


RESEARCH

Open Access



Performance of integrated workload scheduling and pre-fetching in multimedia mobile cloud computing

Kaveh Khorramnejad, Lilatul Ferdouse, Ling Guan and Alagan Anpalagan* 

Abstract

This paper focuses on an integrated workload scheduling and pre-fetching model in a multimedia mobile cloud computing environment to enhance the performance of response time and reduce the cost to process multimedia data. The response time and cost optimization problems are presented along with the computation resources such as virtual machines (VMs) allocation, workload conservation, queueing stability constraints, and to overcome the total response time and cost, a heuristic approach of workload scheduling method is proposed. The integrated workload scheduling at pre-fetcher and cloud are considered to study the effects of various parameters such as VM's processing speed, pre-fetcher's utilization, the user requests arrival rate. The performance analysis results reveal that the cost and transmission speed are directly relevant factors, meaning that, once the rate of data transmission is increasing, the cost is also increasing and vice versa. Hence, the time and cost efficient workload scheduling is essential to satisfy both delay and cost in pre-fetcher enabled multimedia cloud systems.

Keywords: Multimedia mobile cloud computing, Workload scheduling, Pre-fetcher, VM allocation

Introduction

Multimedia communication over the cellular network with minimum latency become critical with the ever increasing number of cellular users along with multimedia data traffic. The evolution of cloud computing technology, data transmission methods have changed. Since cellular devices (e.g., smart phones, tablets etc.) face some limitations such as power, memory and computing power, the cloud providers offer services such as scalable and reconfigurable computing, storage and network services [1, 2]. Moreover, the cloud computing is a technology with growing popularity for many organizations. The main advantages that lead to this popularity are unlimited storage, backup and recovery, cost efficiency, easy maintenance and quick deployment [1]. Even though cloud computing has made our life much easier, there are many challenges to face. Latency along with security is the most important factor to satisfy when it comes to delay-sensitive transactions like

banking, stock market and real time multimedia data transfer.

Another challenging issue is to process multimedia services in cellular network environment [3]. Cellular data transmission was so successful until battery lives could not keep up with the users' need. The cellular users, mostly cell phone users do not have unlimited storage to store all the information they require. To overcome this challenge, mobile cloud computing was introduced [4, 5]. In mobile cloud computing, mobile users utilize cloud servers, known as virtual machines (VMs) to process high computational services such as real time multimedia data processing (e.g., video on demand (VoD), online gaming, etc). Utilizing cloud computing with mobile networks do not need to consider storage capacity and battery life. However, delay became a new challenge in mobile cloud computing. Pre-fetching was one of the techniques that was proposed to reduce the delay in multimedia delay sensitive mobile communication [6]. This technique can reduce the delay significantly if it is well designed and implemented.

*Correspondence: alagan@ee.ryerson.ca

Department of Electrical and Computer Engineering, Ryerson University, 350 Victoria St., Toronto, Canada

This paper investigates the performance of response time and cost in the integrated workload scheduling and pre-fetching model in mobile cloud computing environment to process multimedia data requests. In this paper, we propose a heuristic based workload scheduling method which utilizes both cloud and pre-fetcher to distribute workload to minimize the total response service time and cost.

The remainder of this paper is organized as follows. In the section system model and assumptions, the pre-fetcher enabled multimedia cloud computing model, followed by the description of system model and problem formulation is presented. Response time and cost optimization problem sections present the analytical formulation of time and cost minimization problems along with a combined pre-fetcher in a cloud system. The solution of workload scheduling in multimedia cloud networks is discussed in Heuristic approach of workload scheduling section. Performance analysis section provides the results of our proposed model. Finally, conclusion section concludes the paper and provides some insights of future research in multimedia mobile cloud computing model.

Related work

The topic of cloud computing is undergoing a growing popularity among practitioners, and scientific researchers. The cloud provides services such as software, platform, infrastructure and these services are maintained by cloud providers (e.g., AWS, Google cloud platform, Microsoft azure etc.) [1, 2]. The cloud technologies help service providers to provide faster and more secure computing services to users. The vast computing resources led to the selection of cloud as the integrating stage for the mobile environment. The core of the cloud is data center, consisting of different type of the servers, e.g., master, computing, etc.

Authors in [7, 8] present a three-tier architecture for multimedia data center which comprises three different types of servers; master, computing and transmission, as depicted in Fig 1. In the beginning phase, the master servers distribute the workload to the computing servers and the cloud provider utilizes different types of VM clusters to complete each service task. In the end phase, the transmission server allocates bandwidth resources to transfer the results to the users. One of the challenges considered in [7, 9] is to schedule the workload among the VM clusters in such a way that each service utilizes minimum number of computing resources (e.g., VMs) and simultaneously ensures the quality of experience (QoE) among users.

Authors in [10, 11] allocate VMs at both user level and task level. In [11], authors proposed a task based workload scheduling method where each multimedia service is decomposed into independent tasks. The authors

proposed three types of workload scheduling methods (e.g., serial, parallel, and mixed) where one VM is allocated to one task. However, in [11] the authors did not consider the dependency among tasks, total independent paths and the scheduling weight among paths. The Bayesian based workload scheduling method, proposed in [12] resolves the aforementioned issue and utilizes both task and user level scheduling to minimize the mean service time.

Multimedia processing enforces new challenges to cloud computing, according to attributes such as being delay sensitive, high computation intensity and significant bandwidth demands [13]. For delay sensitive applications, the authors in [8] consider the total round trip time (RTT) (e.g., both service time and transmission time), whereas in [12] only service time is considered. Moreover, to minimize delay, authors in [6, 14] proposed cloudlets model to process multimedia service requests. The cloudlets represent a small scale data center which resides within the proximity to the users and minimize the transmission delay. On the other hand, authors in [15] consider data processing delay in the mobile edge computing model. Similar to the above concepts, we proposed the pre-fetcher enabled multimedia mobile cloud computing model, shown in Fig.1, where the pre-fetcher is located in close proximity to the users and contains high speed VMs and storage servers. The next section presents the pre-fetcher enabled multimedia cloud computing model.

System model and assumptions

Multimedia mobile cloud computing with pre-fetcher

We consider a pre-fetcher enabled multimedia cloud computing model, depicted in Fig. 1 to enhance the multimedia end user communication process. Introduced model consists of three main components: multimedia cloud provider, pre-fetcher and end users. Multi-media cloud provider follows the three-tier architecture, presented in [7, 8, 10, 12]. The multimedia data center has a core which is called master server. Master server receives the portion of the workload sent to computing server. The workload scheduler and load balancer are responsible for distributing the workload over N server clusters for processing. In addition, we consider a pre-fetch updater which is the intelligent core of the transmission server which calculates the probability of the next call on the same request and sends a copy to pre-fetcher. Each server cluster receives a portion of the total workload and processes it with different processing speed until the entire workload is processed. Then pre-fetch updater gradually fills the storage clusters in it by sending a copy of the critical data for next call.

