Cloud Computing: Locally Sub-Clouds instead of Globally One Cloud

Nawsher Khan, Universiti Malaysia Pahang, Malaysia
Noraziah Ahmad, University Ahmad Dahlan, Indonesia
Tutut Herawan, University Ahmad Dahlan, Indonesia
Zakira Inayat, University of Engineering & Technology, Peshawar, Pakistan

ABSTRACT
Efficiency (in term of time consumption) and effectiveness in resources utilization are the desired quality attributes in cloud services provision. The main purpose of which is to execute jobs optimally, i.e., with minimum average waiting, turnaround and response time by using effective scheduling technique. Replication provides improved availability and scalability; decreases bandwidth use and increases fault tolerance. To speed up access, file can be replicated so a user can access a nearby replica. This paper proposes architecture to convert Globally One Cloud to Locally Many Clouds. By combining replication and scheduling, this architecture improves efficiency and easy accessibility. In the case of failure of one sub cloud or one cloud service, clients can start using another cloud under “failover” techniques. As a result, no one cloud service will go down.

Keywords: Cloud Computing, Effectiveness, Efficiency, Replication, Scheduling, Scheduling Techniques, Sub-Cloud

INTRODUCTION
Efficiency, scalability and easy accessibility are the key factors and should be the key features of cloud computing. From end-user computing, data storage and data transferring requirements are growing. Users demand for more capacity, more reliability and the capability to access information from anywhere in the world. Cloud services (computing, storage and transferring) meet this demand by providing transparent, easy and reliable solutions. Since late 2007 the concept of cloud computing was proposed (Weiss, 2007) and it has been utilized in many areas with some success (Brantner, Florescu, Graf, Kossmann, & Kraska, 2008; Moretti, Bulosan, Thain, & Flynn, 2008). Cloud computing is deemed as the next generation of IT platforms that can deliver computing as a kind of utility (Buyya, Yeo, Venugopal, Broberg, & Brandic, 2009). Foster, Yong, Raicu, and Lu (2008) made a comprehensive comparison of grid computing and cloud computing.

By a cloud, we mean an infrastructure that provides resources and/or services over
the Internet. A storage cloud provides storage services (block or file based services); a data cloud provides data management services (record-based, column-based or object-based services); and a compute cloud provides computational services. Often these are layered (compute services over data services over storage service) to create a stack of cloud services that serves as a computing platform for developing cloud-based applications. Examples include Google’s Google File System (GFS), BigTable and MapReduce infrastructure (Dean & Ghemawat, 2004; Ghemawat, Gobioff, & Leung, 2003). Amazon’s S3 storage cloud, SimpleDB data cloud, EC2 compute cloud (Amazon, 2009); and the open source Hadoop system (Borthakur, 2007; Dean & Ghemawat, 2008). Figure 1 shows the simple architecture of cloud computing.

For the majority of applications, databases are the preferred infrastructure for managing and archiving data sets, but as the size of the data set begins to grow larger than a few hundred terabytes, current databases become less competitive with more specialized solutions, such as the storage services (e.g., Borthakur, 2007; Dean & Ghemawat, 2008) that are parts of data clouds. For example, Google’s GFS manages Petabytes of data (Hbase Development Team, 2009).

Cloud architectures are middleware services for different purposes i.e., resource allocation management, job scheduling, security, authorization and data management etc. When a user requests a file, a large amount of bandwidth could be spent to send the file from the server to the client and the delay or response time involved could be high (Bsoul, Al-Khasawneh, EddienAbdallah, & Kilani, 2011; Ben Charrada, Ounelli, & Chettaoui, 2010a). Besides that, maintaining local copies of data on each accessing site are cost prohibitive while storing all data in a centralized manner is impractical due to remote access latency.
(Shorfuzzaman, Graham, & Eskicioglu, 2010; Shorfuzzaman, Graham, & Eskicioglu, 2011; Sashi & Thanamani, 2010a, 2010b). This may lead the Internet turns to be the bottleneck in accessing the files in the Cloud Computing. Due to the high latency of the Wide Area Network (WAN), the main issue is to design the strategy for efficient data access and share around the world with considerably low time complexity in Data Grid and Cloud research (Zhao, Xu, Xiong, & Wang, 2008). Furthermore, in order to manage the data there are another several problems must be considered such as failures or malicious attacks during execution, fault tolerance, scalability of data and etc. These problems can be solving by using the replication techniques (Bsoul, Al-Khasawneh, Eddien-Abdallah, & Kilani, 2011; Naseera & Murthy, 2009; Ben Charrada, Ounelli, & Chettaoui, 2010a, 2010b; Shorfuzzaman, Graham, & Eskicioglu, 2010; Zhao, Xu, Xiong, & Wang, 2008; Al-Mistarihi & Yong, 2009; Shorfuzzaman, Graham, & Eskicioglu, 2011; Zhao, Xu, Wang, Zhang, & He, 2010).

