An Efficient Cloud Design for Email as a Service with Massive Time Constrain Messages

Mauricio Alejandro Gomez Morales, Aymen Abdullah Alsaffar and Eui-Nam Huh (Corresponding author)
Department of Computer Engineering, Kyung Hee University
1 Seocheon-dong, Giheung-gu, Yongin, Gyeonggi-do, South Korea
Tel: +82-31-201-3778   E-mail: {mgomez, aymen, johnhuh}@khu.ac.kr

Abstract
Before Cloud and Grid computing paradigms become visible, there were some things that were not possible to make. Now that parallel computing concept has spread a little bit more, the new trend is to enhance services which are already provided and develop new tools that did not exist before now. In this paper, we concentrate our work in the massive email delivery with a short time constraint. In other words, we concentrate in a real time system that needs to deliver a message to a big list of email destinations within a small time constraint. The main focus in this paper is that cloud providers can have different locations to offer a more acquainted service that can result in less effort and less cost, if workload allocation is made in different places and closer to corresponding destinations. Within this work, we propose a solution which calculates the exact Virtual Machine (VM) provision that scales to the required computation and network resources to give answer to the load charge that represents the massive email requirement.

Keywords: Real time systems, cloud computing, optimal resource utilization, SLA, Location Aware, massive email sending.

1. Introduction
One of the most widely used communication way around the globe is electronic mail (email) that is more accessible and reachable throughout the world. It is the most suitable way of quick communication for the users, regardless of their level of technical expertise and it provides an automatic delivery service allowing users, separated by location and time, to exchange electronic messages quickly [1]. When the message is needed to be sent in a legal way (excluding to be taken as spam) to a huge number of destinations, it is classified as Massive Email.

The most important idea behind Cloud Computing is scalability and the key technology that makes that possible is virtualization [3,4]. One of the fields that can be enhanced is email sending, specifically, massive email communication. As we mentioned before about the importance and ease in having massive email delivery as the most convenient way of communication, it is essential to utilize this facility more efficiently with cloud computing resources. When the message is sent to various destinations, it is needed to be delivered according to the requirement of a Service Level Agreement (SLA). In this case, the system will be considered as Real Time System [5]. Adoption of cloud resources for this purpose is not yet investigated in a real manner until now. A bottleneck could be noticeable when there is communication resource sharing. This resource bottleneck is measured in terms of available bandwidth of a network path $P$ which is the maximum throughput that $P$ can provide to a flow, without reducing the throughput of the cross traffic in $P$ [6] as shown in Figure 1.

Thus, in this paper, we propose an efficient cloud design for email as service with massive time constrains messages. We focus our study in solving the problem of sending massive email where we are taking the advantage of the resources that can be offered by a cloud service provider (CSP). This work, we calculate the number of VMs that is needed to accomplish the SLA requirement of the client. Furthermore, we perform a deep study of the contention that can be generated according to the bottleneck of the structure provided trying to find efficient solution to avoid it.
Finally, we show a comparison of the costs which represent all the options that the user can decide according to its needs.

We examine and answer the following questions:

- What is the number of VMs required, taking into account the network bottleneck problem?
- What would be the best combination of datacenter/VMs by network resource?
- What would be the best solution in terms of cost for that solution?
- With datacenters in different locations, would it improve the system? Is the improvement only considered in terms of effort saving or does it also imply less cost?

Our main contribution is to develop a model that dynamically calculates the combination of datacenters-VMs by network which is needed to accomplish a specific deadline for delivering a massive email, taking into account and solving network contention problem and if the cloud service provider is able to provide those services from different locations around the world as well, then, determine the best load charge distribution by location.

The reminder of this paper is organized as follows: section 2 we describe the related works. In section 3 we present the problem formalization. In section 4 we present our simulation results. In section 5 we describe our conclusion and future work.

![Figure 1. Bottleneck Network Problem Diagram](image)

### 2. Related Work

Some research has been done on designing and implementing an email-based personal cloud storage that aggregates back-end services by establishing a RAID-like system [7]. In [8] the author develop an architecture that eliminate data duplication in storage of email systems in order to save storage space and improve performance of email server. On the other hand, the majority of research area of the mixture in terms of email and cloud are more focused in security and privacy issues, in which they propose a scalable cloud spam analysis framework to address the shortcoming of deploying a server based spam filtering toolkit into a commercial cloud computing environment [9]. In [10] the author focus on the problem of email contents disclosure and establish a secure mail server by using Postfix in Linux platform and then implement it into a cloud service provider as Infrastructure as a Service (IaaS). From the perspective view of service level agreements (SLA), a protocol is presented for optimistic certified email which relies on a trusted third party (TTP) and a time stamping server (TSS) [11]. Most of the previous research works only focus on delivery to destination. However, in our work we focus on time delivery and to ensure that message is delivered to user destination as well.