Pre-fetcher is located near to the users, contains storage clusters and limited number of high speed VM clusters compared to the cloud. Pre-fetcher performs two types of operations: caching and pre-fetching. The latter one is

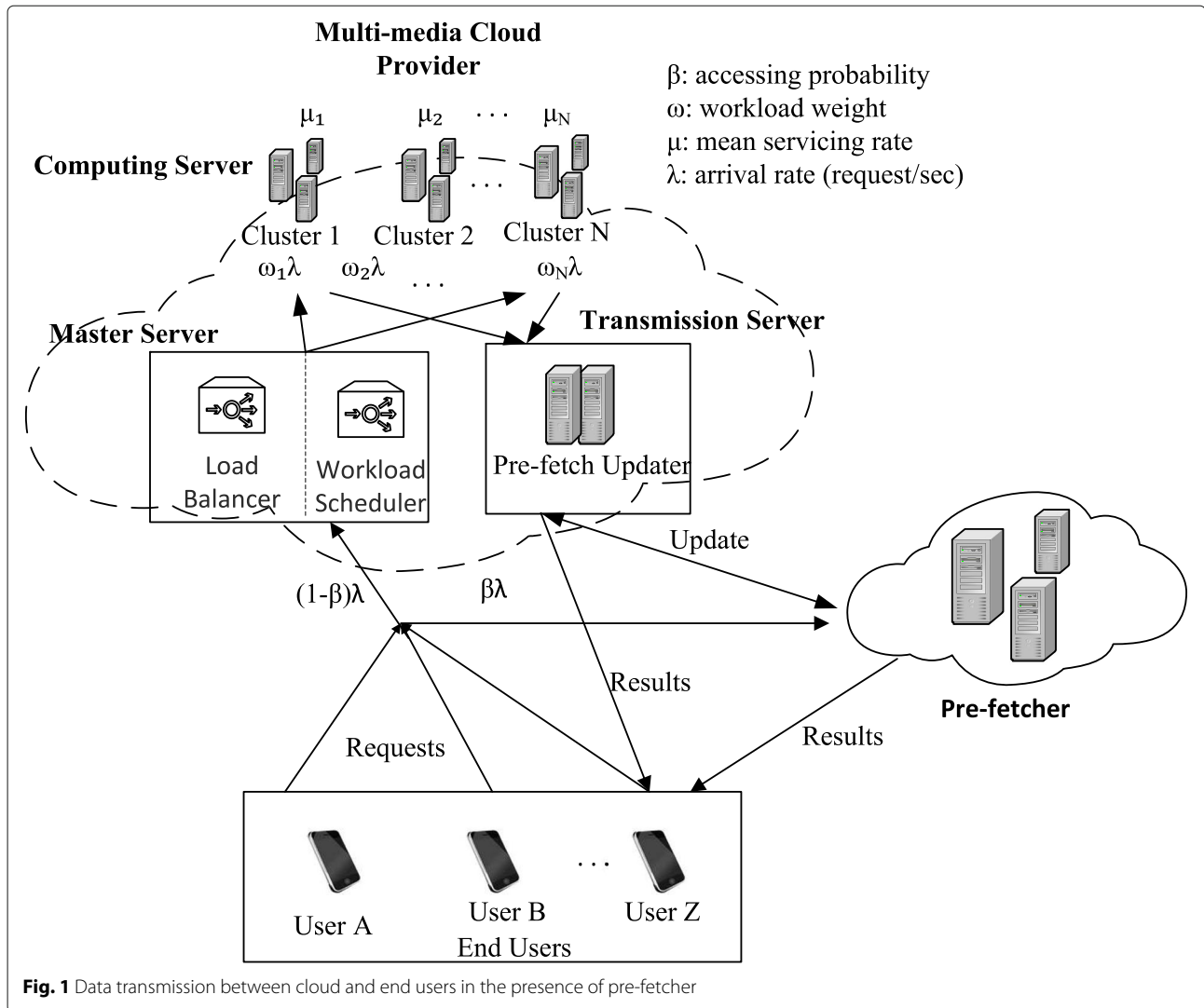


Fig. 1 Data transmission between cloud and end users in the presence of pre-fetcher

the main focus of this paper. For caching, we assume that the storage clusters store data which is more likely to be requested next. Once the memory is full, it will be overridden based on adaptive replacement algorithm (ARA) [16]. For pre-fetching, we develop an integrated model that utilizes workload scheduling and pre-fetching operation both in the pre-fetcher and in the cloud with minimum cost and time.

There are Z end users considered in the model, and the probability of end users reaching pre-fetcher is $\beta\lambda$ versus the probability of reaching multimedia cloud which is $(1 - \beta)\lambda$. β denotes the accessing probability of pre-fetcher and λ denotes the number of user requests arrival per second. All symbols and parameters in the optimization problem are listed in Abbreviation.

Response time optimization problem

We assume that the multimedia cloud data center supports Bayesian method based workload scheduling

[12] where each multimedia service request is represented by directed acyclic graph (DAG) [11] and is decomposed into tasks. Hence, the service requests are modeled by parallel structure of tasks and inter-connected by path. One VM is allocated to one task. Assume that M tasks are scheduled into each path and served by VM cluster i with scheduling weight w_i . Thus, $\sum_{i=1}^N \theta_{ij}$ class- i VMs work together as the class- i virtual cluster with the mean service rate $\sum_{i=1}^N \sum_{j=1}^M \theta_{ij} \mu_{ij}$, and the service time of a request is assumed to be exponentially distributed with an average of $\frac{1}{\sum_{i=1}^N \sum_{j=1}^M \theta_{ij} \mu_{ij}}$. Similar to [10], the service process of each path follows an M/M/1 queuing model. The average response time of path k can be calculated as:

$$T_k = \sum_{i=1}^N \frac{\omega_{k,cloud}}{\sum_{i=1}^N \sum_{j=1}^M \theta_{ij} \mu_{i,cloud} - \omega_{k,cloud} \lambda_{i,cloud}} \quad (1)$$

The total mean response time is considered as the QoS requirement of the system and can be formulated as:

$$T_{QoS} = (1 - \beta) T_{Cloud} + (\beta) T_{PF} \quad (2)$$

where β is the probability (or percentage of time) of using pre-fetcher in time slot t ,

$$\beta = \begin{cases} 1, & \text{if only pre-fetcher is used,} \\ 0, & \text{if only cloud is used,} \\ > 0 \text{ and } < 1 & \text{otherwise.} \end{cases} \quad (3)$$

$T_{Cloud} = \sum_{k=1}^K T_k$ represents the total mean response time in the cloud and T_{PF} represents the total mean response time in the pre-fetcher. The pre-fetcher's mean response time T_{PF} can be formulated as:

$$T_{PF} = \sum_{k=1}^K \frac{\omega_{k,PF}}{\sum_{i=1}^N \sum_{j=1}^M \alpha_{ij} \mu_{i,PF} - \omega_{k,PF} \lambda_{i,PF}}. \quad (4)$$

The response time optimization problem for workload scheduling in cloud system with pre-fetcher can be formulated as:

P1: Minimize : $T_{QoS} = (1 - \beta) T_{Cloud} + \beta T_{PF}$,
 ω, θ, α

Subject to: $\theta_{ij} = \{1, 0\}, \alpha_{ij} = \{1, 0\}, \gamma > 1$,

- C1: $\sum_{j=1}^M \theta_{ij} = 1 \quad \forall i \in N$,
- C2: $\sum_{i=1}^N \theta_{ij} \geq 1 \quad \forall j \in M$,
- C3: $\sum_{j=1}^M \alpha_{ij} = 1 \quad \forall i \in N$,
- C4: $\sum_{i=1}^N \alpha_{ij} \geq 1 \quad \forall j \in M$,
- C5: $\sum_{k=1}^K \omega_{k,Cloud} = 1$
- C6: $\omega_{k,Cloud} \lambda_{i,Cloud} \leq \sum_{i=1}^N \sum_{j=1}^M \theta_{ij} \mu_{i,Cloud}$,
- C7: $\sum_{k=1}^K \omega_{k,PF} = 1$
- C8: $\omega_{k,PF} \lambda_{i,PF} \leq \sum_{i=1}^N \sum_{j=1}^M \alpha_{ij} \mu_{i,PF}$,
- C9: $\mu_{i,PF} = \gamma \mu_{i,Cloud}$.

The objective of **P1** is to minimize the total mean response time of the system. Constraints C1 and C3 illustrate that at time instance t , each VM serves only one task whereas constraints C2 and C4 indicate that one task needs more than one VM. Constraint C5 is the workload conservation constraint for cloud. Constraint C6 is the queuing stability constraint associated with cloud. Constraint C7 is the workload conservation constraint related to pre-fetcher, constraint C8 is pre-fetcher's queuing stability constraint, and C9 shows the processing capacity relation between the VMs in multimedia-cloud and pre-fetcher. Note pre-fetcher is assumed to process the requests faster, γ times faster than the processor in multimedia-cloud.

