum_{j=1}^{K_2} \pi_{0,j}a_1(k)\overline{a_2(K_2-j+1)}$$
$$+\sum_{i=1}^{k+1}\sum_{j=1}^{K_2} \alpha(i,j)\pi_{i,j}a_1(k-i+1)\overline{a_2(K_2-j)}$$
$$+\sum_{i=1}^{k}\sum_{j=1}^{K_2}(1-\alpha(i,j))\pi_{i,j}a_1(k-i)\overline{a_2(K_2-j+1)}. \tag{4}$$

Case *(iii)* $\boldsymbol{k = K_1, 0 < l < K_2}$. It is readily seen that this scenario is almost the same as Case *i*. The equilibrium equations are obtained by exchanging $a_1(\cdot)$ and $a_2(\cdot)$ in (3), where $\cdot$ is a wild card. For brevity, we do not repeat it here.

Case *(iv)* $\boldsymbol{k = K_1, l = K_2}$. Again, if we replace $a_m(\cdot)$ with $\overline{a_m(\cdot)}$ $(m = 1, 2)$ in (3), we get the equilibrium for π_{K_1,K_2}.

With all the above equilibrium equations and the bound condition $\sum_{i=0}^{K_1}\sum_{j=0}^{K_2}\pi_{i,j} = 1$, all the stationary probabilities can now be numerically computed.

3 Marginal Occupancy Distributions

So far, we have derived a computational procedure for obtaining the equilibrium probabilities for a Markov process embedded at the time slots of processing completion. Similarly to the analysis of a single class Geo/G/1/K, we now turn to the marginal occupancy distributions $\pi^\star_{i,*}, \pi^\star_{*,j}$ as seen at an arbitrary time slot. To focus on the quantity of interest: *buffer occupancies* L_1, L_2 of both classes, we present the results directly.

The occupancies of classes 1 and 2 are given by:

$$L_1 = \sum_{i=0}^{K_1} i\pi^\star_{i,*} = \sum_{i=0}^{K_1} i(\eta^\star_i(1) + \omega^\star_i(1)), \tag{5}$$

$$L_2 = \sum_{i=0}^{K_2} j\pi^\star_{*,j} = \sum_{j=0}^{K_2} j(\chi^\star_j(1) + \omega^\star_j(1)). \tag{6}$$

In (5) and (6), $\eta^\star_n(Z), \chi^\star_n(Z), \omega^\star_n(Z)$ are the PGF functions. In the interest of space, we only show parts of class 1's results here. Class 2 is similar. Derivation details are skipped here due to the space limitation.

When $0 \le n < K_1$, $\eta^\star_n(Z), \omega^\star_n(Z)$ satisfy:

$$\begin{aligned}\eta^\star_n(Z) = \rho' \cdot \Big[&\sum_{j=0}^{K_2}\sum_{i=1}^{n+1}\alpha(i,j)(\pi_{i,j} + \varpi_{i,j})A^\star_{n-i+1}(Z)\\ &+ \sum_{j=1}^{K_2}\sum_{i=0}^{n}(1-\alpha(i,j))(\pi_{i,j} + \varpi_{i,j})A^\star_{n-i}(Z)\Big],\\ \omega^\star_n(Z) = (1&-\rho')I^\star_n(Z),\end{aligned}$$

where $A^\star_n(Z), I^\star_n(Z)$ are obtained by

$$A^\star_n(Z) = \frac{1}{\mu}\frac{\lambda_1^n}{(\lambda_1 + \ln Z)^{n+1}}\Big[B^\star(Z) - \sum_{i=0}^{n} a_1(i)\Big(\frac{\lambda_1 + \ln Z}{\lambda_1}\Big)^i\Big],$$

$$I^\star_n(Z) = \frac{1}{\mu}\frac{\lambda_1^n}{(\lambda_1 + \ln Z)^{n+1}}\Big[V^\star(Z) - \sum_{i=0}^{n} v_1(i)\Big(\frac{\lambda_1 + \ln Z}{\lambda_1}\Big)^i\Big].$$

The above expressions allow us to compute the buffer occupancy for each of the two classes of requests arrive at the server under a sleeping policy. Next, we present numerical, simulation and experimental results for this metric under three situations: (i) different wakeup rates; (ii) different buffer sizes; (iii) different sharing percentage of a common buffer with fixed size.

4 Numerical, Simulation and Experimental Results

In this section, we verify our modeling analysis comparing numerical with simulation and experimental results. Moreover, by varying the wakeup rate and buffer sizes of the two classes, we have an insight on the sleeping costs of servers processing heterogeneous requests. First, we describe the configurations of our simulations and experiments. Results and discussions on buffer occupancies are given later.

