Chapter 7

Bulk Services in Cloud Computing Centers

7.1 Introduction

Efficient resource utilization is a critical issue in case of bulk data transfer in cloud computing. With the advent of cloud computing most of the data intensive applications are developed in the cloud system. As applications need a reservation for network resources, the cloud center has to provide service in batches to the applications for the efficient use of resources [34]. For the data intensive applications the data and storages are geographically dispersed and therefore, the performance of the cloud centers and user requests may degrade with the performance of the networks. Reservation and aggregated resource utilization are some of the offsprings to accomplish guarantee of service during bulk data transfer in cloud to meet user's QoS requirement. Large number of applications deployed in the cloud system may need bulk data transfer for example; scientific computing, IPTV and real time multimedia streaming, etc [84–86]. In such systems a huge amounts of data need to be transferred to the geographically distributed users. It is necessary to satisfy the users' QoS requirements by cloud resource reservation model (CRRM) to perform the desired level of service. QoS includes availability,

throughput, reliability, security, etc. and also performance measures such as task blocking probability, mean number of task in the queue, number of busy virtual machines, waiting time, probability of immediate service [87, 88], all of which can be determined using queueing theory tools [89, 90].

In this chapter, we propose a new cloud resource reservation model with a bulk transfer and bulk service mechanism that exploits the capacities of network links to deliver bulk content efficiently. It fills the gap between the performance of the network and data resource to provide service quality guarantee for data accessing [91]. The bulk data transfer technique in cloud computing optimizes the combined resource utilization and reservation. It uses bandwidth aware routing, which adapts to dynamically change available bandwidth across potentially multiple paths between the source and the destination of a bulk transfer. Our focus is on bulk data transfers, for example file backups, software downloads, or data transferred for data mining. The individual packets can be delayed and sent only when spare bandwidth is available within a bulk data transfer. The Internet service providers can use this unused link capacity to deliver greater volumes of bulk data without having to upgrade their backbone links, thus minimizing the cost of data transfers.

7.2 System Model

The proposed cloud resource reservation model provides uniform mechanisms for making cloud resources reservation via broker to guarantee performance [92]. The architecture of the service oriented resource broker for resource reservation is shown in Figure 7.1. The model works as follows:

- Ahead of the user requests to the broker, the dynamic virtualized resource information index service (IIS) automatically register the network and data resource information such as bandwidth, delay.

- Once users submit the requests, the requirements from application layer

pass to broker. The Resource requirement interpreter converts the user requirement to quality of service (QoS) metrics, such as deadline of transmission time, file name in data resource, data file size and cost, etc.

- The resource discovering unit collects the information from resource necessity interpreter to discover possible combination of resources based on the attributes in terms of performance and availability. The dynamic resource index service provides the resource information to the resource discovering unit . This unit establishes potential collection of resources by mapping the resource requirement to the virtualized resources. Afterwards the broker accumulates an aggregated resource information for client request from the resource necessity interpreter. The list comprising all potential aggregated resources passes to resource compounding ranker to find the most beneficial match of resource.

- The priorities of resource combines are compared by the resource compounding ranker. It orders the resource compounding based on dynamic information, and discovers the resource compounding that is likely to render most beneficial service based on the client QoS constraint. It is looking for the suitable data file from end-system resource and takes the clients timing request to calculate the minimum bandwidth for each request. Then it determines all potential aggregated resource between the client and end-system resource. The routes of network resources are decided by routing protocols and covers the path through the server.

- Finally, resource booking unit makes coordinated resource booking through the booking interface. Clients get successful signal when all compounding resources are successfully booked. Otherwise, client receives a failure signal through the booking interface.