Cost optimization problem

According to our system model in Fig. 1, there is delay associated with pre-fetcher's functionality denoted by

T_{\S} (sec). The delay T_{\S} was chosen in a way that it is exponentially increasing with γ and β in our model. This delay is a time frame that pre-fetcher requires to react towards the received request. The cost problem for workload scheduling presence of pre-fetcher can be formulated as follows in (5).

$$\begin{aligned} T_{Total} &= T_{QoS} + T_{\S} \\ T_{QoS} &= (1 - \beta) T_{Cloud} + \beta T_{PF} \\ T_{\S} &= \beta g(\gamma) = \beta C \gamma^m \beta^{m'}, C > 0 \end{aligned} \quad (5)$$

The cost optimization problem can be formulated as

P2: Minimize: $T_{Total} = T_{QoS} + T_{\S}$,
 ω, θ, α

Subject to: $\theta_{ij} = \{1, 0\}, \alpha_{ij} = \{1, 0\}, \beta = [0, 1],$
 $\gamma > 1, C > 0$.

- C1 : to C9: ,
- C10: $0 < T_{Cloud} - T_{PF} < (m' + 1) C \gamma^m$.

The objective of **P2** is to minimize the cost of the system. Constraint C10 is the cloud and pre-fetcher's response time difference threshold constraint that ensures that the difference is always a positive number and less than a maximum.

Heuristic approach of workload scheduling

Since the response time (**P1**) and cost (**P2**) problem are known as a convex mixed-integer problem, it is not easy to solve with mathematical and theoretical solutions. Hence, we propose a heuristic (exhaustive search) approach to optimize the response time and cost problems. However, the access probability for pre-fetcher (β) in our cost optimization problem can be analytically found as follows:

$$\begin{aligned} T_{Total} &= T_{QoS} + T_{\S} \\ T_{QoS} &= (1 - \beta) T_{Cloud} + \beta T_{PF} \\ T_{\S} &= \beta (C \gamma^m f(\beta)) \end{aligned} \quad (6)$$

Assume $f(\beta) = \beta^{m'}, C > 0, 0 \leq \beta \leq 1$

$$\begin{aligned} A &= \sum_{k=1}^K \sum_{i=1}^N \frac{\omega_{k,Cloud}}{\sum_{i=1}^N \sum_{j=1}^M \theta_{ij} \mu_{i,Cloud} - \omega_{k,Cloud} \lambda_{i,Cloud}} \\ A' &= \sum_{k=1}^K \sum_{i=1}^N \frac{\omega_{k,PF}}{\sum_{i=1}^N \sum_{j=1}^M \alpha_{ij} \mu_{i,PF} - \omega_{k,PF} \lambda_{i,PF}} \end{aligned}$$

$$\begin{aligned} \text{Then } T_{Total} &= (1 - \beta)A + \beta (A' + C \gamma^m \beta^{m'}), \\ &= (1 - \beta)A + \beta A' + C \gamma^m \beta^{m'+1}, \\ \partial T / \partial \beta &= -A + A' (m' + 1) C \gamma^m \beta^{m'} = 0, \\ \Rightarrow \beta &= \left(\frac{A - A'}{(m' + 1) C \gamma^m} \right)^{1/m'}, \text{ when } A > A' \end{aligned} \quad (7)$$

In our assumption, initially, we consider a function of β to relate the cost problem to the probability of received workload in pre-fetcher, as well as replace the cloud's and the pre-fetcher's objective functions with A and A' to simplify (6).

As it was discussed previously, heuristic approach for workload scheduling is proposed and algorithms are provided for pre-fetcher (**Algorithm 1**) and cloud (**Algorithm 2**). In every algorithm, there are several steps to be followed in order to achieve the optimized solution. The scheduling procedure of heuristic approach is shown in Fig. 2 and works as follows:

Algorithm 1: Workload scheduling in pre-fetcher with access probability β

Input: $\mu_{PF}, \mu_{Cloud}, \lambda_i$, Number of VMs

Output: $\beta, \lambda_{i,Cloud}, T_{PF}$

- 1 Compute $\gamma = \frac{\mu_{PF}}{\mu_{Cloud}}$ which is the cost of using one class- i VM to process one unit request, respectively.
 - 2 Decompose the requests into tasks and compute the maximum number of path [12].
 - 3 **for all the VMs are processed do**
 - 4 Select the smallest γ from the γ set.
 - 5 **if** $\sum_{i=1}^N \mu_{i,PF} > \sum_{i=1}^N \lambda_{i,PF}$ **is satisfied. then**
 - 6 Calculate Bayesian weight of each path ($\omega_{k,PF}$) from [12] **then**
 - 7 Check the constraint $\sum_{k=1}^K \omega_{k,PF} = 1$.
 - 8 Compute $\beta = \left(\frac{A-A'}{(m'+1)C\gamma^m} \right)^{1/m'}$. *Reservetheⁱth class VM and schedule the $\beta\lambda_{i,PF}$ workload among VM.*
 - 9 Compute the remaining arrival requests for Cloud $\lambda_{i,Cloud} = \lambda_i - \beta\lambda_{i,PF}$.
 - 10 Compute time/cost (T_{PF}) using (4).
 - 11 **return** $\beta, \lambda_{i,Cloud}, T_{PF}$.
-

Pre-fetcher: The pre-fetcher executes **Algorithm 1** to distribute the workload among high speed VMs to faster processing. In the integrated model, we assume that VMs in pre-fetcher are γ times faster than Cloud VMs. For workload scheduling, the pre-fetcher receives requests from users. At first, each multimedia service request is decomposed into tasks based on directed acyclic graph (DAG) method [11] and then the maximum number of path and Bayesian weight is estimated [12] for the allocation of VM clusters. In this process, if the pre-fetcher finds the requested results in the storage servers, the results are served to the users. Otherwise, the lowest cost VMs are selected with Bayesian scheduling weight until all the VMs are occupied. During the workload scheduling stage the

Algorithm 2: Workload scheduling at cloud with access probability $(1 - \beta)$

Input: $\lambda_{i,Cloud}, \beta, \mu_{Cloud}, T_{PF}$

Output: Time/Cost: T_{Total}, T_{QoS}

- 1 Sort i^{th} Class VM in accessing order of cost.
 - 2 Decompose all $(1 - \beta) \lambda_{i,Cloud}$ requests to tasks and compute the maximum number of path [12].
 - 3 **for all** $(1 - \beta) \lambda_{i,Cloud}$ **requests are processed do**
 - 4 **if** $\sum_{i=1}^N \mu_{i,Cloud} > \sum_{i=1}^N \lambda_{i,Cloud}$ **is satisfied. then**
 - 5 Compute Bayesian/Optimum weight $\omega_{k,Cloud}$ from [12] **then**
 - 6 Check the constraint $\sum_{k=1}^K \omega_{k,Cloud} = 1$.
 - 7 Reserve the i^{th} class VM and schedule the workload among VM.
 - 8 Compute T_{Total}, T_{QoS} .
 - 9 **return** T_{Total}, T_{QoS} .
-

access probability (β) and the remaining requests ($\lambda_{i,cloud}$) are estimated if any further processing needed for cloud.

Cloud: The cloud executes **Algorithm 2** to distribute the workload among VMs in cloud. The cloud starts the workload scheduling when the requested results are not found in the pre-fetcher and all the pre-fetcher VMs are occupied with $\beta\lambda_{i,PF}$ service requests. The remaining requests $\lambda_{i,cloud}$ are scheduled at cloud with access probability $(1 - \beta)$. Similar to the pre-fetcher workload scheduling, the cloud decomposes each service into tasks based on DAG method, estimates the maximum number of path and schedules the workload among VMs using Bayesian weight.