Globally, we are sending nearly 3 million emails per second, 20 hours of videos are uploaded to YouTube just in 60 seconds, Google processes 24 petabytes of information, publishing 50 million tweets per day, nearly 73 products are ordered on Amazon for every second (Swamkant, 2011). Symantec cloud provider process more than 6 billion emails and 1 billion web requests every day for businesses of every size, global corporations and Governments (Symantec Cloud, 2011) IDC cloud research report shows that worldwide revenue from public IT cloud services exceeded $37 billion in 2009 and is forecast to reach $55.5 billion in 2014 and in 2020 it will hit 900 billion, this growth representing a compound annual growth rate of 27.4%. This rapid growth rate is over five times the projected growth for traditional IT products (5%) (IDC, 2011).

The estimated amount of data 1,200 Exabyte (1,200 Billion GB) has been generated only during 2010. The amount of digital information increased by 73 percent in 2008 to an estimated 487 billion GB, according to IDC, in this report the population of world was 360,985,492 in 2000 which increased to 6,767,805,208 in 2009, and the number of Internet user in 2009 was 1,802,330,457, it means that the overall internet user growth is 399.3% with the increase of population. As a result, when the number of hosted applications and the complexity of the whole system grow, it is a serious challenge for PaaS providers to deliver satisfied services as they promised (Shao, Wang, & Mei, 2012).

The migration of business to cloud computing can be translated into saving of the software license, maintenance, number of support labor, utilities and office space (Karadsheh & Alhawari, 2011). User of cloud computing can, and have, created a virtual server room on one desktop. 100 million virtual machines are being created per year or 273,972 per day or 11,375 per hour. The number of physical servers in the World today is 50 million. By 2013, approximately 60% of server workloads will be virtualized means will convert to virtual cloud (IDC, 2011). With the popularization and improvement of social and industrial IT development, the people put much higher expectations on the services of computing, communication and network (Sasikala & Chaturvedi, 2011; Khan, Noraziah, Ismail, & Deris, 2012). Hence, for the better management of this day by day increasing heavy storage data, we need efficient scheduling as well replication techniques.

The rest of this paper is organized as follows. First, we explain the related works and elaborate the problems to be solved. Afterwards, we explain the proposed methodology. Then, we deliberate the results and discussion. Finally, we then conclude the paper.

RELATED WORK

There are some recent and related works that address the problem of scheduling and/or replication as well as the combination between them. However, first time Ranganathan and Foster (2003) proposed the realization of importance of data locality in job scheduling problem. The authors presented a Data Grid...
architecture base on three main components i.e., External Scheduler (ES), Local Scheduler (LS) and Dataset Scheduler (DS). ES receives submitted jobs from user, then depend on ES’s scheduler policies, it decide which job to send to which remote site. How to schedule all the jobs, LS of each site decide on its local resources. Keeping track of popularity for each dataset currently available and making data replicating decision. Nguyen and Lim (2007) and Tang, Lee, Tang, and Yeo (2006) have improved the older Ranganathan and Foster (2003) works by integrating the scheduling and replication strategy to improve the scheduling performance.

PROBLEM STATEMENTS

i. Queuing Architecture: Analyzing the works of Tang et al. (2006), Ranganathan and Foster (2003) and Nguyen and Lim (2007), the authors have proposed new scheduling architecture. For the integration of scheduling and replication, in this stage, the author has focused on Total Completion Time (TCT) for a job. In above works, the authors use the following formula for Total Completion Time for a job.

\[ TT_{k,i} = \max \{ QT_{(i)}, DT \{ f(k), i \} \} + ET_{k,i} \]  

\[ ETTC_{j} = \max \{ DT \{ f(j), i \}, QT_{(i)} \} + EET_{j} \]  

Where TT is Total Completion Time, QT is Queuing Time, DT is Data Transfer time, ET is the job Execution Time and in equation 2, ETTC is Estimate The Time for Completion and EET is Estimate Execution Time.