Some research is done in regard to minimizing delivery time. In [12] the author only improves the SMTP protocol to...
minimize delivery time. It does not consider the problem of cloud environment as well as real time system. In [13] the author presents a survey that provides a benefit to the cloud users and researchers to overcome the challenges faced where it discussed the problem of resource allocation strategies. The difference between this work and our paper is that our study is focused on the delivery of email, the resources that generate the bottleneck, and network bandwidth. In our previous work [14] we presented a model to reduce the delivery time of mass email by using the cloud based resources within efficient use of resources. However in this new work, we are considering other factors such as the contention given the difference of time between each command sent and the round trip time (RTT) for each phase of SMTP. Furthermore, we consider the bottleneck bandwidth shared that generates network contention.

3. Problem Formalization

In this section we define the problem formalization. We will present flow diagram of proposed solution and problem solution.

3.1 Flow Diagram of Proposed Solution

Figure 2 illustrates a mixture of the SMTP workflow with our proposal that present all the steps and parts of the process which is based on SMTP. The platform as a service (PaaS) cloud service provider (CSP) provides the virtual machines and configured with the email environment.

Figure 2. Flow Diagram of Proposed Solution

3.2 Problem Definition

Figure 3 illustrates the problem that shows the individual parts of sending a massive email. The shown numbers refer to the quantity of accounts for each destination domain. If we needed to send all emails to 405,350 receivers using single server, then the total lasting time would be 29.39 hours.
Figure 3. Massive E-mail Sending Problem (Number of Emails Accounts)

This problem becomes more difficult to solve when the user also demands the email to be delivered to all the accounts in less than a specified time constraint (e.g. 10 minutes). The reason behind taking more than 29.39 hours for single server to accomplish this task is that SMTP protocol still requires that email has to be sent one at a time using all seven steps by each email account even though they are sent to the same domain.

Figure 4. Urgent Massive Email Delivery Basic Diagram

Nowadays, Cloud providers have datacenters in different locations that can be accessed by their end-users and consumed for their own purposes where they are only paying for the time they use those specific resources. In [15] our previous works, we only calculate the number of VMs to use in one specific location. As it is illustrated in Figure 4, we extend our work by considering the number of VMs that is needed if we distribute the load charge in different locations taking advantage of sending the massive email in a parallel way. Figure 5 illustrate the problem where the round trip time will be different for each combination of location/domain.
We proceed to show our formulas definition. Based on SMTP protocol, send a single email required to send data and wait 7 times in order to receive respond. SMTP allows sending emails by a single connection in a sequential way which will eliminate 3 steps from sent email account. However now days, there are a lot of email spam and to avoid that we used a variable to determine how many emails are going to be sent for each connection. We represent it as \( E \).

### 3.2.1 Calculation for Required VMs

In this section, we focus in the number of VMs that are needed to accomplish the deadline where we made more detailed study considering the time spent to wait for each of the phases of the SMTP. The average time \( t \) to send a massive email can be obtained by the division of \( TT \) or total time to send the email by the total number of emails \( N \) as shown below:

\[
E = \frac{n \Sigma_{i=1}^{n} \left( \frac{S \times E}{B_{th}} + W_{i} \left[ 2(E+1)+1 \right] \right)}{N}
\]  

(1)

Where \( A' \) is the number of accounts of the \( i \)-est destination domain, \( S \) is the size of the email to send, \( E \) is the number of emails to send by connection. Then \( B_m \)

\[
B_m = \min(B_n/T, B_S/T, B_r/T)
\]  

(2)

\( B_n \) denote the minimum of bandwidth between network card, \( B_S \) denote switch communication, \( B_r \) denote connection to the cloud, all of them divided by the number of threads \( T \) that are going to be sent by each VM and \( W_i \) denote the round trip time (RTT) for the \( i \)-est destination domain. The value \( t \) is applied in the formulas of our previous work [14] instead of \( t_i \) where to obtain the number of VMs is required to get before deadline \( D \).

\[
D \geq 2E \sqrt{N}
\]  

(3)
We obtained the total number of VMs \( n \) that we need to allocate.

\[
n = \min \left[ \frac{1}{2} \left( \frac{D}{t} - \sqrt{\left( \frac{D}{t} \right)^2 - 4N} \right) \right]
\]  
(4)

3.2.2 Contention Analysis

In this section, we study the performance of different amount of threads or VMs that share some network resource and consequently generating a bandwidth bottleneck. Therefore, we use different number of threads or VMs to prove that the more processes/threads/VMs access the same resource at the same time, the more performance decreases which illustrated in figure 6.

