Performance of Cloud Computing Centers with Multiple Priority Classes

Wendy Ellens, Miroslav Živković, Jacob Akkerboom, Remco Litjens, Hans van den Berg
Performance of Networks and Systems
TNO
Delft, the Netherlands
Email: wendy.ellens@tno.nl, miroslav.zivkovic@tno.nl, jacob.akkerboom@tno.nl, remco.litjens@tno.nl, j.l.vandenberg@tno.nl

***Abstract*—In this paper we consider the general problem of resource provisioning within cloud computing. We analyze the problem of how to allocate resources to different clients such that the service level agreements (SLAs) for all of these clients are met. A model with multiple service request classes generated by different clients is proposed to evaluate the performance of a cloud computing center when multiple SLAs are negotiated between the service provider and its customers. For each class, the SLA is specified by the request rejection probabilities of the clients in that class. The proposed solution supports cloud service providers in the decision making about 1) defining realistic SLAs, 2) the dimensioning of data centers, 3) whether to accept new clients, and 4) the amount of resources to be reserved for high priority clients. We illustrate the potential of the solution by a number of experiments conducted for a large and therefore realistic number of resources.**

***Index Terms*—cloud computing, performance analysis, queueing theory, rejection probability, Service Level Agreement**

I. Introduction

Cloud computing is the new trend of computing where readily available computing resources are exposed as a service. A *cloud* is defined as both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services [1]. According to this definition, delivery of application as services (*SaaS* — Software as a Service) over the Internet and hardware services (*IaaS* — Infrastructure as a Service) are both parts of cloud computing phenomena. From hardware service (utility computing) point of view, there are a few new aspects in cloud computing [1], the most prominent being the illusion of infinite computing resources and the ability to pay for the use of computing resources on a short-term basis.

As consumers move towards adopting such a *Service-Oriented Architecture*, the quality and reliability of the services become important aspects. However the demands of the consumers vary significantly. It is not possible to fulfill all consumer expectations from the service provider perspective and hence a balance needs to be made via a negotiation process. At the end of the negotiation process, provider and consumer commit to an agreement, usually referred to as a *Service Level Agreement* (SLA). The SLA serves as the foundation for the expected level of service between the consumer

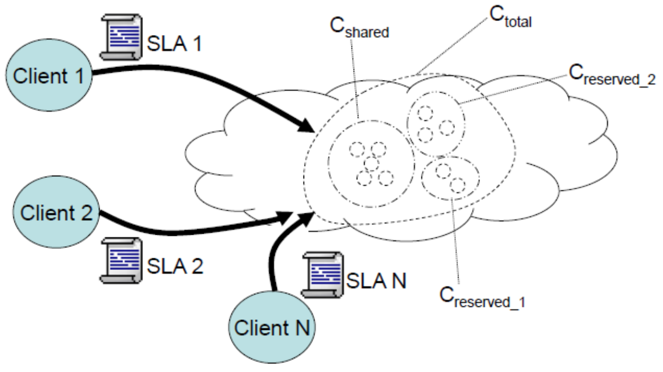


Fig. 1. Considered cloud computing infrastructure

and the provider. The quality of service (QoS) attributes that are generally part of an SLA (such as response time and throughput) however change constantly and to enforce the agreement, these parameters need to be closely monitored [2]. Accurately predicting customer service performance based on system statistics and a customer's perceived quality allows a service provider to not only assure QoS but also to avoid over–provisioning to meet an SLA. Due to a variable load derived from customer requests, dynamically provisioning computing resources to meet an SLA and allow for an optimal resource utilization is an important but complicated task.

As stated in [3], the majority of the current cloud computing infrastructure consists of services that are offered up and delivered through a service center such as a data center that can be accessed from a web browser anywhere in the world. In this paper we study a model for the cloud infrastructure shown in Figure 1, where a service provider offers multiple resources to its clients. In order to accommodate the clients' needs and to serve N clients, the service provider may decide, depending upon the agreed SLAs (one per client), to reserve a certain amount of resources exclusively for certain clients.

In our example at Figure 1, the service provider has decided to reserve $C_{\text{reserved\_1}}$ for client 1 and $C_{\text{reserved\_2}}$ for clients 1 and 2, of the total C_{total} resources at its disposal. Besides, all N clients can use C_{shared} resources. In such a way service provider could offer better service to "more significant" clients (e.g. business clients) who are probably willing to pay more for the negotiated service. If there are no available resources the request is rejected. This scheme corresponds to the trunc reservation policy as proposed earlier in ATM systems [8]. For our example, in case of requests originating from e.g. client 1, a request is rejected when all C_{total} resources are busy, while for client 2, this would occur when all C_{reserved_2} and C_{shared} resources are busy. Other clients' requests will be rejected when all C_{shared} resources are occupied. This means that high priority clients are more likely to get resources at busy times, in other words, the *rejection probabilities* are lower for requests from high priority clients. In cloud computing accepted requests indicate rewards for the administrator, while rejected requests can lead to penalties. We do not explicitely model these costs, instead we use the request rejection probabilities for different clients (higher priority clients are entitled to lower rejection probability) as main performance parameter of the considered system.