Here in \( \max \{ DT \{ f(j), i \}, QT_{(i)} \} \), one value DT either QT is ignoring, because only one of them will be maximum value, the other minimum value is ignoring. Even both are two different parameters, and have their importance separately.

ii. Cloud Services Outage: Cloud computing provides many services (XaaS). Anything can be a XaaS, means anything can be a cloudservice, i.e., Architecture As A Service (AaaS), Framework As A Service (Faas), Network As A Service (NaaS), Hardwar As A Service (Haas), Voice As A Service(VaaS), Recovery-As-A-Service (Raas), Data As A Service (DaaS) etc. Author’s previous work (Stallings, 2000) is describes more about cloud services. Generally “As a whole cloud will never go down,” but a service can (as mentioned in Table 1). When Gmail or some other hosted service has an interruption, it does not indicate a cloud failure, it’s a service failure. Table 1 gives more detail about the outage of various services of cloud, with duration.

Due to these outages cloud does offer unmatched efficiency, flexibility and reliability. As a result cloud computing fail to keep satisfied its user. In reaction going back to an in-house data center requires higher capital costs of hardware, maintenance, and in many cases would require hiring back more IT staff.

The clients (end user) are facing outage problems because globally Cloud is one and there is no other cloud for substitution to take over (using failover techniques) in this critical situation. A secure cloud model must be capable of providing reliability to the customer on the location of the data for customer. Hence here our challenge is how we can improve data efficiency, easy accessibility, strong reliability and always availability by keeping cloud service available forever.

PROPOSED METHOD

For the improvement of data efficiency, easy accessibility, strong reliability and always availability, the current research proposes an approach in order to overcome these qualities in cloud services. We need to make six (let say) local sub clouds on the basis of six sub-
continents instead of one Global cloud. Each local sub cloud will have same anytime updated replica (copy) of each other. Due to the local cloud, data in a wisely manner will offer a faster access to files required by cloud client, hence increase the job execution’s performance. Due to local cloud, reliability and accessibility will increase. If one sub cloud goes down, any client from anywhere in the world can use another sub cloud under the “failover” techniques as depicted in Figure 2.

Data availability and easy accessibility are paramount for cloud storage providers, as data loss or unavailability can be damaging both to the bottom line, by failing to hit targets set in service level agreements (Monash, 2011) and to business reputation outages often make the news (http://wiki.cloudcommunity.org). Data availability and easy accessibility are typically achieved through under-the-covers replication.

Dealing with large amount of data makes the requirement for efficiency, in data access, more critical. A good scheduling strategy will allow shortest access to the required data; therefore reduce the data access time. Vice versa, replication strategy that allows place data in a wisely manner will offer a faster access to required files. The goal of replication is to shorten the data access not only for user accesses but enhancing the job execution performance. For this approach we need an architecture, in which scheduling is compatible with replication.

**REPLICATION TECHNIQUE**

In data replications architecture, the data will be replicated into present available sub clouds. If one of sub cloud failed, it will fail independently and will not affect the others node. Therefore, the data replication will improve the reliability, availability of the data and performance. Replication in distributed environment receives particular attention for providing efficient access to data, fault tolerance (Bsoul, Al-Khasawneh, EddienAbdallah, & Kilani, 2011) and enhance the performance of the system (Zhao, Xu, Wang, Zhang, & He, 2010; Noraziah, Deris, Norhayati, Rabiei, & Shuhadah, 2008). The replication strategy can minimize the time access to the file by creating many replicas and storing replicas in appropriate locations, which provide nearby data access. Furthermore, using replication is to reduce bandwidth consumption (Naseera & Murthy, 2009; Ben Charrada, Onelli, & Chettaoui, 2010a, 2010b; Shorfuzzaman, Graham,