In our modeling, the processing process and sleeping process have general distributions. To show that, we select three distributions for processing and sleeping process: exponential, uniform, and deterministic. Unless mentioned, the simulations and experiments have the same setup. For the selection function $\alpha(i, j)$, we chose four scheduling policies: (i) LJF: longest job first. If $j \geq i$, we select class 2. The job processing length here is related to if there is in-home check, parental control tv rating check, etc. (ii) SJF: shortest job first. If $j \leq i$, we select class 2. (iii) HOL: class 2 has priority over class 1; i.e., we always processing class 2's requests first unless its buffer is empty. (iv) BER: Bernoulli model. Here, we set the probability to select either class request as 0.5.

Figure 3 shows that the average occupancy of classes 1 and 2 decreases with the increase of the wakeup rate θ and tends to be constants after θ is greater than 1. For both classes, the occupancy of LJF is greater than the one of SJF. This

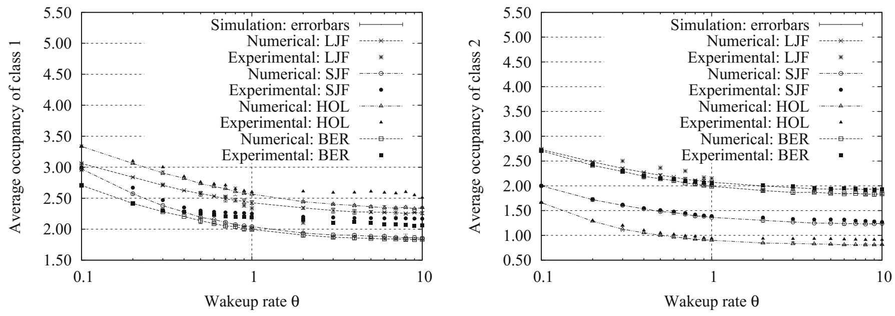

Fig. 3. Average occupancy of classes 1 and 2 varying the wakeup rate θ from 0.1 to 10. The time interval between successive wakeup time epochs is exponential. Both classes have the same Poisson arrival rate $\lambda_1 = \lambda_2 = 0.5$. The processing rate is $\mu = 1$. Each class has a finite buffer with size of 5. The same four selection policies are considered: (a) LJF; (b) SJF; (c) HOL; and (d) BER with probability 0.5 of selecting class 1.

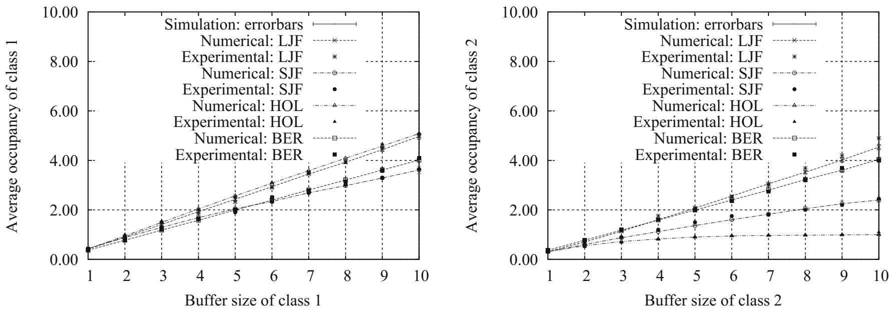

Fig. 4. Average occupancy of classes 1 (top) and 2 (bottom) with buffer size as a parameter varying from 1 to 10. Both classes have the same Poisson arrival rate $\lambda_1 = \lambda_2 = 0.5$. The processing distribution mean is $\mu = 1$. The time interval between successive wakeup time epochs is exponential with mean $1/\mu = 1$. The same four selection policies are considered: (a) LJF; (b) SJF; (c) HOL; and (d) BER with probability 0.5 of selecting class 1.

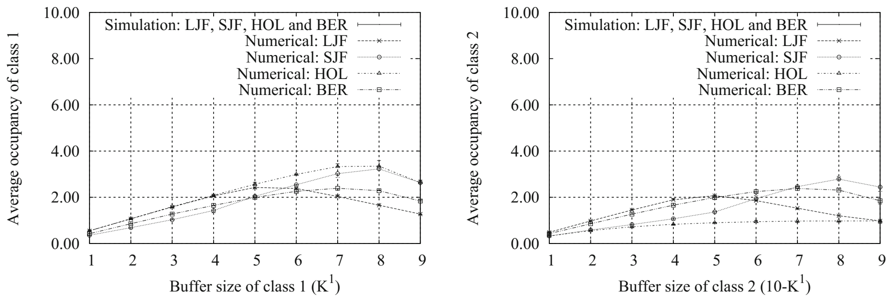

Fig. 5. Average occupancy of classes 1 (top) and 2 (bottom) with a fixed total buffer size of 10. The buffer size K_1 of class 1 is a parameter varying from 1 to 9, while $K_2 = 10 - K_1$. Both classes have the same Poisson arrival rate $\lambda_1 = \lambda_2 = 0.5$. The processing distribution mean is $\mu = 1$. The time interval between successive wakeup time epochs is exponential with mean $1/\mu = 1$. The average occupancy of each of the two classes is shown for the same four selection policies: (a) LJF; (b) SJF; (c) HOL; and (d) BER with probability 0.5 of selecting class 1.