In cloud computing, transmission of user requests are of various types (eg; data, voice, video, etc). There are many applications in the web which are

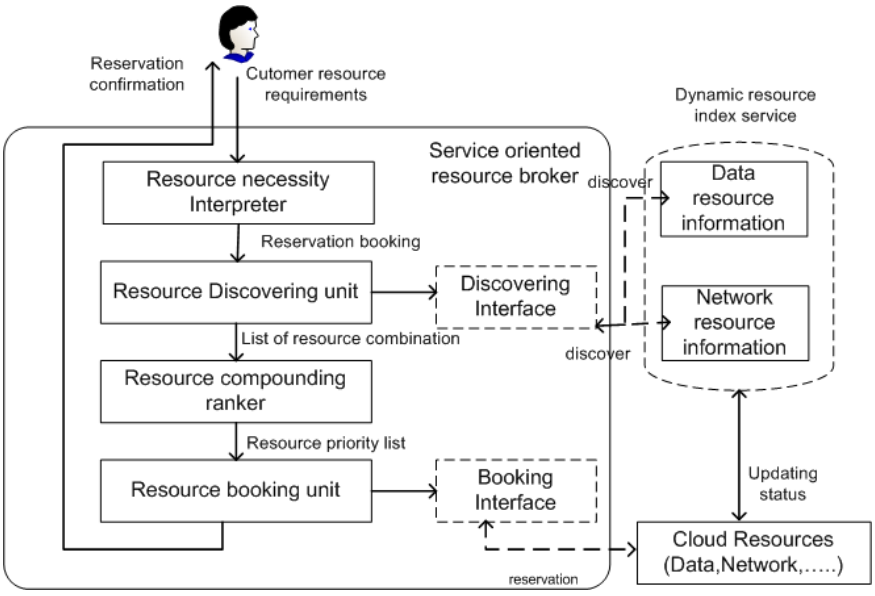

Figure 7.1: The Service Oriented Resource Broker Architecture for Resource Reservation.

accessed by a large number of clients at the same instance. When a client sends a request to a web application the broker forwards the request to the queue of the target web application which needs to be processed in bulk. The instances of the target web application running into VMs act as service centers to process the requests in the queue. We figure out the process of the client requests in a single web application that has the following features. The inter-arrival times between any two successive client requests are independent of each other and have a common probability distribution. The clients will receive responses if requests are processed by web applications in time, or receive exceptions due to timeout of waiting. The cloud platform can create a single VM or a cluster of VMs to process the client requests. When a bulk of requests are submitted in the queue, the requests are forwarded to the VMs which are currently idle. Since the cache(s) or buffer(s) of the VM(s) of an application is finite, the number of waiting requests is limited. It means if the waiting room of an application is fully occupied extra requests arriving at this service would be lost.

7.3 Modelling and Analysis

We consider a cloud server farm as a finite buffer bulk service queueing system with homogeneous VMs. We assume that the inter-arrival times of the client requests are independent and exponentially distributed with mean arrival rate λ. Each web application on cloud has one or multiple instance running on the VMs, and each instance can serve a bulk of requests. The number of VMs used depends upon the number of client requests present in the system. The client requests are served in batch mode with a minimum limit of one and a maximum limit of b requests per VM. The task service time distribution of a batch is assumed to be exponential with mean service rate μ. The system has c homogeneous VMs which deliver service in order of client request arrivals that is, first-come, first-served (FCFS). The system has finite-buffer space of size N, that is, at any time the system can accommodate at most $N + bc$ requests. The traffic intensity of the system is $\rho = \lambda/bc\mu$.

The web applications are modeled as queues and the VMs are modeled as service centers. Because the cloud controller is the portal of the cloud, all the requests will be sent to broker and forward to queues, when there will be a bulk of requests they are forwarded to an instance of a VM and all responses will be sent back to clients. System model of bulk service on cloud is shown in Figure 7.2. We denote the states of the system by (n, i), where n being the number of tasks in the queue and i is an indicator variable. $i = r$ $(0 \leq r \leq c-1)$ implies that r servers are busy and $i = c$ implies that all servers are busy. Let $P_{n,i}(t) = Pr\{$n customers in the queue excluding the batches in service and i servers are busy at time t$\}$, $0 \leq n \leq N$, $0 \leq i \leq c$.