![Figure 6. Bandwidth / Thread Behavior](image)

We defined contention as how many times sending data can fit in the round trip time (RTT). We denote the \( Q \) as of maximum allowed.

\[
Q = \text{round} \left[ \frac{W_i}{S/B_m} \right] + 1
\]  
(5)

It is determined for each of the destination domains. Then we can assume that the time which will take for one Datacenter \( T_{Ax} \) will be given by applying the correct \( Q \) to the number of VMs.

\[
T_{Ax} = \sum T_{A(S)_{x,l}} + T_{A(W)_{x,l}}
\]  
(6)

Where, \( T_{A(S)_{x,l}} \) correspond to the time of the sending part and the \( T_{A(W)_{x,l}} \) round trip time of the message to all the accounts of the \( l \)-est destination domain, respectively. In the next formula, we present the detail of the first part as follow:

\[
T_{A(S)_{x,l}} = B_m \times S \times \frac{A_l}{T}
\]  
(7)
And the second one as follows:

$$T_{A(y,x)} = \left[ B_m \times W_f \times 2 + X_i \times \left( W_f + T_{R(T,D)} - A_i \right) \times 2 \right] / R \times Q$$  \hspace{1cm} (8)

Where $X_i$ denotes the number or connections needed by each domain according to the number of emails sent by connection, this variable is determined as follows.

$$X_i = \frac{A_i}{E}$$  \hspace{1cm} (9)

In formula (8), the second part of the addition is multiplied by the round trip time (RTT) $W_f$ and by the time to send an instruction, $T_{R(T,D)}$, assuming that each time the it has to be sent one instruction which will take some minimum time.

3.2.3 Cost Analysis Comparison from Single Location

In this section, we conducted a comparison of the different combinations of datacenter-VMs that we will have to evaluate in order to get the deadline accomplished from a single location trying to minimize net contention problems and errors. Most importantly is the solution that represents less cost in terms of money to be spent for the rent of the resources needed. We define the group of possible solution $a_i$ as the resulting time of the combinations of datacenters-VMs dividing the total number of VMs that is explained as follows.

$$a_y = \text{ceiling} \left( \frac{R}{E} \right), \forall y < n$$  \hspace{1cm} (10)

Where $y$ shows all the iterations that can be applied to the number of datacenters with a maximum of total number of $n$ VMs. To these values of period is being applied the Ceiling function to obtain integer positives and no fractions, as it is not possible to rent in fractions in the cloud.

$$f(T_a) = \frac{R}{A}$$  \hspace{1cm} (11)

At the end, what is required is to find out the best option for the combination of datacenter and VMs allocated that really accomplishes the deadline required in terms of efficiency and cost; this best option comes by using the next formula.

$$\text{Best} = \text{Min} \left[ P \left( C_v \times \frac{R}{T_{R(T,D)}} + C_c \times y \right) + C_y \right] V_{a_y}$$  \hspace{1cm} (12)

Where each $C$ represents the cost of VMs, datacenter structure and minutes of error combinations respectively, multiplied by $P$, the period of time required and the times of each one, $T_{R(T,D)}$ in the case of VMs and $y$ for datacenters.

3.2.4 Allocation for Minimization of VMs

Here we present an analysis to get the best allocation strategy that will be adopted according to the load charge distribution of the email list. To solve this new objective, we present some formulas in this section. First is the calculation of the best time among the different origins for each of the destinations that is a simple minimum operation as is shown next.
\[ \forall: \hat{W}_1 = \min \{ W_{1,1}, W_{1,2}, \ldots, W_{1,l} \} \]  

(13)

This calculation also represents that the load charge is going to be distributed in equivalent dissemination. Then we proceed to apply this value in all the formulas that show this variable \( W_1 \).