Performance analysis

All simulations are written and done in Matlab scripts. The results are categorized based on whether the values of γ and β are fixed or changing. γ is the speed indicator in pre-fetcher's processor and β is the distribution probability of received workload in pre-fetcher and accordingly $(1 - \beta)$ is the probability of workload reaching to multimedia cloud in our system model. The simulation variables and their value/range are depicted in Table 1.

Response time performance analysis

In this part of our study, the primary objective is the total response time minimization. Hence, we performed simulations while changing different variables to see how advantageous pre-fetching scheme is towards the goal. The main variables in our model are β, γ and λ . Accordingly, we discuss each scenario individually and analyse the results.

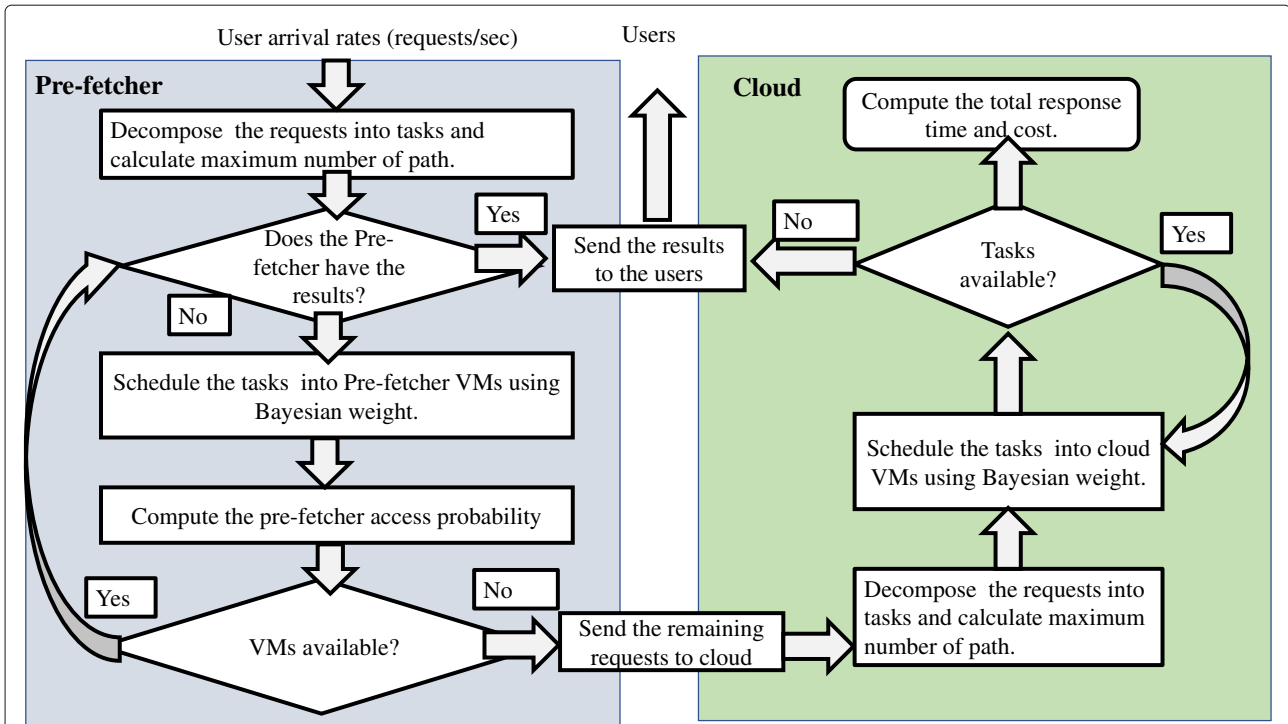


Fig. 2 Heuristic approach of workload scheduling between cloud and end users in the presence of pre-fetcher

Response time study with fixed γ and changing β :

We compare the proposed pre-fetching enabled cloud computing model with the former model [12] where pre-fetcher was not considered ($\beta = 0$). The processing speed in pre-fetcher is assumed to be fixed ($\gamma = 4$) which is four times faster than processing time in multimedia cloud.

As expected, the response time in the results is ascending, while using pre-fetcher benefits us by saving a significant amount of time in the long run. This means, instead of financially investing on data transmission, a portion of the investment can be done to improve and maintain the processor in pre-fetcher.

Table 1 Simulation setup and variable value/range

Variable	Symbol	Value/Range
PF processing speed multiplier	γ	1-50
PF utilization	β	[0,1]
Arrival rate (requests/sec)	λ	1000-9000
Processing delay (sec)	C	1×10^{-4}
PF processing cost factor	m	[0.2,0.25]
PF utilization cost factor	m'	1
Processing speed (sec)	μ	[10,20,30]
Number of clusters	N	3

As presented in Fig. 3, the top curve represents the response time of multimedia cloud based on the arrival rate without considering pre-fetcher. According to Fig. 3, as the number of the requests/sec increases, there is a slight elevation in the curve which means requests will need slightly more time to process and send back the requests to the source. The curve with lower response time represents the solo use of pre-fetcher without considering any load being shared or sent to multimedia cloud. The results show that there is a remarkable performance enhancement in response time when only pre-fetcher is responding to requests. The reason behind the performance enhancement is because the requests already exist in the memory of the pre-fetcher and they just need to be located and sent back, versus in multimedia cloud, there is an extra step required to compute the request. Therefore, their response process in the pre-fetcher is significantly faster. In Fig. 4, metrics and increment units are considered to stay the same similar to the previous figure. The value of γ is also fixed and deemed to be four while β is changing.

In Fig. 4, we increase the workload distribution probability from 0 to 1 for 0.2 units in every increase to show how it would affect the response time as we move toward pushing more workload to pre-fetcher and less to multimedia cloud. As expected, when the percentage of access probability of pre-fetcher increases,

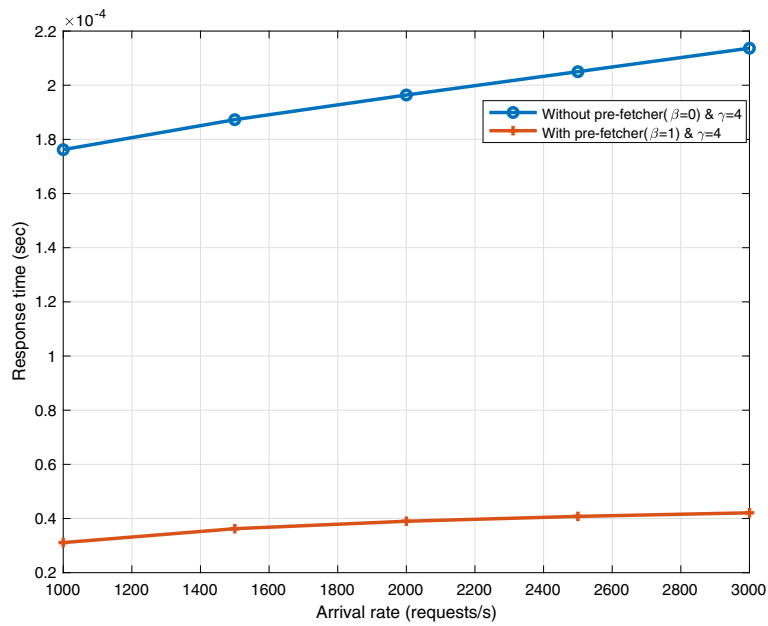


Fig. 3 Response time vs. arrival rate in solo use of the pre-fetcher or multimedia cloud. Without the pre-fetcher ($\beta = 0$ and $\gamma = 4$), With pre-fetcher ($\beta = 1$ and $\gamma = 4$)

the response time drops, and it is minimum in the purple curve when $\beta = 1$. All the results show that, with the assumption of $\gamma = 4$, the processor in pre-fetcher is running four times faster than in multimedia cloud.

Response time study on fixed β and changing γ : It is also shown that Fig. 5, with fixed β regardless of whether we increase the pre-fetcher utilization factor, the increase is still linear and at all various utilization levels there is no difference between using cloud and pre-fetcher.