Relating the state of the system at time t and $t+dt$, and using the Markov process theory, we can use the following differential-difference equations in the steady-state [93]:

$$\begin{aligned} \frac{d}{dt}P_{0,0}(t) &= -\lambda P_{0,0}(t) + \mu P_{0,1}(t) \qquad (7.1) \\ \frac{d}{dt}P_{0,r}(t) &= -(\lambda + r\mu)P_{0,r}(t) + \lambda P_{0,r-1}(t) + (r+1)\mu P_{0,r+1}(t), \end{aligned}$$

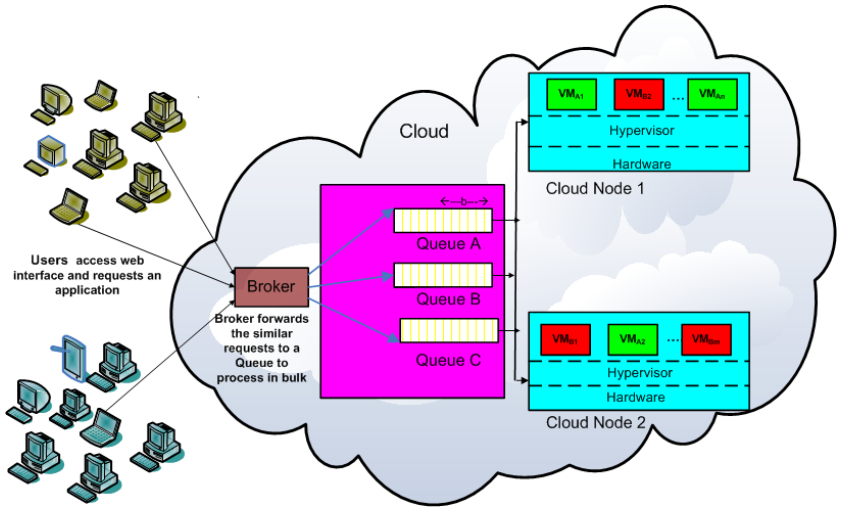

Figure 7.2: System model of bulk service on cloud.

$$1 \leq r \leq c-1, \tag{7.2}$$

$$\frac{d}{dt}P_{0,c}(t) = -(\lambda + c\mu)P_{0,c}(t) + \lambda P_{0,c-1}(t) + c\mu \sum_{i=1}^{b} P_{i,c}(t) \tag{7.3}$$

$$\frac{d}{dt}P_{n,c}(t) = -(\lambda + c\mu)P_{n,c}(t) + \lambda P_{n-1,c}(t) + c\mu P_{n+b,c}(t),$$
$$1 \leq n \leq N-b, \tag{7.4}$$

$$\frac{d}{dt}P_{n,c}(t) = -(\lambda + c\mu)P_{n,c}(t) + \lambda P_{n-1,c}(t),$$
$$N-b+1 \leq n \leq N-1, \tag{7.5}$$

$$\frac{d}{dt}P_{N,c}(t) = -c\mu P_{N,c}(t) + \lambda P_{N-1,c}(t). \tag{7.6}$$

In steady state, let $P_{0,r} = \lim_{t\leftarrow\infty} P_{0,r}(t)$, $P_{n,c} = \lim_{t\leftarrow\infty} P_{n,c}(t)$. The steady state equations of the system become:

$$0 = -\lambda P_{0,0} + \mu P_{0,1} \tag{7.7}$$

$$0 = -(\lambda + r\mu)P_{0,r} + \lambda P_{0,r-1} + (r+1)\mu P_{0,r+1},$$
$$1 \leq r \leq c-1, \tag{7.8}$$

$$0 = -(\lambda + c\mu)P_{0,c} + \lambda P_{0,c-1} + c\mu \sum_{i=1}^{b} P_{i,c} \tag{7.9}$$

$$0 = -(\lambda + c\mu)P_{n,c} + \lambda P_{n-1,c} + c\mu P_{n+b,c},\ 1 \leq n \leq N-b \tag{7.10}$$

$$0 = -(\lambda + c\mu)P_{n,c} + \lambda P_{n-1,c},\ N-b+1 \leq n \leq N-1 \tag{7.11}$$