But also the calculation now changes in the point that for each of the different locations is going to be summarized the time to get a number of VMs that are needed for each of those places. Basically, the value calculated has to be differentiated by each origin. This is how the first direct modification that we are going to get is the enhancement to formula (1) where the average time to send an email finishes as is shown next, note that the symbol changed from to only to symbolize that is the average but also the lowest time within all the locations for each email domain destination.

\[ E_1 = \frac{\sum A_1 \left[ \frac{S \times E}{B_n} + \hat{W}_1 \right]}{N} \]  

(14)

As the enhancement in formula (3) is only made by the symbol shown in formula (14) we are going to skip to show the change. However, the real important enhancement is the one that we get in formula (4) where it will be used for all the different locations as is presented in next formula.

\[ \forall 1: n_3 = \min \left[ \frac{2}{t} - \sqrt{\frac{2}{n}} \right] \]  

(15)

Then, the value of each location has to be summarized to be able to compare it with the quantity that we can get by every location calculated in an individual way, in other words, how it was considered for our previous work [15]. This is how the formula to have the number of VMs by each location summarized is as follows.

\[ S_n = \sum n_3 \]  

(16)

The best allocation schema is going to be determined as the one that requires less effort, in this case # of VMs to be rented, as is displayed in the formula below;

\[ BS = \min (n_1, n_2, \ldots, n_i, s_n) \]  

(17)

4. Simulation Results

We generated a scenario where an end user requests an email to be sent to a list of 405,350 email accounts that belongs to 15 different domains, as illustrated in Figure 3. Also, we consulted [16] that provides a service that can give us a hint of how much time it could take to wait for an answer of the specified domains that will be shown after. The three different locations that can be calculated in that service are: Amsterdam, New York, and Dallas. So, all our calculations are based in the times that we got from the mentioned webpage and are shown above in Table.
Table 1. The Request Load Charge with Time Response by Location

<table>
<thead>
<tr>
<th>Domain (i)</th>
<th>Accounts (A_i)</th>
<th>Round Trip Time (W_{i,l})</th>
<th>Amsterdam</th>
<th>New York</th>
<th>Dallas</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.seoul.go.kr">www.seoul.go.kr</a></td>
<td>25000</td>
<td></td>
<td>0.452</td>
<td>0.302</td>
<td>0.290</td>
</tr>
<tr>
<td><a href="http://www.jeju.go.kr">www.jeju.go.kr</a></td>
<td>10000</td>
<td></td>
<td>0.294</td>
<td>0.238</td>
<td>0.205</td>
</tr>
<tr>
<td><a href="http://www.atlantaga.gov">www.atlantaga.gov</a></td>
<td>12300</td>
<td></td>
<td>0.990</td>
<td>0.019</td>
<td>0.026</td>
</tr>
<tr>
<td><a href="http://www.andong.go.kr">www.andong.go.kr</a></td>
<td>35000</td>
<td></td>
<td>0.360</td>
<td>0.264</td>
<td>0.213</td>
</tr>
<tr>
<td><a href="http://www.muniguate.com">www.muniguate.com</a></td>
<td>10000</td>
<td></td>
<td>0.168</td>
<td>0.085</td>
<td>0.088</td>
</tr>
<tr>
<td><a href="http://www.melbourne.vic.gov.au">www.melbourne.vic.gov.au</a></td>
<td>32000</td>
<td></td>
<td>0.375</td>
<td>0.275</td>
<td>0.219</td>
</tr>
<tr>
<td><a href="http://www.nycgo.com">www.nycgo.com</a></td>
<td>51000</td>
<td></td>
<td>0.074</td>
<td>0.016</td>
<td>0.028</td>
</tr>
<tr>
<td><a href="http://www.visitguatemala.org">www.visitguatemala.org</a></td>
<td>12300</td>
<td></td>
<td>0.024</td>
<td>0.052</td>
<td>0.042</td>
</tr>
<tr>
<td><a href="http://www.kigalicity.gov.rw">www.kigalicity.gov.rw</a></td>
<td>54000</td>
<td></td>
<td>0.182</td>
<td>0.324</td>
<td>0.285</td>
</tr>
<tr>
<td><a href="http://www.dhakacity.org/">www.dhakacity.org/</a></td>
<td>34500</td>
<td></td>
<td>0.191</td>
<td>0.025</td>
<td>0.045</td>
</tr>
<tr>
<td><a href="http://www.busan.go.kr">www.busan.go.kr</a></td>
<td>23150</td>
<td></td>
<td>0.382</td>
<td>0.234</td>
<td>0.209</td>
</tr>
<tr>
<td><a href="http://www.suwon.go.kr">www.suwon.go.kr</a></td>
<td>17800</td>
<td></td>
<td>0.301</td>
<td>0.219</td>
<td>0.214</td>
</tr>
<tr>
<td><a href="http://www.hotmail.com">www.hotmail.com</a></td>
<td>12300</td>
<td></td>
<td>0.044</td>
<td>0.029</td>
<td>0.067</td>
</tr>
<tr>
<td><a href="http://www.gmail.com">www.gmail.com</a></td>
<td>25000</td>
<td></td>
<td>0.006</td>
<td>0.002</td>
<td>0.003</td>
</tr>
<tr>
<td><a href="http://www.yahoo.com">www.yahoo.com</a></td>
<td>51000</td>
<td></td>
<td>0.009</td>
<td>0.003</td>
<td>0.004</td>
</tr>
</tbody>
</table>