<table>
<thead>
<tr>
<th>Service</th>
<th>Duration Days/Hours/Minutes</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook (Co, 2011)</td>
<td>1h</td>
<td>Aug 10, 2011</td>
</tr>
<tr>
<td>Amazon Web Services (2011)</td>
<td>11 h</td>
<td>Apr 21, 2011</td>
</tr>
<tr>
<td>Amazon (Hickey, 2011b)</td>
<td>52 m</td>
<td>Apr 21, 2011</td>
</tr>
<tr>
<td>Foursquare (<a href="http://status.foursquare.com/">http://status.foursquare.com/</a>)</td>
<td>4 h</td>
<td>Aug 9,10,25, 2011</td>
</tr>
<tr>
<td>Foursquare (<a href="http://status.foursquare.com/">http://status.foursquare.com/</a>)</td>
<td>33 m</td>
<td>Oct 22, 2011</td>
</tr>
<tr>
<td>Amazon S3 (Hickey, 2011a)</td>
<td>30 m</td>
<td>Aug 16, 2011</td>
</tr>
<tr>
<td>Amazon EC2 (Butcher, 2011)</td>
<td>2.4h</td>
<td>Apr 21, 2011</td>
</tr>
<tr>
<td>Amazon (Miller, 2010)</td>
<td>8h</td>
<td>May 8, 2011</td>
</tr>
<tr>
<td>Microsoft Sidekick (Williams, 2010)</td>
<td>6 d</td>
<td>Mar 13, 2009</td>
</tr>
<tr>
<td>Microsoft Azure (Williams, 2010)</td>
<td>22 h</td>
<td>Mar 13, 2009</td>
</tr>
<tr>
<td>Google GMail, Google Apps (Williams, 2010)</td>
<td>24 h</td>
<td>Aug 15, 2008</td>
</tr>
</tbody>
</table>
& Eskicioglu, 2010) to achieve efficient and dependable data access to improve access time (Monash, 2011; Noraziah, Deris, Norhayati, Rabiei, & Shuhadah, 2008) fault tolerance (Bsoul, Al-Khasawneh, EddienAbdallah, & Kilani, 2011) and load balancing (Zhao, Xu, Wang, Zhang, & He, 2010).

There are three fundamental questions (Zhao, Xu, Wang, Zhang, & He, 2010; Ben Charrada, Ounelli, & Chettaoui, 2010a, 2010b; Zhao, Xu, Xiong, & Wang, 2008; Tang, Lee, Tang, & Yeo, 2006; Al-Mistarihi & Yong, 2009; Ranganathan & Foster, 2001) that must be answered in managing replica placement strategy in data grid/cloud:

- a. When should the replicas be created?
- b. Which files should be replicated?
- c. How many replicas should be created?

Different replication strategies have been developed or designed to answer these questions above. For the solution, the current research proposes an approach in order to improve data efficiency, easy accessibility, strong reliability and forever availability; these qualities are the key needs of cloud services. We need to make six clouds on the basis of six cotenants instead of one cloud globally. Each local sub cloud will have same replica (copy) of each other anytime updated. Due to the local cloud, data in a wisely manner will offer a faster access to files require by cloud client, hence increase the job execution’s performance. Due to local cloud, data will be reliable, no risk for losing data in presence of other replicas. Accessibility will increase data locality, data will access from local near cloud instead of global faraway cloud. Quick Access of data due to shortest distance because the execution performance is greatly impacted by the data locality (Ranganathan & Foster, 2001; Nguyen & Lim, 2007; Khan, Noraziah, Deris, & Ismail, 2011).

To fulfill this approach the ROWA (Read-Once-Write-All) is the most suitable Model. ROWA is the simplest technique for managing replicated data. A read operation is allowed to read any copy of data, meanwhile, a write operation is required to write all copies of data to the target places. All of the replicas have the same.

Figure 2. Simple sub-cloud structure for the globe
value when an update transaction commits. This technique has the lowest communication cost of read operation, because the read operation is using only once.

**SCHEDULING ARCHITECTURE**

The scheduling architecture is encapsulated in three distinct modules, as shown in Figure 3.

- **External Scheduler (ES):** Each user is associated with an External Scheduler in the system and submits jobs to that External Scheduler. ES decides the remote site to which to send the job to depending on some scheduling algorithm. ES uses external information for taking decision as input such as load at a remote site or the location of a dataset.

- **Local Scheduler (LS):** Assigned jobs are managing by Local Scheduler to run at a particular site. The allocation of allocated jobs is also responsibility of the LS. LS decide about the priority and refusing to run jobs submitted by a certain user.

- **Dataset Scheduler (DS):** At each site DS keeps track the popularity of each locally available data set. By using external information such as whether the data already exist at the concern site and loading to target remote site.