$$0 = -c\mu P_{N,c} + \lambda P_{N-1,c}. \tag{7.12}$$

Solving recursively, we will get steady-state probabilities $P_{n,c}$ from (7.7) - (7.12) as follows

$$P_{n,c} = \psi_c \left(1+\psi_c\right))^{N-n-1} P_{N,c},\ N-b \le n \le N-1, \tag{7.13}$$

$$\begin{aligned} P_{n,c} &= \psi_c \left(1+\psi_c\right)^{N-b-2-n} \Big[\left\{ \left(1+\psi_c\right)^b - 1 \right\} \left(1+\psi_c\right) \\ &\quad - (N-b-1-n)\psi_c \Big] P_{N,c}, \\ &\quad n = N-b-1, \ldots, 1, 0, \end{aligned} \tag{7.14}$$

$$P_{0,c-1} = \left(1+\psi_c\right) P_{0,c} - \psi_c \sum_{i=1}^{b} P_{i,c} \tag{7.15}$$

$$\begin{aligned} P_{0,r-1} &= \left(1+\psi_r\right) P_{0,r} - \psi_{r+1} P_{0,r+1}, \\ &\quad r = c-1, c-2, \ldots, 1, \end{aligned} \tag{7.16}$$

where $\psi_c = c\mu/\lambda$. Using the normalization condition $\sum_{i=1}^{N} P_{i,c} + \sum_{r=0}^{c} P_{0,r} = 1$, we get the unknown $P_{N,c}$. Multiplying by $P_{N,c}$ in the above equation we get different state probabilities.

7.4 Performance measures

In this section,some performance measures of the proposed bulk transfer data multi VMs cloud computing model has been discussed. Once the steady-state probabilities are evaluated, various performance measures of the model can be evaluated.

Probability of an immediate service

An arriving task gets into service immediately if there is an idle server at the time of its arrival. In such a situation, the response time for the task is equal to the service time. That is, $P_{is} = \sum_{i=0}^{c-1} P_{0,i}$.

Probability of blocking

If a queue is full when a task arrives, it will be discarded, or blocked. So the probability that a task is blocked (P_{loss}) is exactly the same as the probability

that the queue is full. That is, $PBL = P_{N,c}$.

Expected number of busy VMs

The expected number of busy VMs ($E[BS]$) are obtained by

$E[BS] = c - \sum_{j=0}^{c-1} P_{0,c}$.

Mean number of tasks in the queue

The mean number of tasks in the queue (L_q) can be obtained as $L_q = \sum_{i=1}^{N} i.P_{i,c}$.

The mean response time by using Little's rule, is given as $W_q = \frac{L_q}{\lambda'}$, where $\lambda' = \lambda(1 - P_{loss})$ is the effective arrival rate.

7.4.1 Waiting time analysis

Let $W_q^*(s)$ be the L.-S. transform of the distribution function of the waiting time in the queue of a task who is accepted into the system under the FCFS service discipline. An arriving task may find the system in any one of the following states:

Case I: When a new task arrives and finds (n, r); $\quad n = 0, \quad 0 \leq r \leq c-1$, an empty queue and idle VM server, then task is taken for service immediately. Therefore, in this case, task waiting time is zero.

Case II: If new task finds $(nb+k, c)$; $\quad (0 \leq n \leq \nu - 1, 0 \leq k \leq b-1)$, where $\nu = \left\lceil \frac{N}{b} \right\rceil$, tasks in the queue and VMs server are busy, then it joins the queue and waits till the service completion of $(n+1)$ groups of tasks.