Table 1 illustrate the time for round trip time (RTT) per data/command sent (not by email) to different domains from different possible locations of CSP. In addition, part of the request is defined by variables in Table 2.

Table 2. General Variables and their values

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email Size (Y)</td>
<td>1 Mb</td>
</tr>
<tr>
<td>Deadline required (D)</td>
<td>10 minutes</td>
</tr>
<tr>
<td>Threads required (T)</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 2 illustrate general variables and their values such as, Email size with the value of 1MB, Required deadline with value of 10 minutes and required thread with value of 5 threads.

Table 3. Request of CSP Variables and their values

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIC Bw (E_a)</td>
<td>100 Gb</td>
</tr>
<tr>
<td>Backbone/switch B_s (E_b)</td>
<td>10 Gb</td>
</tr>
<tr>
<td>Router/Internet B_r (E_r)</td>
<td>512 Mb</td>
</tr>
<tr>
<td>Cost for VM/hour (E_c)</td>
<td>$10</td>
</tr>
<tr>
<td>Cost for Datacenter / hour (E_d)</td>
<td>$50</td>
</tr>
<tr>
<td>Cost for Error/min (E_e)</td>
<td>$25</td>
</tr>
</tbody>
</table>

Table 3 illustrate the request of Cloud service provider variables and their values such as NIC BW with value of 100gb, Backbone/switch with value of 10gb, Router/Internet with value of 512mb, Cost of VM/hour with value of $10, Cost of Datacenter / hour with value of $50, and Cost for Error/min with value of $25.
Figure 7 illustrate the time with respect to the number of thread/VMs in 1 Datacenter. It shows that when more threads or VMs share the same network resource, then the time will reaches one point where it is not decreasing and as a result, it is impossible to meet the requested deadline.

Figure 8 illustrate the result of randomly evaluating the performance for 10 threads on each VM and how it behaves to the change in the number of VMs. The real line was applied on the formula where depends on how many real VMs are going to be able to send without causing more contention and delay on the delivery time. The normal curve shows the behavior of sending time if there was no contention. Both curves are compared against the goal of 10 minutes in the same figure.

Figure 8 illustrate the real vs. normal time with different VMs with 10 threads per machine in the same Datacenter. Beyond 10VMs for the real case, when we have 10 threads for each VM, we notice that the time improvement is minimum when you compared to the normal case that could meet the SLA if there were no contention. This is known as network resource saturation. By examining the result, we are motivated us to think about the best combination of Datacenters and VMS which will accomplish the requested deadline. We did this by applying the formula (1) of $10, Cost of Datacenter/hours with value of $50 dollar and Cost of Error/min with value of $25.
formula (9) to obtain the combination mentioned in figure 8 and find out that when we divide the number of VMs by the number of Datacenters with their own network resource we will get a simple decreasing curve. After that, not only the load charge is distributed in the different combinations of Datacenter-VMs but also that the time and the cost are going to vary, derived from applying the formula to obtain best α using formula (10) as illustrated in figure 9.

Figure 9. Time by Datacenter-VM of 5 Threads Combination

Figure 9 illustrate the result of time for all the combinations of Datacenter-VM from 1 to 12 Datacenters. The curves changes in an oscillatory way because we cannot rent fractions of Datacenters or VMs only on 1 by 1 basis. It is impossible to have a value with decimals. Therefore, we applied the ceiling function at the end of it. The best option in this case is to have 6 Datacenters with 7 VMs per Datacenter.

