Rate Optimal Virtual Machine Placement in Cloud Computing

Kamran Zamanifar*, Nader Nasri** and Mohammad-Hossein Nadimi-Shahraki***
*Islamic Azad University, Najafabad Branch, zamanifar@eng.ui.ac.ir
**Islamic Azad University, Najafabad Branch, nadersa@iaun.ac.ir
***Islamic Azad University, Najafabad Branch, nadimi@iaun.ac.ir

Abstract—Infrastructure-as-a-service cloud provides a suitable environment where data-intensive user applications and their required data can be hosted by various networked computing and storage servers. In the cloud environment, users can package their required resources into virtual machines (VMs) and submit their VM requests to the cloud. The performance of user applications strongly depends on the data transfer delay, i.e., the time it takes to transfer the data files required by applications to the corresponding VMs. This delay is a function of the size of data files, the location of VMs that run the user applications as well as the allocation of data rates to VMs. In this paper, we propose a novel rate optimal VM placement (ROVMP) algorithm which not only determines the optimal placement of VMs but also assigns optimal data rates to VMs such that the overall data transfer delay is minimized. Using simulation results we show that the proposed algorithm can significantly reduce the data transfer delay for VMs compared to placement algorithms previously proposed in the literature.

Keywords: Virtual Machine Placement, Rate Allocation, Linear Mix-integer Programming, Cloud Computing.

1. INTRODUCTION

Cloud computing is considered a major step in the distributed computing paradigm. Cloud computing has enabled a wide range of user applications by providing cloud users with a pool of computing and storage resources. In such an environment, user applications are placed on physical machines, called computing centers, where they can use the local computation and memory resources to execute their tasks. In addition, the data required by applications are stored in distributed storage centers across the cloud and are accessible to cloud applications through a network.

Furthermore, virtualization technology has enabled the abstraction of the services provided by the cloud and helps to hide the details of lower layers of the cloud structure (such as the hardware and operating system) from the user. Thereby, virtualization allows the cloud users to package their required resources into virtual machines (VMs) and submit their VM requests to the cloud. These requests are then processed by a placement algorithm which is responsible for determining suitable computing centers where VMs can be executed on.

As virtualization is a core issue in cloud computing, the problem of virtual machine placement has attracted considerable attention recently. In particular, [1] proposes a placement algorithm that determines the placement of VMs such that the energy consumption in computing and storage centers is minimized. [2] employs a particle swarming algorithm to find a near-optimal placement for VMs by taking into account the task completion time of VMs, the cost of VMs, as well as the cost of communications between resources. In [3], a broker-based architecture has been proposed where the broker implements a placement algorithm that dynamically determines the placement of VMs by taking into account average prices as well as the price trend of the computing resources provided by different cloud providers.

Most previous results on VM placement in the literature do not consider the location of the data files required by VMs. However, the data file locations can strongly affect the data transfer delay, i.e., the time it takes to transfer the required data files to VMs. The data transfer delay can substantially increase the time required by the VMs to complete their tasks. This is especially important during the cloud initialization period, where large data files need to be transferred to VMs. These files include, e.g., executable and image files that have to be transferred to VMs before they can start their operation. This problem is exacerbated for a data-intensive application, for which the requested data might be spread in a number of vastly distributed data centers.

In [4], a placement algorithm has been proposed that places VMs, one at a time, such that the data transfer delay is minimized. The method of [4], however, assumes that only a single VM can request a file transfer at a given time, and therefore does not address the problem of optimal rate allocation. However, according to Amazon EC2, one of the most popular cloud computing providers, unlike other resources, data rates are not dedicated and have to be shared among VMs [5]. As a result, it is crucial to devise a placement strategy that carefully takes into account the problem of rate allocation among VMs at the time of placement.

For the purpose of data rate sharing among VMs, conventional transmission control protocol (TCP) algorithms such as TCP Tahoe, Reno, and Vegas [6] can be employed. These algorithms, however, do not take into account the size of data files for rate allocation, and therefore are not generally successful in effectively reducing the data transfer delay. To tackle this problem, in this paper, we propose a novel Rate Optimal VM Placement (ROVMP) algorithm which jointly optimizes
the placement of VMs as well as their allocated data rates. In particular, we formulate a ROVMP optimization problem and show that it can be transformed into a linear mixed-integer programming problem. We then employ a branch-and-bound technique [7] to efficiently solve the ROVMP optimization problem. Using simulation results we show that the proposed algorithm can significantly reduce the data transfer delay for VMs compared to placement algorithms previously proposed in the literature. The proposed ROVMP algorithm can therefore greatly facilitate the adoption of cloud computing by users, especially users with data-intensive applications.