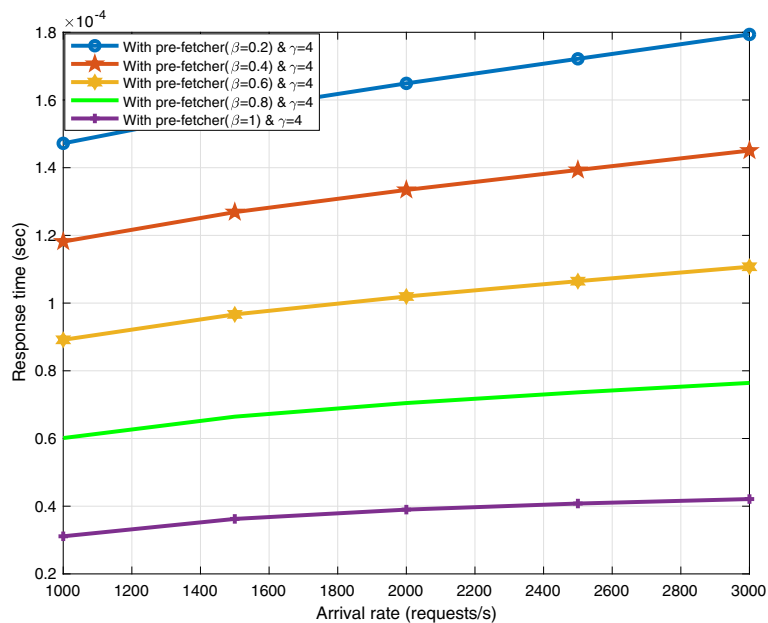


Fig. 4 Response time vs. arrival rate while the workload is shared between multimedia cloud and pre-fetcher at probability distribution of β . With pre-fetcher ($\beta = 0.2$) and $\gamma = 4$, With pre-fetcher ($\beta = 0.4$) and $\gamma = 4$, With pre-fetcher ($\beta = 0.6$) and $\gamma = 4$, With pre-fetcher ($\beta = 0.8$) and $\gamma = 4$, With pre-fetcher ($\beta = 1$) and $\gamma = 4$,

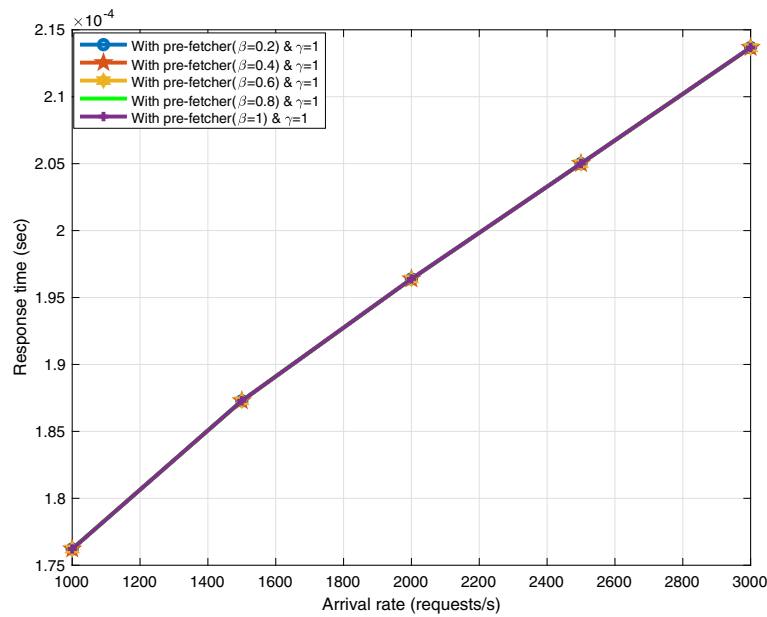


Fig. 5 Response time vs. arrival rate while the workload is shared between multimedia cloud and pre-fetcher at probability distribution of β . With pre-fetcher ($\beta = 0.2$) and $\gamma = 1$, With pre-fetcher ($\beta = 0.4$) and $\gamma = 1$, With pre-fetcher ($\beta = 0.6$) and $\gamma = 1$, With pre-fetcher ($\beta = 0.8$) and $\gamma = 1$, With pre-fetcher ($\beta = 1$) and $\gamma = 1$,

Even though, increasing processing speed in pre-fetcher decreases response time, it also increases the cost (processing delay) associated with processing time. The delay is also considered and optimized using proposed heuristic approach. Results are shown and compared later. Initially, we perform simulations on pre-fetcher while considering the impact of delay and later, after considering the impact of cost (processing delay), we compare the results and show how an integrated model using pre-fetcher and cloud computing is beneficial towards minimization of total response time. Note that in our model we only study the dynamic communication of multimedia cloud, pre-fetcher and end users. We will also find the optimal β for different situations, since it changes based on circumstances from time to time.

In this part, we study the response time behaviour of the arrival rate $\lambda = [1000, 3000, 500]$ and $\gamma = [1, 4, 1]$ at the probability distribution of $\beta = 0.6$. As presented in Fig. 6, we observe the speed enhancement while speed multiplier γ is increasing as expected, the shortest response time belongs to the bottom curve with $\gamma = 4$. Another noticeable result in Fig. 6 is that the response time gap between each curve is significantly decreasing as a result of speed enhancement. Here all the results and the performance analysis are on using pre-fetcher without considering the delay associated with its processor.

Response time study while β is changing: Figure 7 could be one of the most interesting comparisons. In this

figure we have total response time (sec) on Y axis versus workload distribution probability (β) changing on X axis from 0.2 to 1 with 0.1 increment unit. As can be interpreted, when the response time increases and the load is pushed toward the pre-fetcher, only the top line is constant value at $\gamma = 1$, since it is running at the same speed as the cloud and the other curves are all descending. Descending curves show that the response time decreases as the processor’s processing time increases and it reaches its minimum on the bottom curve.

As the workload is leaning more towards pre-fetcher, the processing speed is highlighted more than before. As we can observe in Fig. 7, the bottom curve has the minimum total response time, and the total response time is minimized when entire workload is sent to pre-fetcher. Accordingly, a faster processor is needed which means higher cost, and we will study that in the next part.

Performance analysis of cost

In this section, we investigate the cost improvements based on the simulation results. We will change the values of β , γ and λ to observe the changes in total cost and total response time.

Total cost improvement while PF utilization is $\beta = 0.6$ and λ is increasing: In Fig. 8, we observe the simulation result when only 0.6 ($\beta = 60\%$) of the total workload is sent to pre-fetcher and arrival rate is increasing. γ is also considered to be increasing from 0 to 50. As expected, the

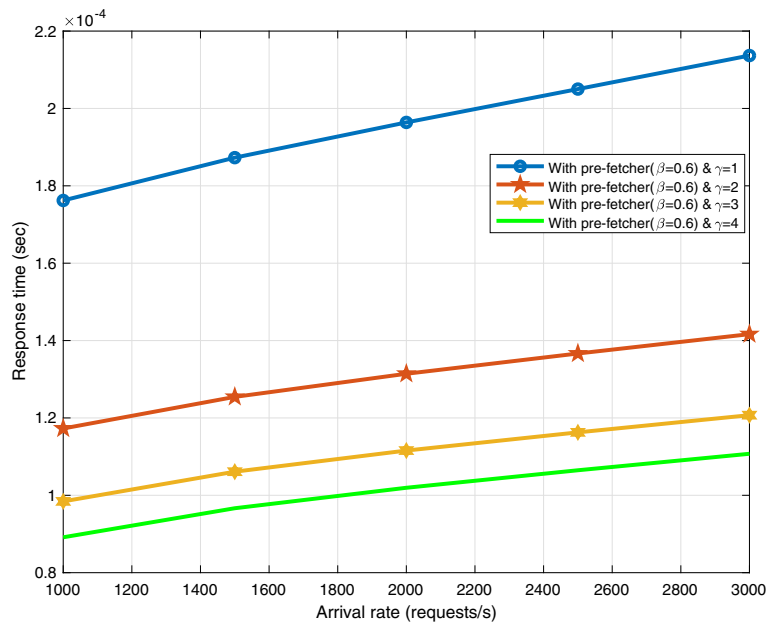


Fig. 6 Workload response time vs. arrival rate while the workload is shared between multimedia cloud and pre-fetcher at probability distribution of $\beta = 0.6$. With pre-fetcher ($\beta = 0.6$) and $\gamma = 1$, With pre-fetcher ($\beta = 0.6$) and $\gamma = 2$, With pre-fetcher ($\beta = 0.6$) and $\gamma = 3$, With pre-fetcher ($\beta = 0.6$) and $\gamma = 4$,

total cost is increasing while the number of requests/sec increases. This cost is associated with the delay that occurs when pre-fetcher is processing the requests.