With encapsulation of three schedulers ES, LS, and DS, the Time for Completion of a job is also encapsulation of three *Times*. $W_q$ or $t_1$ is that time, which a job passing in waiting after entrance and before starting execution. Time denoted by $W_s$ or $t_2$ is the job execution time after entrance from queue and before starting transferring process. $T_t$ or $t_3$ is the time after completion execution time and upto the completion of data transferring process. Means the Total Completion Time for a job is the sum of all these three times (wait in queue, execution time and transfer time).

**QUEUING MODELS**

In queuing theory, a queuing model is used to approximate a real queuing situation or system, so the queuing behaviour can be analysed mathematically. These measures are important as issues or problems caused by queuing situations are often related to customer dissatisfaction with service or may be the root cause of economic losses in a business. Analysis of the relevant

---

*Figure 3. Scheduling architecture*
queuing models (Stallings, 2000; Blanc, 2011; Dombacher, 2009; Lipsky, 2009; Jain, Mohanty, & Bohm, 2007; Ng & Soong, 2008; Kobayashi, & Konheim, 1977; Philippe, 1998; Adan & Resing, 2008) allows the cause of queuing issues to be identified and the impact of proposed changes to be assessed. Queuing models can be represented using Kendall’s notation as in Dombacher (2009), Philippe (1998), and Adan and Resing (2008) where $A/B/S/K/N/D$

A: Is the inter-arrival time distribution or arrival rate per unit time, denoted by $\lambda$.

B: Is the service time distribution or service rate per unit time, denoted by $\mu$.

U: Is the server utilization, denoted by $\rho$.

S: Is the number of servers, denoted by $C$.

K: Is the queue system capacity, same notation $K$.

N: Is the calling population or total number of jobs, denoted by $P$.

D: Is the service discipline assumed, same notation $D$.

**M/M/1: Single-Server Queue Model**

$M/M/1/\infty/\infty$ represents a single server that has unlimited queue capacity and infinite calling population. The mathematical nature of the exponential distribution, a number of quite simple relationships are able to be derived for several performance measures based on knowing the arrival rate and service rate. The arrival is Poisson process, meaning the statistical distribution of the inter-arrival times still follow the exponential distribution. The distribution of the service time may follow any general statistical distribution, not just exponential. Relationships are still able to be derived for a (limited) number of performance measures if one knows the arrival rate and the mean and variance of the service rate. Models briefly discussed in Dombacher (2009), Lipsky, (2009), Jain, Mohanty, and Bohm (2007), Ng and Soong (2008), and Little and Graves (2008).

**M/M/C: Multiple-Servers Queue**

Multiple (identical)-servers queue situations are frequently encountered in telecommunications or a customer service environment. When modelling these situations care is needed to ensure that it is a multiple servers queue, not a network of single server queues, because results may differ depending on how the queuing model behaves (Dombacher, 2009; Lipsky, 2009; Jain, Mohanty, & Bohm, 2007).

**M/M/\infty: Infinitely Many Servers**

While never exactly encountered in reality, an infinite-servers (e.g., $M/M/\infty$) model is a convenient theoretical model for situations that involve storage or delay, such as parking lots, warehouses and even atomic transitions. In these models there is no queue, such as, instead each arriving customer receives service. When viewed from the outside, the model appears to delay or store each customer for some time (Jain, Mohanty, & Bohm, 2007; Ng & Soong, 2008).

**M/M/c/K/\infty Model: Capacity Constraints**

By customizing the parameters for the load dependent model, capacity constraints may be introduced to a multiserver system $M/M/c/K$. The queue capacity limitation is the only difference between the limited and the unlimited model (Dombacher, 2009; Lipsky, 2009).

**M/M/c/K/P Model: Finite jobs population**

The $M/M/c/K/P$ queuing system is another variant of the $M/M/c$ queuing system. In this system, there is a finite number $P$ (General notation for population is $M$) of potential jobs, while there is room (queue) for $K$ jobs, including the jobs in service ($P \geq K \geq C$). Note that this is in fact a closed system where each of the
P jobs is either inside the system (waiting or being served) or passive outside the system until its next visit to the system. In this case, the arrival rate depends on the number of passive customers so that the arrival process is not a Poisson process. It is assumed that each potential customer returns to the system after an exponentially distributed passive time with rate $l$. Such a system is stable for every positive value of $l$, and service rate $m$. Special case is the M/M/c/c/P loss systems which are often referred to as Engset loss systems. Stallings (2000) and Blanc (2011) have discussed and more detail in Dombacher (2009), Lipsky (2009), Jain, Mohanty, and Bohm (2007), Ng and Soong (2008), Kobayashi and Konheim (1977), and Adan and Resing (2008).