behavior is a result of sacrificing some cost in terms of delaying requests processing. For HOL, since class 2 has priority over class 1, the average occupancy of class 2 is lower than the one of class 1 due to the increased delay of class 1.

In Fig. 4, it is evident that LJF has greater average occupancy than SJF for both classes no matter what their buffer sizes are. Similar conclusions can be drawn for HOL. Again, due to the same selection probability, BER has similar performance for both classes.

From Fig. 5, we see that while the buffer size K_1 of class 1 is less than 5 (i.e., half of the fixed total buffer size), the occupancy of classes 1 and 2 under LJF increase as class 1's buffer size increases. After that, the reverse occurs:

occupancy of classes 1 and 2 decrease with increasing K_1. Changing the sharing percentage of the common buffer only changes the saddle point, but not the curve tendency.

5 Conclusion

In this paper, we apply the $Geo/G/1$ and Geo/G/1/K vacation models to the cDRV server processing QAM and IP stream recording requests to study the cost of adopting sleeping policies. The performance costs, namely, queue length and delay were obtained by theoretical derivation and are convergent in most cases. We found that the processing time distribution does not affect the cost. We also found cases where the requests arrival process and scheduling function do affect the costs.

References

1. Wang, L., Xiao, Y.: Energy saving mechanisms in recording service networks. In: Proceedings of the 2nd International Conference on Broadband Networks (BROADNETS), October 2005
2. Wang, L., Xiao, Y.: A survey of energy-efficient scheduling mechanisms in recording service networks. Mob. Netw. Appl. **11**, 723–740 (2006)
3. Jurdak, R., Ruzzelli, A.G., O'Hare, G.: Adaptive radio modes in recording service networks: How deep to sleep? In: 5th Annual IEEE Communications Society Conference on Recording Service, Mesh and Ad Hoc Communications and Networks, SECON 2008, pp. 386–394, June 2008
4. White, H., Christie, L.S.: Queuing with preemptive priorities or with breakdown. Oper. Res. **6**(1), 79–95 (1958)
5. Avi-Itzhak, B., Naor, M.: Some queueing problems with the service station subject to server breakdown. Oper. Res. **10**, 303–320 (1963)
6. Yadin, M., Naor, P.: Queueing systems with a removable server. Oper. Res. **14**, 393–405 (1963)
7. Levy, Y., Yechiali, U.: Utilization of idle time in an M/G/1 queueing system. Manage. Sci. **22**, 202–221 (1975)
8. Fuhrmann, S., Cooper, R.: Stochastic decompositions in the M/G/1 queue with generalized vacations. Oper. Res. **33**, 1117–1129 (1985)
9. Teghem, J.: Control of the service process in queueing system. Eur. J. Oper. Res. **23**, 141–158 (1986)
10. Doshi, B.: Queueing systems with vacations - a survery. Queueing Syst. **1**, 22–66 (1986)
11. Kella, O.: Optimal control of the vacation scheme in an M/G/1 queue. Oper. Res. **38**, 724–728 (1990)
12. Li, Q., Tian, N., Cao, J.: Conditional stochastic decomposition in the M/M/c queue with server vacation. Stoch. Models **14**, 367–377 (1999)
13. Lee, T.: M/G/1/N queue with vacation time and exhaustive service discipline. Oper. Res. **32**, 774–784 (1984)
14. Frey, A., Takahashi, Y.: A note on an M/GI/1/N queue with vacation time and exhaustive service discipline. Oper. Res. Lett. **21**, 95–100 (1997)

15. Alfa, A.S.: A discrete MAP/PH/1 queue with vacations and exhaustive service. Oper. Res. Lett. **18**, 31–40 (1995)
16. Alfa, A.S.: A discrete MAP/PH/1 vacation queue with gate time-limited service. Queueing Syst. **29**, 35–54 (1998)
17. Alfa, A.S.: Vacation models in discrete time. Queueing Syst. **44**(1), 5–30 (2003)
18. Çinlar, E.: Introduction to Stochastic Processes. Prentice-Hall, Englewood Cliffs (1975)
19. Cooper, R.: Introduction to Queueing Theory, 2nd edn. North-Holland, New York (1981)