Case III: If on arrival new task finds $(nb+k, c); (n = \nu, 0 \leq k \leq N-1-nb)$, then also it has to wait in the queue till the service completion of $(n+1)$ groups of tasks.

$$W_q^*(s) = \frac{1}{1 - P_{loss}} \left[\sum_{r=0}^{c-1} P_{0,r} + \sum_{n=0}^{\nu-1} \left(\frac{c\mu}{c\mu + \theta} \right)^{n+1} \sum_{k=0}^{b-1} P_{nb+k,c} + \left(\frac{c\mu}{c\mu + \theta} \right)^{\nu+1} \sum_{k=0}^{N-1-nb} P_{\nu b+k,c} \right].$$

The mean waiting time in the queue (W_q) can be obtained from above expression and is given by

$$W_q = \frac{1}{c\mu(1-P_{loss})}\left[\sum_{n=0}^{\nu-1}(n+1)\sum_{k=0}^{b-1}P_{nb+k,c} + (\nu+1)\sum_{k=0}^{N-1-\nu b}P_{\nu b+k,c}\right].$$

One may note here that using Little's rule, we can also obtain the average waiting-time in the queue (W_q).

7.5 Numerical Illustrations

In this section, some numerical results in the form of tables and graphs has been presented. Distributions of task in the system is given in Table 7.1. Various performance measures such as the probability of blocking, the average queue length and the average waiting time in the queue are given at the bottom of table.

Figure 7.3 compares the effect of buffer size (N) on the blocking probability (PBL) or loss probability when $\lambda = 5.2$, $\mu = 0.7$, $b = 3$ for various VMs. It can be seen that the blocking probability decreases when the buffer size increases. We further observe that for fixed buffer size, the blocking probability decreases as the number of VMs, c, increases. We can calculate the minimum buffer size required to maintain the blocking probability under a given pre-assigned measure.

Figure 7.4 shows the effect of service rate (μ) on the loss probability (PBL) for various batch size b. The parameters are taken as $\lambda = 5.0$, $c = 2$ and $N = 10$. It can be seen that for all values of b, the loss probability decreases rapidly when the service rate increases. Further with fixed service rate, blocking probability decreases when the batch size increases. Therefore, we can infer that the blocking probability depends on the choice of batch size b to a greater extent.

Figure 7.5 depicts the effect of traffic load (ρ) on the average waiting time (W_q) when $N = 15$, $b = 4$ for various values of VMs (c). It can be

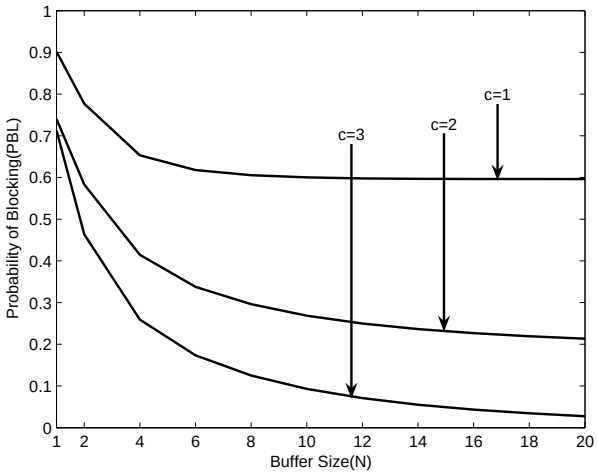

Figure 7.3: Effect of buffer size on PBL with varying c.

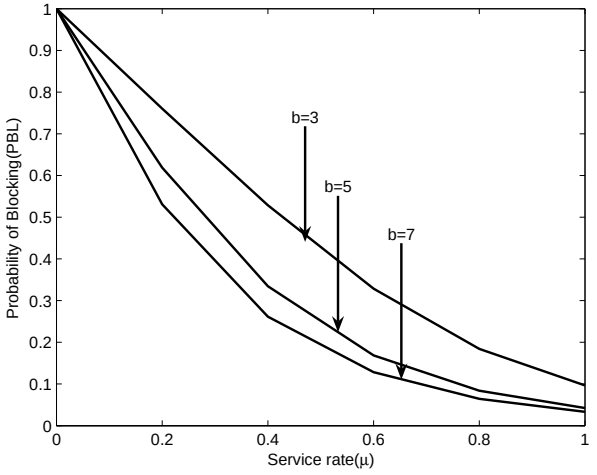

Figure 7.4: Effect of μ on PBL with varying b.

Table 7.1: Steady state probabilities .