As presented, the initial γ is set to be 1 in all curves, which means processor of the multimedia cloud and pre-fetcher are both running at the same speed.

Total cost improvement while $\lambda = 3000$ and PF utilization is increasing: As presented in Fig. 9, in this simulation arrival rate is set at 3000 while the use of pre-fetcher is increasing. Total cost is linear and constant when there is no pre-fetcher. While pre-fetcher is operating, initially $\beta = 1$ has the minimum total cost and,

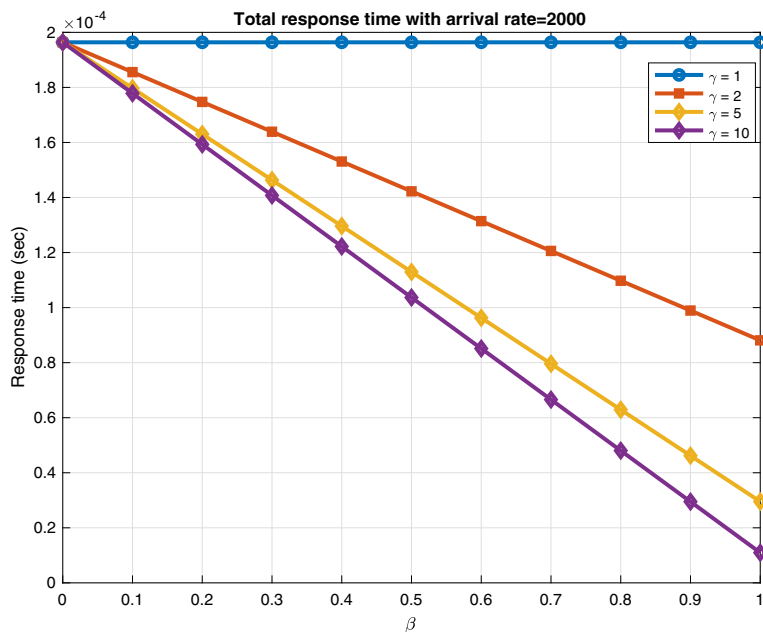


Fig. 7 Total response time vs. probability distribution while pre-fetcher's processor speed multiplier is changing. $\gamma = 1, \gamma = 2, \gamma = 3, \gamma = 4$,

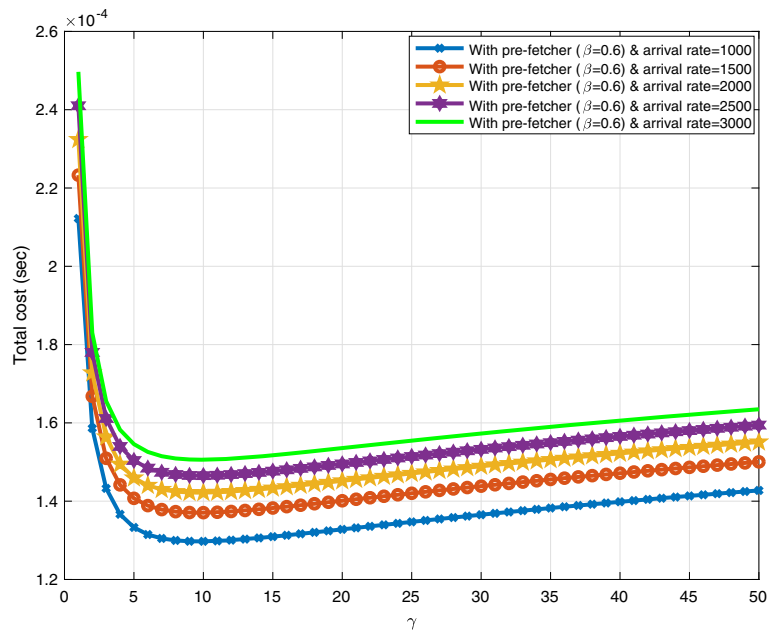


Fig. 8 Impact of increasing arrival rate and γ on Total cost when $\beta = 0.6$. With pre-fetcher ($\beta = 0.6$) and arrival rate=1000, With pre-fetcher ($\beta = 0.6$) and arrival rate=1500, With pre-fetcher ($\beta = 0.6$) and arrival rate=2000, With pre-fetcher ($\beta = 0.6$) and arrival rate=2500, With pre-fetcher ($\beta = 0.6$) and arrival rate=3000,

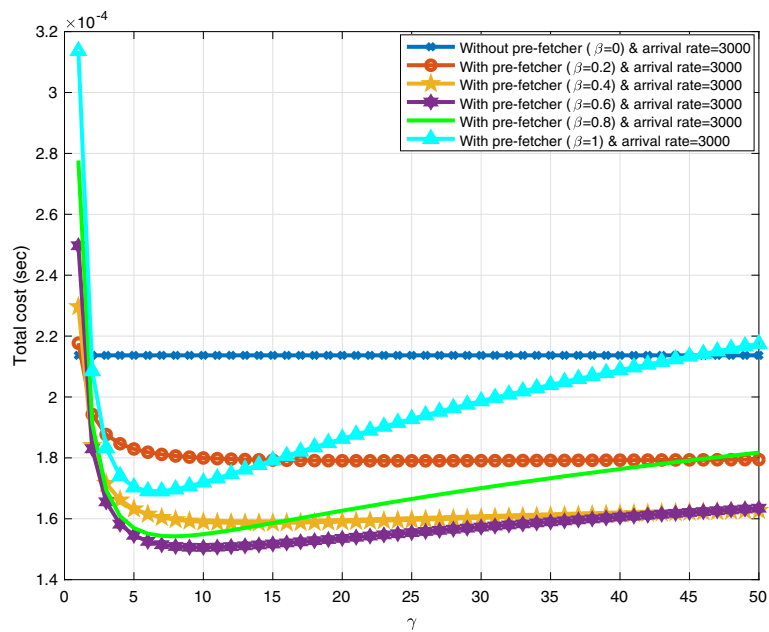


Fig. 9 Impact of increasing β and γ when $\lambda = 3000$. Without pre-fetcher ($\beta = 0$) and arrival rate=3000, With pre-fetcher ($\beta = 0.2$) and arrival rate=3000, With pre-fetcher ($\beta = 0.4$) and arrival rate=3000, With pre-fetcher ($\beta = 0.6$) and arrival rate=3000, With pre-fetcher ($\beta = 0.8$) and arrival rate=3000, With pre-fetcher ($\beta = 1$) and arrival rate=3000,