Figure 10. Cost Analysis

In Figure 10, we present how the variation of cost evolves according to the three factors that we considered: VMs, Datacenter and Error where we applied formula (12), which indicates that the best choice is to have 7 VMs for each of the 6 datacenters in the same location. We proceeded to evaluate each of the locations, this is how the values that we got for Amsterdam are as follows in Figure 11, where, for 1 thread it is impossible to accomplish the SLA, because of network contention, and as is shown starting from 2 threads, then the VMs required are 192 for 256 Mb. of bandwidth. In addition, for 50 threads in the 256 Mb. It is impossible to accomplish the SLA for the same reason of network contention. As is shown in the table 1 round trip table, response times for Amsterdam are the highest with 512 Mb. As for the bandwidth, the best option is 10 threads because it only requires 42 VMs, while for 1 Gb, it is 15 threads.
As for New York, if we were comparing only the options of 256 Mb against Amsterdam, it could be able to accomplish the deadline renting 272 VMs and for that bandwidth the best option is to use only 5 threads, that give us as a result 67 VMs, because after that, due to network contention the number of VMs are increased for 256 Mb until we get to 50 threads that also, because of network contention, then it is not possible to accomplish the deadline. For the cases of 512 Mb and 1 GB, the recommended threads are 10 and 15 respectively, and VMs required are 44 and 29. The curve looks similar for 10 and 100 GB, where the best option is 50 threads. In this example, as it was evaluated only until 50 threads. All the result mentioned in this paragraph is shown in Figure 12.

Figure 13 illustrate Dallas VMs required for different threads. After evaluating Dallas location, we notice that its behavior is similar to New York but with better performance in almost all cases. This is because of the response time for almost all the email destination domains is lower than Amsterdam and New York. Analyzing a little deeper for 256 Mb, it starts with 241 VMs required and showing like the best option for 256 Mb with 5 threads with 58 VMs and showing that with 50 threads, it is impossible to accomplish the deadline of 10 minutes because of network contention as is shown by the three locations evaluated in an individual way. For the cases of 512 Mb and 1 Gb, the
best options are also 10 and 15 threads with 38 and 28 VMs respectively, but it is worth to mention that the difference between 10 and 15 threads for 1 Gb. is only of 1 VM as for 10 threads the VMs required are 29. As for 10 and 100 Gb. the behavior is the same as others. Comparing bandwidth with other locations, where more threads are applied, the less the VMs are required then, as illustrated in Figure 13.

![Figure 13. Dallas VMs required for different threads](image)

We wanted to see if there is a solution that could require less VMs which would be considered as less effort. We also made the analysis mentioned and illustrated from formulas (13) to (16). We made also the same graph for the best time per email destination domain as illustrated in Figure 14. Due to network contention for 256 Mb, 512 Mb and 1 Gb, the curve is parabolic, where the best option is 5, 10, and 10 threads that represent 58, 41, and 28 VMs required respectively. As for 10 and 100 GB, also is the same behavior than for the three locations individually, but in terms of VMs required the best time distribution is the better option.

![Figure 14. Best time scenario VMs required for different threads](image)

We made a comparison of the three locations and the best time in the same graph, but to be able to compare them, we did it only for 5 threads, and results are illustrated in Figure 15, where only for 256 Mb, the best option can be chosen between Dallas and best time scenario with 58 VMs required by each of them. However, for all the other bandwidths evaluated, that are 512 Mb, 1 and 10 Gb, the best option is the scenario created as the best time, because the values for all the different bandwidth were less for best time scenario except for the 256 Mb bandwidth.
5. Conclusion and Future Work

In this paper was made a research about massive email delivery to accomplish a short time constraint and how can it be done with less effort with different globally distributed locations that CSP can provide to users. The analysis shows a better performance because it requires less number of VMs in a 14% lower than what is needed for the location of Dallas that is the best for individual evaluation. This solution considers network contention generated when different VMs/threads try to use the same network resources. Furthermore, we made a cost evaluation of the solution to find out the best combination of datacenter-VMs-threads, in term of cost for single location where the best option is 7 VMs for each of the 6 datacenters, that is also the first not incurring in SLA failure.

As future work, we want to take this simulation into real situation as well as making a study to evaluate the live system where we are able to allocate more resources when it is necessary. The reasons that motivate us to think about it is because, as in the cloud, everything is unknown, maybe communication performance would be deteriorated.

References