(n,r)	$b=2,c=3,N=10$ $\lambda=4.2,\mu=0.9$	$b=3,c=4,N=15$ $\lambda=4.2,\mu=0.9$	$b=3,c=2,N=20$ $\lambda=1.2,\mu=0.7$
(0,0)	0.009164	0.012116	0.178089
(0,1)	0.042766	0.056539	0.305296
(0,2)	0.099787	0.131926	
(0,3)		0.205218	
(0,c)	0.155223	0.239421	0.261682
(1,c)	0.130887	0.142970	0.129132
(2,c)	0.110572	0.085373	0.063722
(3,c)	0.093029	0.050982	0.031445
(4,c)	0.078971	0.030444	0.015517
(5,c)	0.065741	0.018177	0.007657
(6,c)	0.057104	0.010858	0.003779
(7,c)	0.045159	0.006483	0.001865
(8,c)	0.043668	0.003865	0.000920
(9,c)	0.026581	0.002320	0.000454
(10,c)	0.041348	0.001378	0.000224
(11,c)		0.000811	0.000111
(12,c)		0.000518	0.000055
(13,c)		0.000279	0.000027
(14,c)		0.000150	0.000013
(15,c)		0.000175	0.000007
(16,c)			0.000003
(17,c)			0.000002
(18,c)			0.000001
(19,c)			0.000000
(20,c)			0.000000
Sum	1.000000	1.000000	1.000000
PBL	0.041348	0.000175	0.000000
L_q	2.936500	0.878913	0.503288
W_q	0.729322	0.209302	0.419407

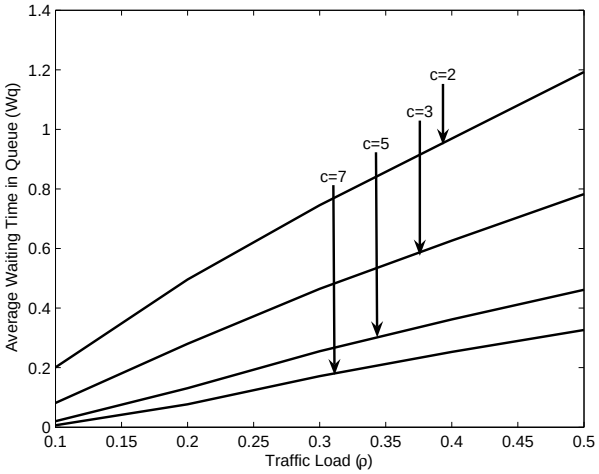

Figure 7.5: Effect of ρ on W_q with varying c.

observed that for all values of c, the average waiting time increases as traffic load increases. When VMs are more the average waiting time is less as compared to less number of VMs. The gap being widening up considerably for larger traffic intensity. Clearly, the optimal choice of c is not always simple but becomes more critical for higher variation of traffic intensity. Further, we could observe that change in the traffic intensity and the number of VMs affect the delay performance of queue.

Figure 7.6 illustrates dependence of blocking probability of the buffer size N varying from 1 to 30 and the batch size b varying from 1 to 7. The parameters are taken as $\mu = 0.7$, $\lambda = 5.2$ and $c = 3$. We note that for a fix batch size as the buffer size increases the loss of probability decreases. With fixed buffer size the loss of probability increases as the batch size increases. Thus we have to setup an permissible batch size and the sufficient buffer size in the cloud system to maintain lower blocking probability.

The variation in the queueing delay for different values of the number of servers and the batch size is shown in Figure 7.7, when $\lambda = 10.0$, $\mu = 1.0$ and $N = 10$. The number of VMs (c) is varied from 1 to 8, where as the batch size b is varied from 1 to 10. It is found that the average waiting time

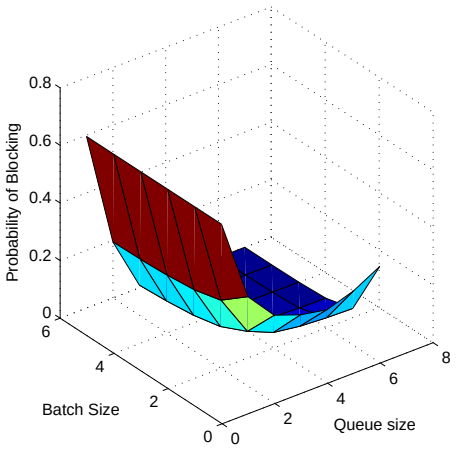

Figure 7.6: Effect of blocking probability for different values of b and N.