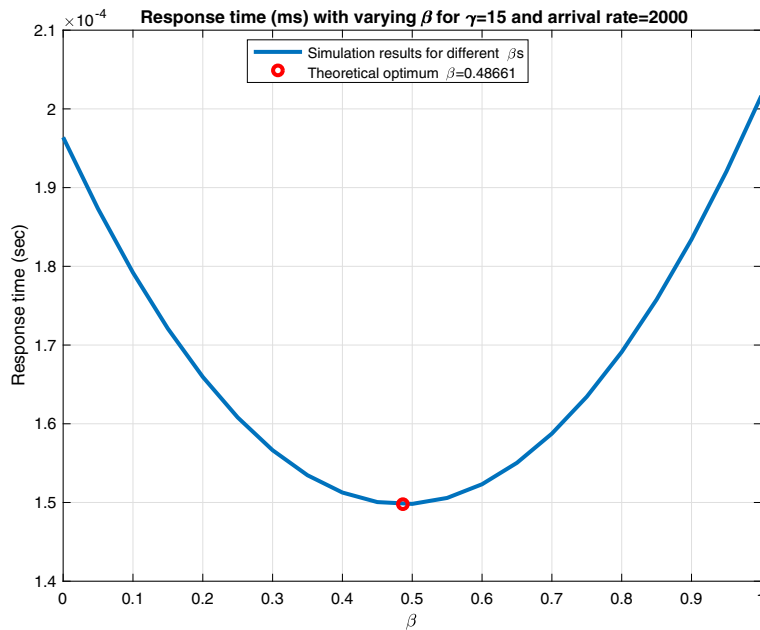


Fig. 10 β vs. Response time (sec) with for $\gamma = 15$, and arrival rate=2000. Simulation results for different β , Theoretical optimum $\beta = 0.48661$,

as γ is increasing $\beta = 0.6$ takes its places and becomes the overall optimum β for this particular instance.

The value of optimal β varies as we change the simulation criteria and change the arrival rate or pre-fetcher’s processing speed. For example, if we take $\gamma = 50$, the optimum value for β is 0.4 and similarly optimal β changes its place based on the condition applied in the simulation.

In other words, the percentage of using pre-fetcher varies from time to time, and it is not always best to allocate the entire workload to pre-fetcher even though it operates faster than cloud.

Response time Vs. β while PF is processing at fixed speed $\gamma = 15$: One of our goals is to find the optimal

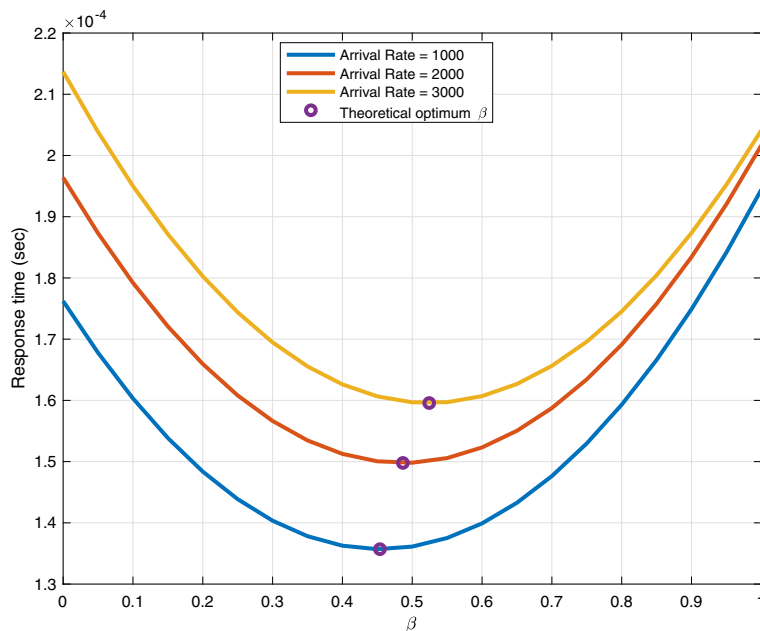


Fig. 11 Comparison of optimum β values while $\gamma = 15$ and λ is changing. Arrival rate=1000, Arrival rate=2000, Arrival rate=3000, Theoretical optimum β ,

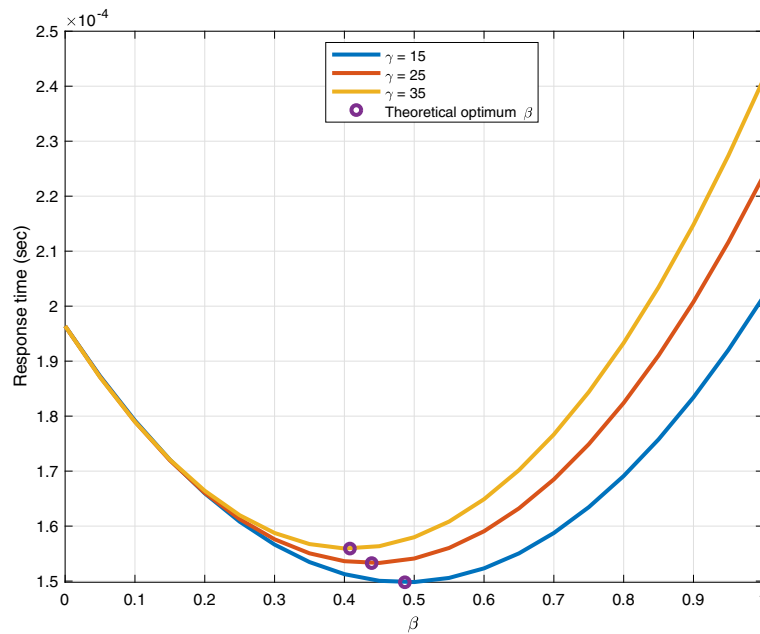


Fig. 12 Comparison of optimum β values while γ is changing and $\lambda = 2000$. $\gamma = 15$, $\gamma = 25$, $\gamma = 35$, Theoretical optimum β

response time for the amount of workload sent to pre-fetcher. In order to study that, we performed a simulation where arrival rate is set to be 2000 requests/sec and $\gamma = 15$. Values of m are considered to be 0.25 and m' is considered as 1.

Based on the results, response time is optimum when ($\beta = 0.48$) almost half of the workload is sent to

pre-fetcher when $\gamma = 15$. The theoretical optimum β is also calculated and marked on Fig. 10 to show that both simulation and theoretical results are aligned at the same point. Starting from $\beta = 0$, response time (sec) is considerably high, and it decreases slowly while the use of pre-fetcher increase up to 0.5, and after that, it starts rising again until it hits $\beta = 1$. It is understandable that if

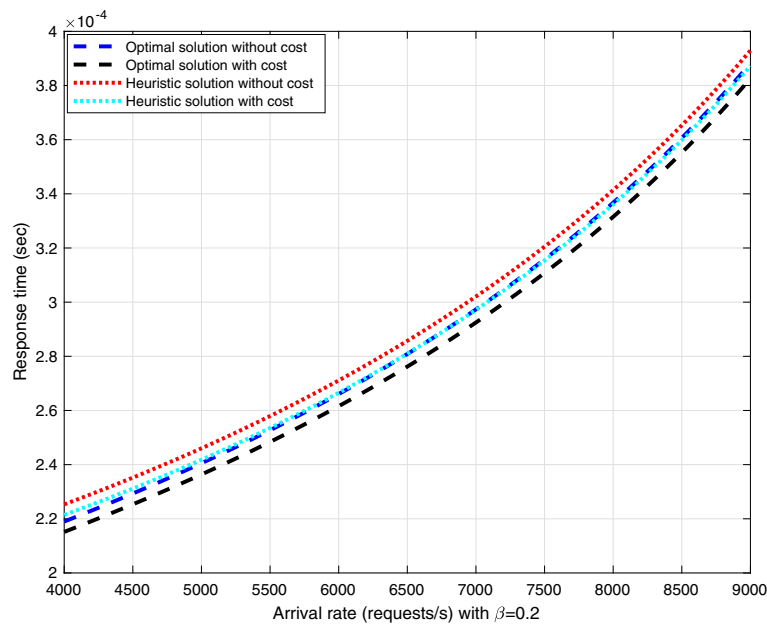


Fig. 13 Cost vs. time optimization. Optimal solution without cost, Optimal solution with cost, Heuristic solution without cost, Heuristic solution with cost

only pre-fetcher is used, it is not the optimum condition, since response time and accordingly costs are increasing.