The proposed framework is of great value for cloud computing service providers as it supports them in the decision making about 1) defining realistic SLAs, 2) the dimensioning of data centers, i.e. determining the total resource capacity that is needed to meet the SLAs, 3) whether to accept new clients and how many, and 4) the amount of resources to be reserved for high priority clients. We develop a model to obtain the answers to the following specific questions:

1) What are realistic rejection probabilities that could be specified in SLAs offered by the cloud provider to its clients, when the arrival rates of service requests (for both classes), the mean service time at a single server in the cloud, C_{total} and C_{reserved} (total and reserved resources) are known?
2) For given arrival rates of service requests, a given mean service time and a given number of reserved resources, what should be the value of C_{total} in order to assure previously negotiated rejection probabilities for both classes?
3) In case the values of C_{total} and the fraction of high priority requests are known, and C_{reserved} is chosen in an optimal way, what is the maximum arrival rate of service requests such that previously negotiated rejection probabilities can be guaranteed?
4) In case the values of C_{total} and the fraction of high priority requests are known and the arrival rate of service requests is maximal (meeting the target rejection probabilities), what is the optimal value of C_{reserved}?

The model we use to describe cloud centers with several clients holding different SLAs corresponds to a $M/M/C/C$ queueing system with different priority classes. The arrivals are Poissonian, the service time is exponentially distributed, there are C servers and the system capacity is C (there are no buffers). In the literature queueing systems like the basic $M/M/C$ queue have been applied to cloud computing centers. Variations on this basic queueing system by changing the service time distribution, the buffer length, or considering batch arrivals can also be found, see e.g. [4], [5], [6]. We propose a model that can deal with different performance criteria for different clients by reserving parts of the computing capacity for specific clients.

Hu et al. [7] describe a cloud computing model close to ours. They also have two priority classes with different SLAs. One of our goals corresponds theirs, namely determining the minimal needed capacity for a given load, but we consider also other questions. However, their model is different in terms of resource allocation. They compare two setups: 1) both classes have their own resources and 2) both classes share the resources. We consider the following setup: 3) part of the resource is shared by both classes and part of the resources is reserved for the class with the SLA that is most difficult to meet.

The paper is organized as follows: in Section II we describe the model of the cloud computing system we used for our analysis. Section III covers the mathematical approach for calculating the rejection probabilities. Next, in Section IV, we give some numerical examples in order to show the potential of our framework for cloud computing management. We conclude the paper with a summary of our main achievements and indicate the possible directions for further research in Section V.

II. A Cloud Computing Model

In this section we describe a model of a cloud computing center with multiple priority classes. A schematic representation of the model has been depicted in Figure 2.

We consider a cloud computing environment that serves requests from a total of N clients. The clients' requests are served by provider that has a total of C_{total} resources. The available resources C_{total}, are split into shared resources, C_{shared}, and reserved resources, C_{reserved}, i.e. $C_{\text{total}} = C_{\text{shared}} + C_{\text{reserved}}$. Shared resources are used to serve the requests originating from any client, while the reserved resources C_{reserved_j} are exclusively used to process the requests originating from clients $i \leq j$. The total of reserved resources is $C_{\text{reserved}} = \sum_{j=1}^{N} C_{\text{reserved}\_j}$. We assume in our model that $C_{\text{shared}}, C_{\text{reserved}} > 0$ while $C_{\text{reserved}\_j} \geq 0, j = 1, \ldots, N$.

The concept of reserved resources allows the provider to prioritize requests originating from different clients. Requests from high priority clients (clients i for which $\sum_{j=i}^{N} C_{\text{reserved}\_j} > 0$) are accepted as long as there are less than $C_{\text{shared}} + \sum_{j=i}^{N} C_{\text{reserved}\_j} = C_{\text{total}} - \sum_{j=1}^{i-1} C_{\text{reserved}\_j}$ request being processed at the moment, while other requests (from clients for which $\sum_{j=i}^{N} C_{\text{reserved}\_j} = 0$) are accepted whenever less than C_{shared} resources are used. For an illustration of the principle see Figure 2.

As the number of potential clients that independently generate requests is large, we assume that requests arrive according to a Poisson process. The rate at which new requests from client i arrive is denoted by λ_i. The time it takes to process a request is modeled following an exponential distribution, with the same average process time $1/\mu$ for all requests.