### Scheduling Architecture

When receiving a job submission, the LS will estimate the time for completing executing (ETTC) a job in a site $i$, as mentioned by Ranganathan, K., & Foster (2001), Ranganathan and Foster (2003), and Nguyen and Lim (2007).

$$ETTC_{j,i} = \max \left\{ DT \left( f\left(j,i\right)\right), QT_{i}\right\} + EET_{j,i}$$

(3)

Where $DT$ is the Data Transfer Time for job; $QT$ is the Queuing Time in site; and $EET$ is the Estimate Execution Time for a job.

For Turnaround Time simply we use, Total Completion Time ($TCT$) for a job which is the sum of all these three Times.

$$TCT_{j,i} = QT_{i} + ET_{j,i} + DT \left( f\left(j\right)\right)$$

(4)

According to Little's Law, Equation 4 illustrates some important parameters associated with a queuing model. Items arrive with some average rate (items arriving per second). At any given time, a certain number of items will be waiting in the queue (zero or more); assume the average number of items waiting is $w$, and the mean time that an item must wait is $T_w$. $T_w$ is averaged over all incoming items, including those that do not wait at all. The server handles incoming items with an average service time $T_s$; this is the time interval between the dispatching of an item to the server and the departure of that item from the server. Finally, two parameters apply to the system as a whole. The average number of items resident in the system, including the item being served (if any) and suppose the items waiting (if any), is $r$; and the average time that an item spends in the system, waiting and being served, is $T_r$; we refer to this as the mean residence time. If we assume that the capacity of the queue is infinite, then no items are ever lost from the system; they are just delayed until they can be served. FIFO (First In First Out) is suitable principals to use (Figure 4).

**Assumptions:** According to Little’s Law (Little & Graves, 2008)

$$T_w = W_q = \text{mean waiting time}$$

$$T_s = W_s = \text{mean service (execution) time for each arrival;}$$

$$T_r = W_q + W_s = \text{mean residence time (time spends of an item in system)}$$

Here, $T_w = W_q = QT$, $T_s = ET = W_s$ and $DT = Tt$

(5)

Hence, $T_r = W_q + W_s$

(6)

In our Equation 2, the residence time is

$$T_r = W_q + W_s$$

(7)

Where $Wq$ is the Wait in Queue for a job and $Ws$ is the Wait in Server which is generally Execution time. As described in Figure 5.
Using M/M/c/K/P Model: Finite Jobs Population

If there are $n$ jobs in the queue there are $N - n$ jobs in the source. We assume that jobs wait in the source an exponentially distributed amount of time with average before returning to the queue and that they are independent of each other. If there are $N - n$ jobs in the source then they create a Poisson input process to the queue with rate $\lambda$ (Figure 6).

In Finite Population Model (Ghemawat, Gobioff, & Leung, 2003; Amazon, 2009; Borthakur, 2007; Dean & Ghemawat, 2008).

Putting Equations 4 in Equation 2, the Total Completion Time for a job $TCT$

$$TCT_{j,i} = QT_w + ET_s + DT(f(j)) = W_q + W_s + Tt$$

(8)

To evaluate this technique, M/M/c/K/P Model has used. By using this model, calculation has done to determine all other parameters, which we need for the calculation of $TCT$. The purpose of using this model is, because we need the job size in advance to calculate the total time for transferring.
Here $M$ denotes the total size of the population, means number of jobs. Assuming a system with $C < M$ service units (Dean & Ghemawat, 2004; Ghemawat, Gobioff, & Leung, 2007; Amazon, 2009; Borthakur, 2007; Dean & Ghemawat, 2008), i.e.

$$\mu_n = \begin{cases} n\mu & \text{for } 1 \leq n < c \\ c\mu & \text{for } n \geq c \end{cases}$$