The rest of this paper is organized as follows. In Section 2, we present the system model for the considered cloud computing environment. In Section 3, we present the problem formulation and describe our proposed ROVMP algorithm. We then present the numerical and simulation results in Section 4. Finally, conclusions are drawn in Section 5.

2. SYSTEM MODEL

The considered cloud computing environment is shown in Fig. 1 and comprises several cloud users, computing centers, and storage centers. Computing centers provide a pool of computing power, memory and resources where cloud users can run their applications. Furthermore, storage centers are capable of storing data files required by user applications. Cloud users package their required computing, memory, and disk resources into VMs. The VM requests are then sent to a centralized entity called the cloud broker, where they are stored in a VM repository. The cloud broker is responsible on behalf of cloud users to ensure that the requested VMs are placed on appropriate computing centers. For this purpose, the cloud broker implements a placement algorithm to properly place the VMs on the computing centers. The broker executes the placement algorithm every T seconds, where T is the time interval between consecutive placements. This time interval is adjusted by the broker based on the workload.

At the time of each placement, a set of VMs are already running on the computing centers from previous placements. Furthermore, a set of VMs are stored in the VM repository and are ready for placement. Let \( V \) and \( \hat{V} \) denote these sets, respectively. In addition, let \( V' \triangleq \hat{V} \cup V \), i.e., \( V \) is the set of the VMs that are running or ready to be placed. The computing power, memory, and disk resources that VM \( i \in V \) requires for its operation are denoted by \( R_c(i) \), \( R_m(i) \), and \( R_d(i) \), respectively. Let \( P \) and \( S \) denote the set of computing centers and storage centers, respectively. The maximum computing power, memory, and disk resources provided by computing center \( j \in P \) are denoted by \( \lambda_c(j) \), \( \lambda_m(j) \), and \( \lambda_d(j) \), respectively. In addition, \( p_c(j) \), \( p_m(j) \), and \( p_d(j) \) represent the hourly usage cost of computing power, main memory and disk resources for computing center \( i \), respectively. As a result, the hourly cost for running VM \( i \) in computing center \( j \) is given by

\[
P_r(i, j) = p_c(j)R_c(i) + p_m(j)R_m(i) + p_d(j)R_d(i) \quad (1)
\]

The cloud users also report the maximum acceptable cost for one-hour usage of virtual machines. Let \( \nu_{r, \max} \) denote the maximum acceptable cost for one-hour usage of VM \( i \). Therefore, VM \( i \) has to be placed on a computing center where its hourly cost does not exceed \( \nu_{r, \max} \).

VMs require access to data files stored in distributed storage centers across the cloud. In particular, VM \( i \) requires access to the set of data files \( s^i_k \) with file sizes \( s^i_k \), which are located on storage centers \( k \in S \) and have to be transferred to VM \( i \) before it can start its operation. To enable the data transfer between VMs and storage centers, as shown in Fig. 1, the computing and storage centers are connected to a central wide area network (WAN) using separate access links. \( C_c(j) \) and \( C_s(k) \) denote the capacities of access links that connect computing center \( j \) and storage center \( k \) to the WAN, respectively. We assume that the bandwidth provided by the WAN is sufficiently larger than the capacities of the access links. As a result, the WAN does not impose any constraints on data transfers. Furthermore, we assume that the VMs report their required data files to the broker in advance, and therefore the broker has knowledge about the data files \( f^i_k \), \( k \in S \), and their respective locations for all VMs at the time of placement.

3. ROVMP ALGORITHM

As mentioned in the previous section, the broker implements a placement algorithm to ensure that VMs are placed on appropriate computing centers. In this section, we present an ROVMP algorithm for the broker which
jointly optimizes the placement of VMs and their allocated data rates. In particular, we first present an ROVMP optimization problem and describe how it can be converted to a linear mixed-integer programming problem. We then employ a branch-and-bound technique to efficiently find the optimal solution to the ROVMP optimization problem.