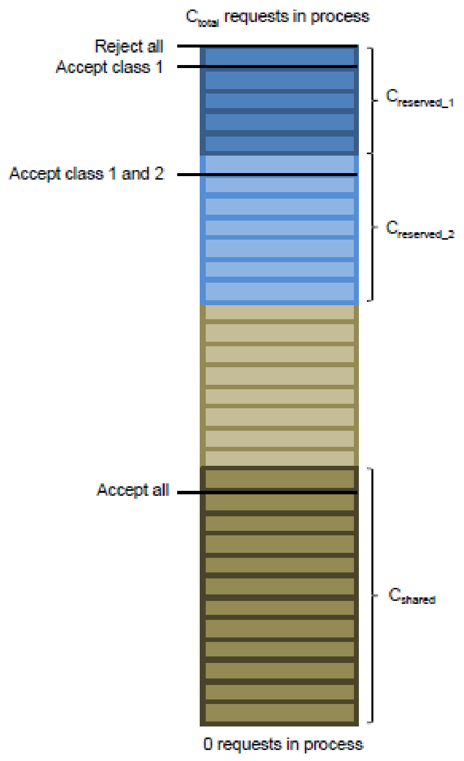


Fig. 2. Schematic representation of the general model

The exponential distribution of the process time allows exact analysis of the rejection probabilities.

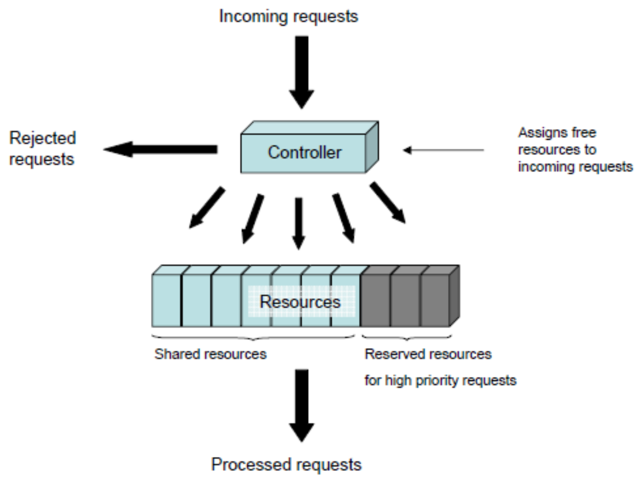
In order to evaluate whether the SLA for client i will be met for the given configuration of the cloud computing center (i.e. for the given values of $C_{\text{total}}, C_{\text{reserved}\_j}, j = 1, \ldots, N$) and given arrival and departure processes (characterized by $\lambda_i, i = 1, \ldots, N$ and μ), the service provider needs to know the rejection probability p_i for request of client i. In the next section we discuss a method to analytically determine the rejection probabilities $p_i, i = 1, \ldots, N$ for all clients.

In order to simplify the notation, we use the above-described model with two priority classes, as illustrated in Figure 3. In this case, we have that $C_{\text{reserved}\_1} > 0, C_{\text{reserved}\_2} = 0$, thus $C_{\text{reserved}} = C_{\text{reserved}\_1}$. We therefore consider high priority clients and low priority clients, and we define high (low) priority requests as requests originated by a high (low) priority client. In case λ is the overall request arrival rate and q is the fraction of high priority requests, the arrival rate for high priority requests is $\lambda_{\text{high}} = q\lambda$ while the arrival rate for low priority requests is $\lambda_{\text{low}} = (1-q)\lambda$.

III. Theoretical Analysis

This section briefly discusses the mathematical theory of Markov chains, birth-death processes and queueing system [8], [9], [10]. We use this framework to calculate the probability that a cloud service request is rejected.

A *Markov chain* is a memoryless random process on a countable number of states, i.e. a system that moves between



Incoming requests
Rejected requests
Controller
Assigns free resources to incoming requests
Resources
Shared resources
Reserved resources for high priority requests
Processed requests

Fig. 3. Schematic representation of the model with two priority classes as used in the experiments of Section IV

a countable number of states and for which the transition probability from one state to another only depends on the current state, not on the system's history. *Birth-death processes* are *continuous-time Markov processes* — Markov chains with transitions that occur at random moments — where a state corresponds to a number $0, 1, 2, \ldots$ and only transitions between state i and $i+1$ (and reverse) are possible. An important application of birth-death processes are queueing systems, because in most queueing systems, the number of individuals/jobs in the system follows a birth-death process.

For birth-death processes it is possible to calculate the *stationary distribution* — giving the probability that the system is in a certain state in the long run. The rate at which transitions from i to $i+1$ occur is denoted $\lambda(i)$ (the arrival rate if i individuals/jobs are in the system) and the departure rate in state i is denoted $\mu(i)$. The stationary probability $p(k)$ for state k can be determined by solving the set of *balance equations* (which state that the flux into a state should be equal to the flux out of this state when the system is stationary):

$$\lambda(0)p(0) = \mu(1)p(1)$$

and

$$(\lambda(k) + \mu(k))\, p(k) = \lambda(k-1)p(k-1) + \mu(k+1)p(k+1)$$

for $k > 0$. Solving these equations gives

$$p(k) = \frac{\prod_{i=0}^{k-1} \lambda(i)}{\prod_{i=1}^{k} \mu(i)} p(0),$$

with

$$p(0) = \frac{1}{1 + \sum_{k>0} \frac{\prod_{i=0}^{k-1} \lambda(i)}{\prod_{i=1}^{k} \mu(i)}}.$$