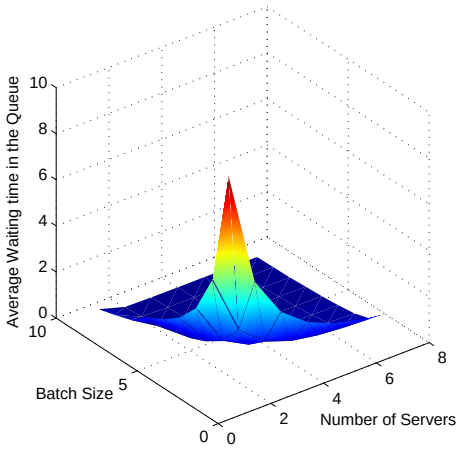

Figure 7.7: Effect of W_q for different values of c and b.

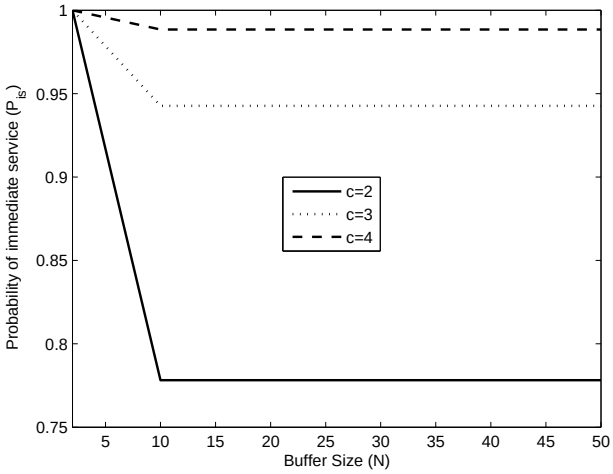

Figure 7.8: Effect of P_{is} on buffer size.

decreases as the number of VMs increases for fixed batch size. Therefore we need to carefully set the number of VMs and the batch size in the system in order to guarantee the minimal average waiting-time in the waiting buffer.

Figure 7.8 depicts the buffer capacity on probability that a task gets immediate service without queueing. The parameters are taken as $\lambda = 1.5$, $\mu = 1.75$ and $b = 3$. It can be seen that as the number of VMs increases probability that a task gets immediate service without queueing also increases. It requires the presence of at least one idle VM but it should not depend on the buffer capacity. Its value is close to 1 at low buffer capacity, and then decreases and almost static as the buffer size increases. This is due to fact that at low buffer capacity an arriving task either gets blocked or immediately serviced. Hence, the increase of buffer capacity leads to decrease of both the probability of blocking and the probability of getting immediate service. However, increase of buffer capacity beyond a certain level may not noticeably affect either of the probability of blocking or the probability of getting immediate service.

7.6 Conclusion

Cloud computing enables to use hardware resources effectively and economically. It increases flexibility of organizations due to information sharing and collaboration, scalability, expediency in new service roll out, availability, and mobility. This chapter analyzes a queueing model for performance evaluation of cloud server farms for processing bulk data. This type of service framework intents to meet users' QoS requirements. Some important performance measures such as mean number of tasks in the queue, blocking probability, and probability of immediate service, as well as waiting-time distribution in the system have also been discussed. A variety of numerical results are carried out and reported in the form of table and graphs showing the effect of model parameters on key performance measures. The services and architecture of cloud computing contain some areas of concern like service processing in bulk. The result presented in this chapter suggest that processing services in bulk may lead to reduction of overall cost of the services. For example in cloud security measures, bulk implementation of filtering, authentication, access control measures, federated identity management, etc. may reduce the overall investment cost.