For average queue size $L_q$ and average waiting time in queue $W_q$ calculation, we need probability of ‘0’ entity in the system $P_0$, probability of ‘n’ entities being in the system $P_n$, average number of jobs in the system $L$, and average time spent in the system $L_s$. Using Hickey (2011a, 2011b), Butcher (2011), Williams (2010), Foursquare (http://status.foursquare.com), and IDC (2011) we can calculate as:

$$P_0 = \frac{\sum_{n=0}^{c-1} \left( \frac{M!}{(M-n)!n!} \right) \rho^n}{\sum_{n=c}^{M} \left( \frac{M!}{(M-n)!n!} \right) \frac{n!}{c^{n-c}} \rho^n}$$

$$L = \frac{\sum_{n=0}^{c-1} \left( \frac{M!}{(M-n)!n!} \right) \rho^n}{\sum_{n=c}^{M} \left( \frac{M!}{(M-n)!n!} \right) \frac{n!}{c^{n-c}} \rho^n} P_0$$

$$L_q = \sum_{n=c}^{M} (n-c) p_n = \sum_{n=c}^{M} np_n - c \sum_{n=c}^{M} p_n$$

Using the definition of the expected value, it is now easy to drive the average system size means number of jobs in the system (Hickey, 2011a, 2011b; Butcher, 2011; Williams, 2010; Foursquare, http://status.foursquare.com; IDC, 2011).
By using Little’s Law (Foster, Yong, Rai-cu, & Lu, 2008) and detail in Dean and Ghemawat (2004) and Ghemawat, Gobioff, and Leung (2003) with mean arrival rate \( \lambda \) can give

\[
W_L = \frac{L}{\lambda (M - L)}
\]  

(13)

and

\[
W_q = \frac{L_q}{\lambda (M - L)}
\]  

(14)

For Consuming Time (Execution Time) by Processor

\[
ET_j = L - c + P_0 \sum_{n=0}^{c-1} (c - n) \left( \frac{M!}{(M - n) !n!} \right) \rho^n
\]  

(15)

If \( N_p \) is the Number of Jobs in Processor and \( T_p \) is the Total Time Consumed by processor for those jobs. Then Data Transfer Time according to Ranganathan and Foster (2003) and Nguyen and Lim (2007) will be as

\[
DT(f(j)) = \text{sizef}(k) / BW_{(i,j)}
\]  

(16)

Where \( \text{Size} f(k) \) is the FileSize in bytes and \( BW \) is the available Bandwidth between computing sites. Now Putting Equations 13, 14, and 16 in Equation 8, result will be as

\[
TCT_{ji} = \frac{L_q}{\lambda (M - L)} + L - c + P_0 \sum_{n=0}^{c-1} (c - n) \left( \frac{M!}{(M - n) !n!} \right) \rho^n
\]  

(17)

RESULTS AND DISCUSSION

In order to evaluate the performance of proposed architecture, author has removed the MAX function from equation 1 and 2. And add all three parameters as mentioned in equation 4 or in equation 8. There is great impact on accuracy in calculation of Total Time of Completion for a job, which brings improvement in efficiency (in term of accuracy).

Dealing with large amount of data makes the requirement for efficiency in data access more critical. Users of the world need more copy (replicas) of the Global one cloud. This one cloud should be replicate in many replicas on the base of sub-continents or may be on the base of population (number of user). Each local sub cloud will have same anytime updated copy of each other. Due to the local cloud, data in a wisely manner will offer a faster access to files required by cloud client, hence increase the job execution’s performance. Due to local cloud, reliability and accessibility will increase. In this case if one sub cloud goes down, any client from anywhere in the world can use another sub cloud under the “failover” techniques. Cloud service failure is not acceptable in this architecture. In propose architecture, ROWA is the most suitable technique to combine replication with scheduling.

For evaluating new architecture, by using formulas, first we calculate the Total consume time just in transferring of data, excluding the queue time and system time. As described in Figure 7.

This section compares overall results, and calculates the variance between Total Time (\( TT \)) according to existing models and Total Comple-
tion Time \((TCT)\) according to new proposed model. Here \(TT\) is actual and \(TCT\) is estimated value. The following general formula for error estimation has been used, to compare both time variations.

After General Error and Accuracy Estimation, now we need to compare the change in Error Estimation with various bandwidths. Error is decreasing with increasing population \(M\), it means that new proposed model is more efficient to use for large data transferring as shown in Figure 8 by using \(C=1, \lambda=1, \mu=1, M=2 \sim 50, BW=56Kbps\).