Using the theory of birth-death processes, we are able to calculate the rejection probabilities for our model (see Section II), for given values of the class i arrival rates λ_i the service rate μ, the total capacity C_{total} and the reserved capacities $C_{\text{reserved\_}i}$ $(i = 1, \ldots, N)$. We will give an example of the calculation for the case of two priority classes. In terms of birth-death processes we have $\lambda(k) = \lambda = \lambda_1 + \lambda_2$ if k (the number of requests being processed) is less than C_{shared}. If k is equal to or greater than C_{shared} then $\lambda(k) = \lambda_1$, i.e. only high priority requests are admitted. The departure rate is $k\mu$ if there are k requests in process. The stationary probabilities are therefore

$$p(k) = \begin{cases} \dfrac{\lambda^k}{k!\mu^k} p(0) & \text{if } k \le C_{\text{shared}} \\ \dfrac{\lambda^{C_{\text{shared}}} \lambda_1^{k - C_{\text{shared}}}}{k!\mu^k} p(0) & \text{if } k > C_{\text{shared}}, \end{cases}$$

with

$$p(0) = \frac{1}{1 + \sum_{k=1}^{C_{\text{shared}}} \frac{\lambda^k}{k!\mu^k} + \sum_{k=C_{\text{shared}}+1}^{C_{\text{total}}} \frac{\lambda^{C_{\text{shared}}} \lambda_1^{k-C_{\text{shared}}}}{k!\mu^k}}.$$

The rejection probabilities are given by

$$p_{\text{high}} = p(C_{\text{total}}) \text{ and } p_{\text{low}} = \sum_{k=C_{\text{shared}}}^{C_{\text{total}}} p(k).$$

The rejection probabilities are computed using the following recursion:

$$r(0) = 1$$

and for $k > 0$

$$r(k) = \begin{cases} \dfrac{\lambda}{k\mu} r(k-1) & \text{if } k \le C_{\text{shared}} \\ \dfrac{\lambda_1}{k\mu} r(k-1) & \text{if } k > C_{\text{shared}}, \end{cases}$$

The stationary probabilities are

$$p(k) = \frac{r(k)}{\sum_{k=0}^{N} r(k)},$$

for all k. This recursive computational method allowed us to perform the simulations for a large number of resources. We present the results of these in the next section.

Although we have chosen to work with two priority classes, analytical results are also available for the case of N classes. It is also possible to perform similar analyses for an $M/M/C/C + R$ queueing system with different client classes, where a buffer of size R is present. In the case of "exclusive reserved resources" (meaning that clients cannot use the resources reserved for lower ranked clients), a multi-dimensional Markov chain needs to be solved to calculate the rejection probabilities, because we need to keep track of the class that requests in process belong to. Therefore no closed-form formula for the rejection probabilities is available, but they can be calculated by solving a system of equations. Also for the case where requests from different classes have a different process time, a multi-dimensional Markov chain arises. For the case we considered — where reserved resources are also available for higher ranked clients and all requests have the same process time distribution — closed-form formulas are available because it is enough to know the number of requests in process in order to decide whether an incoming request of a specific class is accepted.

IV. Numerical Examples

In this section we use the model described in Section II and the analytical results of Section III to give the answers to the questions posted in Section I. The numerical examples given in this section show how our framework can assist in solving practical cloud management issues. Subsection IV-A describes the scenario we considered in our numerical examples, including the values (or ranges) of the model parameters. The following subsections each answer one of the following questions:

1) What are realistic rejection probabilities that could be specified in SLAs offered by the cloud provider to its clients, when the arrival rates of service requests (for both classes), the mean service time at a single server in the cloud, C_{total} and C_{reserved} (total and reserved resources) are known? (Subsection IV-B)
2) For given arrival rates of service requests, a given mean service time and a given number of reserved resources, what should be the value of C_{total} in order to assure previously negotiated rejection probabilities for both classes? (Subsection IV-C)
3) In case the values of C_{total} and the fraction of high priority requests q are known, and C_{reserved} is chosen in an optimal way, what is the maximum arrival rate of service requests such that previously negotiated rejection probabilities can be guaranteed? (Subsection IV-D)
4) In case the values of C_{total} and q are known and the arrival rate of service requests is maximal (meeting the target rejection probabilities), what is the optimal value of C_{reserved}? (Subsection IV-E)

A. Scenario Description

Due to the fact that we used efficient calculations of the rejection probabilities based on the recursive formulas given in section III, we have been able to run the simulations for high numbers of servers. Therefore, different from e.g. [7], [6] we could perform our simulations for numbers of servers that better reflect the actual numbers. In practice, large cloud computing service providers have a total amount of servers of the order of tens of thousands [11]. We performed the simulations for a total number of servers up to $C_{\text{total}} = 40000$. In order to illustrate the importance of the number of requests generated by clients in different priority classes, parameter q that represents the fraction of the service requests made by high priority customers has been set to change between 0.2 and 0.8 with step 0.2 i.e. 20%, 40%, 60% and 80% of the total requests. Without loss of generality (by scaling the arrival rate λ) the parameter that represents the expected service time is set to one, i.e. $\mu = 1$. The rejection probabilities promised by the service providers to its higher priority and lower priority clients are $p^*_{\text{high}} = 0.5\%$ and $p^*_{\text{low}} = 5\%$, respectively. This means that the rejection probability for the