To formulate the placement optimization problem we first define the following variables. Let $X'_j$ denote a binary placement variable which determines the placement of VM $i$. In particular, $X'_j = 1$ when VM $i$ is placed on computing center $j$, and $X'_j = 0$ otherwise. Furthermore, let $V_i$ denote the set of VMs that have requested access to storage center $k$, while $S_j$ denotes the set of storage centers that VM $i$ has requested access to. In addition, $r'_k$ denotes the data rate allocated to VM $i$ for communication with storage center $k$. Consequently, $s'_k/r'_k$ is the time required to transfer the data files requested by VM $i$ from storage center $k$.

The ROVMP optimization problem can now be mathematically cast as

$$\text{Minimize } \sum_{i \in V} \sum_{k \in S_j} (s'_k/r'_k) (2a)$$

subject to:

$$\sum_{j \in V} X'_j \sum_{k \in S_j} r'_k \leq C_r(j), \forall j \in P \text{ (2b)}$$

$$\sum_{i \in V} r'_k \leq C_r(k), \forall k \in S \text{ (2c)}$$

$$\sum_{i \in V} R_m(i)X'_j \leq \lambda_m(j), \forall j \in P \text{ (2d)}$$

$$\sum_{i \in V} R_m(i)X'_j \leq \lambda_m(j), \forall j \in P \text{ (2e)}$$

$$\sum_{i \in P} P_m(i,j)X'_j \leq P_{r',\text{max}}, \forall i \in V \text{ (2f)}$$

$$\sum_{i \in P} P_m(i,j)X'_j = 1, \forall i \in V \text{ (2g)}$$

$$X'_j \in \{0,1\}, \forall i \in V, \forall j \in P \text{ (2h)}$$

where $X'_j = 1, i \in \hat{V}, j \in P$, and $r'_k, i \in \hat{V}, k \in S$ are the optimization variables. We note that placement variables $X'_j, i \in \hat{V}$, are fixed from previous placements, and therefore are not included in the optimization. In the above optimization problem, the objective function (2a) aims at minimizing the maximum data transfer time for all VMs, i.e., we have adopted a min-max fair objective function. Constraint (2b) states that the summation of data rates allocated to VMs placed on computing center $j \in P$ is less than the capacity of the access link which connects this computing center to WAN. Similarly, constraint (2c) ensures that the summation of allocated rates to VMs for transmitting files located on storage center $k \in S$, is less than the capacity of the link connecting this storage center to WAN. Constraints (2d), (2e), and (2f) ensure that computing power, memory, and disk resources allocated to VMs do not exceed the maximum amount offered by computing centers.

Constraint (2g) guarantees that the cost of VM $i$ does not exceed the maximum acceptable cost $P_{r',\text{max}}$ for this virtual machine. Finally, constraint (2h) states that each VM can be placed only on a single computing center, while (2i) constraint states that $X'_j$ is a binary variable.

The optimization problem in (2) is a non-linear mixed-integer problem as the objective function (2a) and constraint (2b) are non-linear. To obtain a linear objective function, we introduce an auxiliary scalar variable $t$, which is a lower bound for the inverse of data transfer delays $s'_k/r'_k$, i.e.,

$$t \leq \min_{i \in V, k \in S_j} \left(\frac{r'_k}{s'_k}\right) \Rightarrow t \leq \left(\frac{r'_k}{s'_k}\right) \forall i \in V, k \in S_j \text{ (3)}$$

Furthermore, the constraint (2b) involves the product of $X'_j$ and $r'_k$, and therefore is a non-linear constraint.

We apply the linearization method [8] to covert (2b) into a linear constraint. In particular, we define the new variable $y'_j = X'_j r'_k$, which satisfies the following two linear constraints

$$0 \leq y'_j \leq r'_k \text{ (4)}$$

$$r'_k - r'_{k,\text{max}} (1-X'_j) \leq y'_j \leq r'_{k,\text{max}} X'_j \text{ (5)}$$

where $r'_{k,\text{max}}$ is a loose upper-bound on $r'_k$. The linearization technique is explained in more detail in the Appendix. Using (3)-(5) in (2), we obtain the following optimization problem:

Maximize $t$ \text{ (6a)}

$$t \leq \left(\frac{r'_k}{s'_k}\right) \forall i \in V, k \in S_j \text{ (6b)}$$

$$\sum_{i \in V} \sum_{k \in S_j} y'_j \leq C_r(j), \forall j \in P \text{ (6c)}$$

subject to:

$$\sum_{i \in V} r'_k \leq C_r(k), \forall k \in S \text{ (6d)}$$

$$0 \leq y'_j \leq r'_k \text{ (6e)}$$

$$r'_k - r'_{k,\text{max}} (1-X'_j) \leq y'_j \leq r'_{k,\text{max}} X'_j \text{ (6f)}$$

The constraints (2d) – (2i) \text{ (6g)}

The optimization problem in (6) is a linear mixed-integer optimization problem [9] which can be efficiently solved using a combination of the binary relaxation technique and branch-and-bound method [7]. These techniques are widely implemented by commercial optimization softwares such as MOSEK [10] and CPLEX [11]. The resulting placement algorithm will be used in the next section to obtain the optimal VM placements as well as the optimal VM data rates.