Using \(C=1, \lambda=1, \mu=1, M=2, BW=56Kbps\), the accuracy is 85.500%, it is increasing upto 92.4639% by using \(M=50\), and by taking \(C=1,\)

---

**Figure 7. Transfer time for data**

![Graph showing transfer time for data](image)

**Figure 8. Error estimation with \(C = 1\) and \(BW= 56Kbps\)**

![Graph showing error estimation](image)
\( \lambda = 1, \mu = 1, M=2, \text{BW}=512\text{Kbps} \) as shown in Figure 8. Accuracy is 98.5000\%, by using \( M=50 \), it will increase up to 99.1753\% as mentioned in Table 2. Result shows that with increase in population (M), by using same bandwidth and same number of server, the gap is decreasing between the results of both techniques. Decreasing the gap, means accuracy is increasing by using new technique. When \( M>500 \), stability point (where accuracy is 100\%) can be achieved. Hence new technique is more efficient when we need to transfer large amount of data.

**ACKNOWLEDGMENT**

Appreciation conveyed to Ministry of Higher Education Malaysia for project financing under Exploratory Research Grant Scheme, RDU120608.

**CONCLUSION**

This paper presents scheduling architecture for cloud computing to support efficient data access for the job. Cloud should be a “true service provider” which is the demand of each user. Proposed architecture fulfills this property, because there is no chance to go down all local clouds at a time. This scheduling technique gives importance to each parameter while calculating Total Time of Completion for a job. This research evaluate the result of total completion time for jobs only using single server finite population model using 56–512kbps bandwidth. There is a great impact on accuracy by taking each parameter separately. Further we need to evaluate our new model by using others M/M/C, M/M/Inf, M/M/C/K and M/M/C/*/M queuing models.

In future work, we plan to investigate more realistic scenarios and real user access patterns. Additionally, we want to propose a complete real time model by using CloudSim, with combination of replication and scheduling.

**REFERENCES**


Nawsher Khan is a Doctoral Scholar at the University Malaysia Pahang (UMP) Malaysia. He received his bachelor degree in computer science from University of Peshawar, Pakistan and master degree in computer science from Hazara University, Pakistan. Since 1999, he has worked in various educational institutions. In 2005, he was appointed as CM in NADRA (National Database and Registration Authority) under the Interior Ministry of Pakistan. Currently he is a Research Assistant, in Faculty of Computer System and Software Engineering at University Malaysia Pahang (UMP) Malaysia and his research interest include Cloud Computing, Data Management, Distributed System, Scheduling and Replication. In addition he has published more than 25 articles in various International journals and conference proceedings.

Noraziah Ahmad received PhD in Distributed Database from University Malaysia Terengganu (UMT) in 2007. She has published more than 130 papers in the journals and conference proceedings. Currently, she is the senior lecturer at Faculty of Computer Systems & Software Engineering, University Malaysia Pahang. In addition to serving as international program committee member and reviewers in many conferences, she is currently an editorial board member of the International Journal of Engineering and Technology (IJET), International Journal of Web Application (IJWA) and Journal of Emerging Technologies in Web Intelligence (JETWI); a member of IEEE Computer Society, International Association of Engineers (IAENG), World Academy of Science, Engineering and Technology (WASET), Malaysian National Computer Confederation (MNCC) and Senior member of International Association of Computer Science and Information Technology (IACSIT). She is also an advisory board of The Society of Digital Informatics and Wireless Communications (SDIWC). Her current research interests include distributed database systems, distributed systems, data grid, cloud computing and biofeedback.
Tutut Herawan received his BEd degree in year 2002 and MSc in year 2006 degree in Mathematics from Universitas Ahmad Dahlan and Universitas Gadjah Mada Yogyakarta Indonesia, respectively. Late, he obtained his PhD from Universiti Tun Hussein Onn Malaysia. Currently, he is a Senior Lecturer at Department of Computer Science and leader of Database and Knowledge Management (DBKM) research group at Faculty of Computer Systems and Software Engineering, Universiti Malaysia Pahang (UMP). He published more than 90 research papers in journals and conference proceedings. His research area includes Data Mining and Knowledge Discovery, Rough Set theory, Soft Set theory, and Mathematics Education.

Zakira Inayat received her MS Degree from Institute of management Sciences (IMSciences) Peshawar, Pakistan in 2012, Master in Computer Science from Hazara University, Pakistan in 2006 and bachelor degree from University of Peshawar, Pakistan in 2003. Currently she is the senior lecturer at Department of Computer Science and Information Technology, University of Engineering and Technology, Peshawar, Khyber Pakhtunkhwa, Pakistan. Her research interests include Cloud Computing, Data Management and Distributed System. In addition she has published 5 articles in various International journals and conference proceedings.