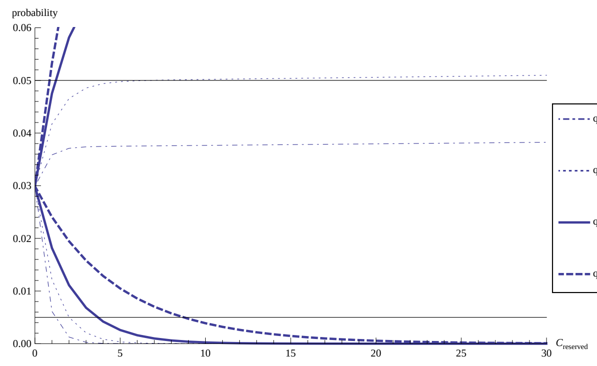


Fig. 4. The rejection probabilities for high (lower curves) and low (upper curves) priority jobs as a function of the reserved resources C_{reserved}. Here $C_{\text{total}} = 40000$, $\lambda = 1.03 \cdot C_{\text{total}}$ and $\mu = 1$. The curves have been drawn for $q = 0.2, 0.4, 0.6, 0.8$.

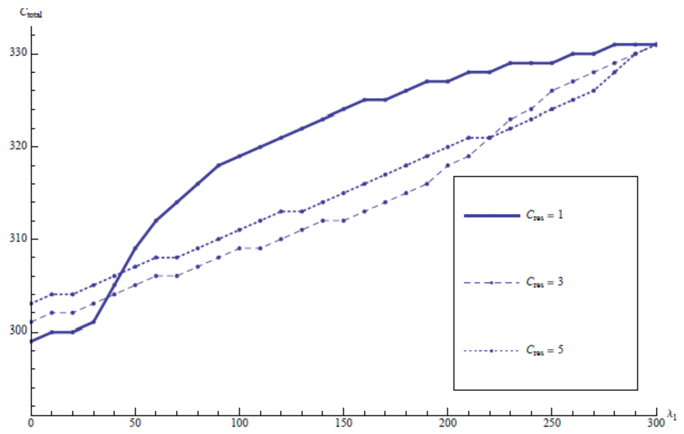


Fig. 5. The minimal number of resources C_{total} as a function of the arrival rate of high priority requests λ_1. The overall arrival rate λ is 300, $\mu = 1$, $p^*_{\text{high}} = 0.5\%$ and $p^*_{\text{low}} = 5\%$. The curves have been drawn for $C_{\text{reserved}} = 1, 3, 5$.

high (lower) priority clients is never higher than p^*_{high} (p^*_{low}). It could be lower depending on the dimensioning of the data center, i.e. the concrete values for C_{total} and C_{reserved}.

B. Determination of Realistic Rejection Probabilities

In this scenario, we calculate the rejection probabilities p_{high} and p_{low} for high and low priority requests as a function of the reserved resources C_{reserved}, in order to determine realistic target rejection probabilities for both classes. In Figure 4 we give the results for $C_{\text{total}} = 40000$ servers, and the total arrival rate of all requests (high and low priority) $\lambda = 1.03 \cdot C_{\text{total}}$. The rejection probability curves have been drawn for the four above-mentioned values of q, the fraction of high priority requests. We have shown the most informative part of the curves, which corresponds to the interval $0 \le C_{\text{reserved}} \le 30$.

The following observations can be made from the figure:

- It always holds that $p_{\text{low}} \ge p_{\text{high}}$, with equality when number of reserved resources C_{reserved} is zero.
- When C_{reserved} increases, p_{low} increases, while p_{high} decreases.
- The probability a high priority customer is rejected vanishes quickly as the number of reserved resources increases. Even when the percentage of requests generated by high priority customers is 80% it is enough to have only nine reserved servers (out of 40 thousand) to guarantee rejection probability $p^*_{\text{high}} = 0.5\%$.
- For fixed C_{reserved} the rejection probabilities for both classes, p_{high} and p_{low}, are increasing in the fraction of high priority requests q, because the more high priority requests (relatively), the lower the overall rejection probability (for a random incoming request which can have high priority as well as low priority), the fuller the system (i.e. the higher the probability of having between C_{shared} and C_{total} requests in process) and the higher the probability that a new high or low priority request (i.e. a request of a given class) is rejected.
- When the majority of requests is generated by high priority clients, more specifically when $q \ge 0.6$, service provider cannot guarantee the target rejection probabilities of $p^*_{\text{high}} = 0.5\%$ and $p^*_{\text{low}} = 5\%$. For example, when $q = 0.6$ the rejection probability p_{low} can be guaranteed only when number of reserved resources is less than 2. On the other hand, when $q = 0.6$ rejection probability p_{high} can be guaranteed only when of the number of shared resources is greater than 4. These two requirements cannot both be satisfied, which means respective SLAs should allow higher rejection probabilities. The graph can be used to determine the minimal target rejection probabilities, the minimal p^*_{high} and p^*_{low}, for a given value of C_{reserved}.
- For $q = 0.4$, $C_{\text{total}} = 40000$, $\lambda = 1.03 \cdot C_{\text{total}}$ and $\mu = 1$, $p^*_{\text{high}} = 0.5\%$ and $p^*_{\text{low}} = 5\%$ are realistic target rejection probabilities if $2 \le C_{\text{reserved}} \le 6$. Similarly, for $q = 0.2$, the given target probabilities are realistic if $2 \le C_{\text{reserved}} \le 424$.