4. SIMULATION RESULTS

In this section, we study the effectiveness of the proposed placement algorithm using simulation results. For this purpose, we consider a cloud environment with two computing centers (denoted by $J_1, J_2$) and two storage centers (denoted by $S_1, S_2$). Without loss of generality, we assume that all virtual machines require equal amounts of computing power, memory, and disk resources.
resources. Computing centers $J_1$ and $J_2$ can host up to 100 and 50 virtual machines, respectively. Furthermore, for access link capacities we assume $C_p(1) = 4$ Gbps, $C_p(2) = 10$ Gbps, $C_d(1) = 7$ Gbps, and $C_d(2) = 10$ Gbps. We assume that the users request for a maximum of 10 virtual machines. The sizes of data files required by VMs 1 to 10 from storage centers $S_1$ and $S_2$ are listed in Table I. All file sizes are given in GBs. In order to focus on the effect of data file locations on the performance of the proposed placement algorithm, we consider the case that cloud users do not impose any constraints on the cost of VMs, i.e., we assume $P_{v_{max}} = \infty$ for all VMs. In addition, we assume that all VMs from previous placements have already completed their data transfers, i.e., $\bar{V}$ is an empty set.

In Fig. 2 we show the simulation results for the data transfer delay of the proposed algorithm versus the number of requested VMs. For comparison, we also show the simulation results for a random placement (RP) algorithm which randomly places the requested VMs on the computing centers. For RP, all possible VM placements are considered and the average data transfer delay for all placements are shown. Also shown in the figure is the simulation result for the placement algorithm of [4] which places virtual machines one-by-one without considering optimal rate allocation for VMs. This algorithm is assumed to employ TCP Vegas [6] for VM data rate allocation. As seen from the figure, the proposed algorithm significantly outperforms both RP and the algorithm of [4], as it carefully takes into account the location of the required data files and optimally assigns data rates to VMs.

### 5. CONCLUSIONS

In this paper, we have proposed a novel virtual machine placement algorithm that reduces the data transfer delay in cloud environments by the joint optimization of VM placement and VM data rate allocation. Simulation results revealed that the proposed algorithm is substantially more effective in reducing the data transfer delay compared to the competing algorithms in the literature.

### APPENDIX

**LINEARIZATION TECHNIQUE**

Let $x$ and $a$ be a non-negative real variable and a binary variable, respectively. Furthermore, let $y$ be the product of these variables. For variable $y = ax$ we can write

$$y = \begin{cases} x, & \text{if } a = 1 \\ 0, & \text{if } a = 0 \end{cases} \quad (7)$$

Using linearization technique, an equivalent representation for (7) can be obtained as follows [8]:

$$0 \leq y \leq x \quad (8)$$

$$x - x_{max}(1-a) \leq y \leq x_{max}a \quad (9)$$

where $x_{max}$ is a loose upper-bound on $x$.

### REFERENCES


### TABLE I: The size of data files required by VMs $i = 1, \ldots, 10$ from storage centers $S_1$ and $S_2$. All file sizes are given in GBs.

<table>
<thead>
<tr>
<th>VM</th>
<th>$S_1$</th>
<th>$V_1$</th>
<th>$V_2$</th>
<th>$V_3$</th>
<th>$V_4$</th>
<th>$V_5$</th>
<th>$V_6$</th>
<th>$V_7$</th>
<th>$V_8$</th>
<th>$V_9$</th>
<th>$V_{10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>$S_2$</td>
<td>20</td>
<td>40</td>
<td>60</td>
<td>80</td>
<td>100</td>
<td>120</td>
<td>140</td>
<td>160</td>
<td>180</td>
<td>200</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 2: Comparison of maximum data transfer delay for different placement algorithms. Solid lines with markers: Simulated data transfer time.