In Figs. 11 and 12, we investigate the impact of changing γ and λ , on optimum β . As presented, in both figures theoretically calculated optimum β is marked as well. As we can observe from the trends as increase in the arrival rate of the response time slightly increases, the β optimum point moves to the right. In other words, if the amount of the requests/sec is increasing, it is more beneficial to push the load toward pre-fetcher.

As shown in Fig. 11, all the curves for response time are slowly decreasing to the optimal point, and then it increases. The gap between all curves is also shrinking as we move away from the optimal point using (6) and we allocate more load toward pre-fetcher to handle. The optimum point of β aligns with the theoretical value β in all the curves. The response time gap between curves is also maximized at the optimal point and minimized at $\beta = 1$.

In Fig. 12, we study the changing trend of the curves according to the processing speed, while arrival rate is fixed at $\lambda = 2000$. Unlike the previous figure, in this figure, the gap between curves is slowly increasing as we push more load toward pre-fetcher.

As we push more load to pre-fetcher, initial response time decreases to the optimal point and after that, increases to maximum where all the workload is allocated to pre-fetcher. All the curves start basically from the same starting point and initially, their response time decrease is linear to $\beta = 0.5$. Using pre-fetcher is most advantageous at pre-fetcher's utilization optimum point. Similar to the previous figure, the optimum point of β aligns with theoretical value β in all the curves. In both Figs. 11 and 12, we can observe the impact of varying γ and λ when $\beta = [0, 1]$.

Optimal and heuristic optimization performance analysis

Figure 13 compares optimal and heuristic optimization approaches with and without cost factor. We performed both optimal and heuristic approaches on the total response time. In this simulation, pre-fetcher processing speed, γ is at 3, pre-fetcher utilization factor, β , is set at 0.6 while arrival rate changes from 4000 requests/sec to 9000 requests/sec. The heuristic scheme has longer response time than the optimal scheme with a small difference. The optimal scheme weights vary with the change of λ , while weights in the heuristic scheme are constant since it is normalized by the service rate.

Conclusion and future works

Our previous work was done with the primary objective in convex workload scheduling problem for cloud based multimedia applications. In this paper, we propose path selection algorithm to reduce response time by adjusting weights among the paths of tasks both in pre-fetcher and cloud. The considered problem was more

complex compared to previous similar work, and hence the heuristic algorithm was considered to be the solution. In this study, we used a pre-fetcher that adopts and learns new frequent requests to cache and store them prior to the actual request. Even though our goal was to design and develop an intelligent pre-fetcher, there are times that request does not exist within the storage of the pre-fetcher (miss-ratio). The multimedia cloud communication method is becoming one of the major focuses of researchers nowadays, yet intelligent pre-fetching and caching learning schemes can be improved and reducing miss-ratio of intelligent caching schemes would be an interesting topic to work on in future studies.

Abbreviations

ARA: Adaptive Replacement Algorithm; DAG: Directed Acyclic Graph; MCC: Mobile Cloud Computing; PF: Pre-fetcher; RTT: Round Trip Time; RT: Response Time; VM: Virtual Machine; M : number of given tasks; N : number of VM clusters; K : maximum number of paths; i : VM indexing indicator; j : task indexing indicator; k : path indexing indicator; β : accessing probability associated with pre-fetcher; λ : arrival rate (requests/sec); μ : mean servicing rate, μ_i represents the mean servicing rate for VM i ; γ : processor's speed multiplier; α : allocation index, α_{ij} represents the allocation index between the VM i and the task j for pre-fetcher; θ : allocation index, θ_{ij} represents the allocation index between the VM i and the task j for cloud; ω : workload weight, ω_i is the workload weight associate with i^{th} VM cluster; $g(\gamma)$: cost function(delay) associated with pre-fetcher; C : constant delay value in cost function; m : pre-fetcher processor's speed increment indicator in the cost function; m' : pre-fetcher usability increment indicator in the cost function

Acknowledgements

The authors thank Dr. X. Nan for his initial contributions.

Authors' contributions

The work presented in this paper is based on KK MASc research and thesis, which was supervised by Prof. Anpalagan and Prof. Guan. LF is a PhD candidate and she contributed to this work with the system model and problem formulation, and writing a part of this paper. All the authors read and approved the final manuscript.

Authors' information

K. Khorramnejad was a MASc student (2015-17) and L. Ferdouse is a PhD candidate (2016-) in Ryerson University. L. Guan and A. Anpalagan are professors in the Department of Electrical and Computer Engineering, Ryerson University, Toronto, Canada.

Competing interests:

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Received: 20 April 2018 Accepted: 6 July 2018

Published online: 20 July 2018

References

1. Amazon Elastic Compute Cloud. <https://aws.amazon.com/ec2/>. Online; accessed 1-July-2017
2. Bartolini C, El Kateb D, Le Traon Y, Hagen D (2018) Cloud providers viability. *Electronic Markets*. <https://doi.org/10.1007/s12525-018-0284-7>
3. Harnal S, Chauhan RK (2016) Multimedia support from cloud computing: A review. In: 2016 International Conference on Microelectronics, Computing and Communications (MicroCom), pp 1-6. <https://doi.org/10.1109/MicroCom.2016.7522440>

4. Zhu W, Luo C, Wang J, Li S (2011) Multimedia cloud computing. *IEEE Signal Processing Magazine* 28(3):59–69. <https://doi.org/10.1109/MSP.2011.940269>
5. Fernando N, Loke W, WennyRahayu S (2013) Data processing delay optimization in mobile edge computing 29(1):84–106. <https://doi.org/10.1016/j.future.2012.05.023>
6. He Y, Yu FR, Zhao N, Leung VCM, Yin H (2017) Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach. *IEEE Communications Magazine* 55(12):31–37. <https://doi.org/10.1109/MCOM.2017.1700246>
7. Nan X, He Y, Guan L (2012) Optimal allocation of virtual machines for cloud-based multimedia applications. *IEEE 14th International Workshop on Multimedia Signal Processing (MMSp)*:175–180
8. Nan X, He Y, Guan L (2014) Towards optimal resource allocation for differentiated multimedia services in cloud computing environment. *International Conference on Acoustics, IEEE, Speech and Signal Processing (ICASSP)*:684–688
9. Gong W, Chen Z, Yan J, Qianjun S (2014) An optimal VM resource allocation for near-client-datacenter for multimedia cloud. *IEEE International Conference on Ubiquitous and Future Networks (ICUFN)*:149–254
10. Nan X, He Y, Guan L (2013) Optimization of workload scheduling for multimedia cloud computing. *IEEE International Symposium on Circuits and Systems (ISCAS)*:2872–2875
11. Nan X, He Y, Guan L (2013) Optimal task-level scheduling for cloud based multimedia applications. *International Conference on Acoustics, IEEE, Speech and Signal Processing (ICASSP)*:3771–3775
12. Ferdouse L, Li M, Guan L, Anpalagan A (2016) Bayesian workload scheduling in multimedia cloud networks:83–88. <https://doi.org/10.1109/CAMAD.2016.7790335>
13. Zhu W, Luo C, Wang J, Li S (2011) Multimedia cloud computing. *IEEE Signal Processing Magazine* 23(3):59–69
14. Zhijun H, Grande RED, Boukerche A (2017) Towards efficient data access in mobile cloud computing using pre-fetching and caching. In: 2017 IEEE International Conference on Communications (ICC). pp 1–6. <https://doi.org/10.1109/ICC.2017.7997078>
15. Guangshun Li, JW Jiping Wang, Song J (2018) Data processing delay optimization in mobile edge computing:83–88. <https://doi.org/10.1155/2018/6897523>
16. Megiddo N, Modha DS (2004) Outperforming lru with an adaptive replacement cache algorithm. *Computer* 37(4):58–65

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► [springeropen.com](https://www.springeropen.com)