Let us illustrate how figures like Figure 4 can be used: The above observations show that a data center with 40000 servers that handles $1.03 \cdot 40000 = 41200$ requests per second with an average service time of 1 second, having 40% of high priority clients, can guarantee rejection probabilities of 0.5% and 5% for high and low priority requests respectively if it reserves only 2 (up to 6) servers for high priority clients.

C. Determination of the Minimal Number of Servers that Guarantees Rejection Probabilities Smaller than the Targets

In the second scenario we determine what should be the value of C_{total} in order to prevent SLA violations, for a given rate of service requests per customer class and a given number of reserved resources for high priority requests C_{reserved}. We have set the total arrival rate λ to 300 (which means that requests arrives 300 times as fast as they can be processed as $\mu = 1$), and rejection probability targets are, as usual, $p^*_{\text{high}} = 0.5\%$ and $p^*_{\text{low}} = 5\%$ for high, respectively low, priority requests. We have considered values λ_1 (the arrival rate of high priority requests) from zero to 300, with increments of 10. Therefore varying the percentage of the high priority requests within the range 0% – 100%. For each value of λ_1 and given value of C_{reserved} from the set $\{1, 3, 5\}$, we have determined the minimal C_{total} for which the rejection probability targets are met.

These results are illustrated in Figure 5. We draw the following conclusions from the figure:

- The minimal number of servers needed increases almost linearly with the high priority arrival rate λ_1. The only exception is when the number of reserved resources, C_{reserved} is very small, i.e. one.
- The endpoints of the curves show us the following. If all clients have the same low priority ($\lambda_1 = 0$) we need approximately 300 servers to serve 300 requests per second if the process time is 1 second. If we want to decrease the rejection probability for all the clients from 5% to 0.5% (i.e. if we only have high priority clients) we need approximately 330 servers.
- The minimal number of servers depends strongly on the number of reserved servers — the more high priority requests, the higher the number of reserved resources should be in order to minimize the total number of resources.
- We can roughly identify three areas in the graph. In the first area, which represents the area with very low arrival rate of high priority customers, the number of reserved resources should be kept as low as possible, which gives the lowest number of C_{total}. This results from the fact that, in order to serve many requests originating from low priority clients, the number of resources that serve them ($C_{\text{total}} - C_{\text{reserved}}$) should be as high as possible. The higher C_{reserved}, the smaller $C_{\text{total}} - C_{\text{reserved}}$ and the harder it becomes to guarantee p_{low}. Therefore, the only option for the provider(s) is to increase C_{total}.
- In the second area, which represents the area of moderate arrival rate of high priority requests, $C_{\text{reserved}} = 1$ is inferior to the other values of C_{reserved} considered. We see that, for $C_{\text{reserved}} = 1$, C_{total} should be high in order to accommodate relatively many high priority requests. For $C_{\text{reserved}} = 3$ and $C_{\text{reserved}} = 5$ the situation is similar to that of the first region — the number of reserved resources is over-dimensioned for $C_{\text{reserved}} = 5$ leading to increase of C_{total} in order to accommodate low priority requests as well.
- In the last area, which represents the area of high arrival rate of high priority requests, the number of reserved resources plays an important role in order to accommodate all high priority requests. Therefore, C_{total} is smallest for the highest values of C_{reserved}.

Figure 5 shows that 312 servers are enough to accommodate 150 high priority and 150 low priority requests per second if the average process time is 1 second and not more than 0.5%, respectively 5% of the requests, for high and low priority clients, may be rejected.

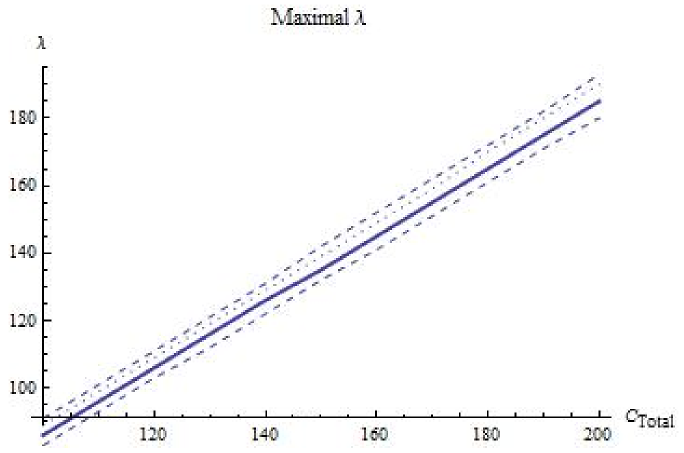


Fig. 6. The maximal arrival rate λ as a function of the available resources C_{total} such that the target rejection probabilities are met. Here, C_{reserved} is chosen in an optimal way, $\mu = 1$, $p^*_{\text{high}} = 0.5\%$, $p^*_{\text{low}} = 5\%$ and $q = 0.2, 0.4, 0.6, 0.8$ for the different curves.

D. Determination of the Maximal Arrival Rate that Guarantees Rejection Probabilities Smaller than the Targets

In this simulation scenario we determine the maximal total arrival rate, λ, for a given number of resources C_{total} and the optimal value of C_{reserved}. The criterion for the determination of λ is that the target rejection probabilities for all client classes are satisfied. Therefore, for $\mu = 1$, a given fraction of high priority requests q (again we have four graphs for $q = 0.2, 0.4, 0.6, 0.8$), and given target probabilities $p^*_{\text{high}} = 0.5\%$ and $p^*_{\text{low}} = 5\%$, we have determined λ as a function of the total number of available resources C_{total}. We illustrate the results for C_{total} that varies in the range 100–200 in Figure 6.

From this figure, one notices that there is an almost linear relation between λ and C_{total}, with a linearity coefficient of 1. For every extra server, the arrival rate can increase by one unit. For example, when the average process time is 1 second and the available resources increase from 150 to 200, the data center can accommodate 50 more requests per second. The other way around, when a service provider needs to process more requests (i.e. higher arrival rates) than initially given, the necessary increase in number of resources is linear in the number of extra arrivals per time unit.

Using these results, and based on the exact number of resources at its disposal, a cloud provider can determine the number of requests they can serve. Knowing the actual number of clients, the method gives a way to calculate how many new clients can be accommodated.

E. Determination of the Number of Resources to be Reserved for High Priority Clients

The last scenario is related to the third scenario. Again the arrival rate λ is maximized for a given number of servers C_{total} such that the rejection probabilities do not exceed the targets of $p^*_{\text{high}} = 0.5\%$ and $p^*_{\text{low}} = 5\%$. We are now interested in the value of C_{reserved}, the amount of resources reserved for high priority clients, for which the maximal arrival rate is attained.

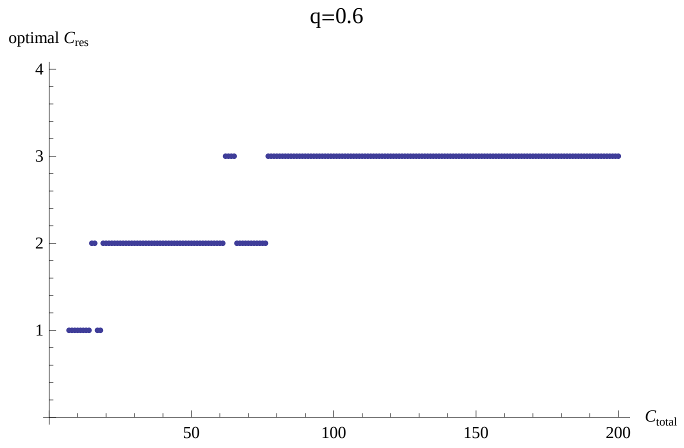


Fig. 7. The number of reserved resources C_{reserved}, that maximizes the arrival rate λ as a function of the available resources C_{total} such that the target rejection probabilities are met. We have $p^*_{\mathrm{high}} = 0.5\%$, $p^*_{\mathrm{low}} = 5\%$ and $q = 0.6$ for the different curves.

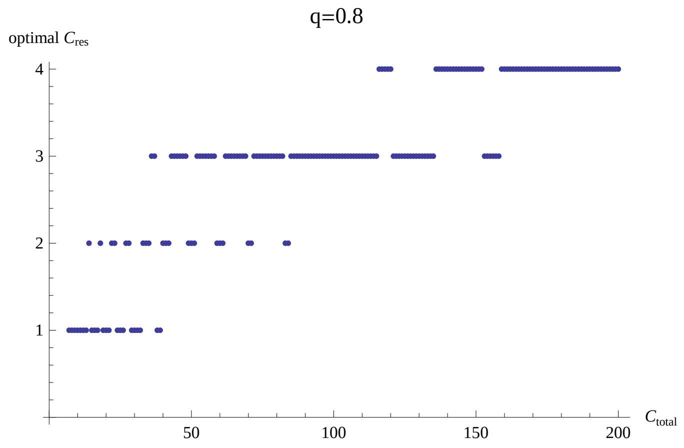


Fig. 8. The number of reserved resources C_{reserved}, that maximizes the arrival rate λ as a function of the available resources C_{total} such that the target rejection probabilities are met. We have $p^*_{\mathrm{high}} = 0.5\%$, $p^*_{\mathrm{low}} = 5\%$ and $q = 0.8$ for the different curves.

The results are illustrated in Figures 7 and 8 for values $q = 0.6$ and $q = 0.8$, respectively.

In general we see that the higher the total number of resources, the higher the optimal number of reserved resources, but the lower the optimal fraction of reserved resources. We see that there are "overlapping intervals", i.e. that the optimal number of reserved resources is not increasing in the total number of resources. For example, in Figure 7 we see that $C_{\mathrm{reserved}} = 2$ for $C_{\mathrm{total}} = 19$, and $C_{\mathrm{reserved}} = 1$ for $C_{\mathrm{total}} = 20$. It would be advisable to have the number of C_{reserved} as given in Table I.

Another conclusion that can be drawn from Figures 7 and 8 is that the higher the percentage of requests generated by high priority customers, the more resources need to be reserved for these (for a fixed number of total resources).

We have seen in Subsection IV-D that a cloud center with

TABLE I
THE NUMBER OF RESOURCES C_{reserved} THAT SHOULD BE RESERVED FOR HIGH PRIORITY CLIENTS, FOR THE GIVEN VALUES OF C_{reserved} AND q. THE TARGET REJECTION PROBABILITIES ARE $p^*_{\mathrm{high}} = 0.5\%$ AND $p^*_{\mathrm{low}} = 5\%$.

q	Interval C_{total}	C_{reserved}
0.8	1–12	1
0.8	13–38	2
0.8	39–118	3
0.8	119–200	4
0.6	1–18	1
0.6	19–62	2
0.6	63–200	3

an average process time of 1 second can serve 50 more clients per second if it increases its resources from 150 to 200. Figure 7 shows that it does not have to change the number of reserved resources if it has 60% of high priority clients.

V. Conclusions and Future Work

In this paper we analyzed the general problem of resource provisioning within cloud computing. In order to support decision making with respect to resource allocation for a cloud resource provider when different clients negotiated different service level agreements (SLAs) we have modeled a cloud center using the $M/M/C/C$ queueing system with different priority classes. The main performance criterion in our analysis is the rejection probability for different customer classes, which can be analytically determined. We have shown that a number of common questions providers may have — about realistic SLAs, dimensioning of data centers, acceptance of new clients and reservation of resources — can be answered using this result.

We have conducted a number of experiments for the case of two priority classes (corresponding to high and low priority clients) with realistic (i.e. high) number of available resources. These experiments show that 1) it is possible to offer a minority of important clients request rejection probabilities that are ten times smaller than the request rejection probabilities of other clients by only reserving a small fraction of the available resources for important requests, 2) the minimal number of servers increases approximately linearly with the fraction of high priority requests, 3) the maximal number of clients increases almost linearly with the available resources and 4) the higher the number of servers, the smaller the fraction that needs to be reserved.

Our model assumes class-dependent arrival rates and a high number of resources, in contrast to traditional applications of queueing theory and rejection probability formulas, such as telecommunication, in which the total capacity is generally small. The main reason for the simplicity of our model is that we consider situations in which different priority classes have the same average process time. In further research we plan to relax this assumption. No closed-form formulas for the rejection probabilities are available for this case, but a system of equations has to be solved. We further plan to investigate batch arrivals and time-dependent arrival processes.

In addition, we plan to do a cost analysis including rewards (penalties) for accepted (rejected) calls (which are higher for high priority clients) and costs for resources.

Acknowledgment

Part of this work has been carried out in the context of the IOP GenCom project Service Optimization and Quality (SeQual), which is supported by the Dutch Ministry of Economic Affairs, Agriculture and Innovation via its agency Agentschap NL. The authors kindly acknowledge initial paper discussions with Behnaz Shirmohamadi.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, pp. 50–58, April 2010. [Online]. Available: http://doi.acm.org/10.1145/1721654.1721672

[2] A. Keller and H. Ludwig, "The wsla framework: Specifying and monitoring service level agreements for web services," *J. Netw. Syst. Manage.*, vol. 11, pp. 57–81, March 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=635430.635442

[3] http://en.wikipedia.org/wiki/Cloud_computing.

[4] B. Yang, F. Tan, Y. Dai, and S. Guo, "Performance evaluation of cloud service considering fault recovery," *Cloud Computing*, pp. 571–576, 2009.

[5] T. Kimura, "Optimal buffer design of an m/g/s queue with finite capacity*," *Stochastic Models*, vol. 12, no. 1, pp. 165–180, 1996.

[6] H. Khazaei, J. Mišić, and V. Mišić, "Performance analysis of cloud centers under burst arrivals and total rejection policy," in *IEEE Globecom*, 2011.

[7] Y. Hu, J. Wong, G. Iszlai, and M. Litoiu, "Resource provisioning for cloud computing," in *Proceedings of the 2009 Conference of the Center for Advanced Studies on Collaborative Research*. ACM, 2009, pp. 101–111.

[8] J. Roberts, U. Mocci, and J. Virtamo, Eds., *Broadband Network Teletraffic*. Springer, 1996.

[9] H. Tijms and J. Wiley, *A first course in stochastic models*. Wiley Online Library, 2003, vol. 2.

[10] L. Kleinrock, "Queueing systems. volume 1: Theory," 1975.

[11] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.