cost minimization of virtual machine placement and network in federated clouds

***Abstract*—We plan to consider the placement and migration of virtual machines under the dynamic request arrival scenario in the cloud federation platform. The objective is to minimize the cost of an autonomy cloud for both network consumption and virtual machine running.**

I. System Model

We study the problem that how the cloud federation minimizes its total cost for hosting virtual machines and transferring data. The cost of an individual cloud provider includes three parts: the operational cost for hosting virtual machines; the cost of data transferring among virtual machines; the migration cost.

Let J be the number of cloud providers in the cloud federation. We assume each cloud provider owns one data center. Each cloud in the federation provides M types of instances(i.e., virtual machines). Each type of instances corresponds to a set of configurations of CPU, storage and memory. The server resources in one cloud are as follows: Each cloud $j \in [1, J]$ has N_m^j homogeneous servers configured to provide instances of type m, each of which can provide at most H_m^j instances of this type. The total number of servers in cloud j is $\sum_{m=1}^{M} N_m^j$. The cloud providers are interconnected. The capacity into cloud j is B_I^j. The capacity out of cloud j is B_O^j.

The cost of hosting a type m instance in cloud j is β_m^j. The cost of transferring one volume of data out is λ_O^j. The cost of transferring one volume of data in is λ_I^j.

A. Instance Provision

Each cloud j needs to host the VMs of all running jobs in the federation. Let $r_m^{ji}(t)$ denote the number of available type m instances in cloud i that cloud j could host type m VMs in at time slot t.

The total number of type m instances provided by cloud j should not exceed the number that servers in cloud j can host, *i.e.*,

$$\sum_{i=1}^{J} r_m^{ij}(t) \le N_m^j \cdot H_m^j, 1 \le j \le J, 1 \le m \le M. \tag{1}$$

B. Job Scheduling

Each individual cloud receives job requests independently. There are K types of jobs. Job type $k \in [1, K]$ can be denoted by a three-tuple $< m_k, g_k, w_k >$. Here, m_k represents job type k requests type m_k VMs. g_k is the number of requested VMs. VMs are labeled from 1 to g_k. w_k is the request service time, i.e., the time duration that requested VMs should run continuously. An individual job l of type k_l is associated with a $g_{k_l} \times g_{k_l}$ traffic matrix T^l. In T^l, the entry of row r and column s is the traffic from r-th VM to s-th VM.

The system operates in a time slotted manner. Let $t = 0, 1, 2, 3..., T$ be the time slots. Let $\mathcal{A}_k^j(t)$ be the set of type k jobs arriving at time slot t in cloud j, $|\mathcal{A}_k^j(t)| = A_k^j(t)$.

Let $U_k^j(t)$ denote the number of newly served type k jobs in cloud j at time slot t. Then, the number of type k jobs at cloud j newly served at time slot $t-w, 0 \le w \le w_k - 1$ is $U_k^j(t-w), 0 \le w \le w_k - 1$. Index the jobs receiving service at time slot $t-w$ from 1 to $U_k^j(t-w)$. Let $\mathcal{L}^j(t)$ be the set of all jobs from cloud j being leftover and still served in the federation at time slot t, $|\mathcal{L}^j(t)| = \sum_{k=1}^{K} \sum_{w=1}^{w_k-1} U_k^j(t-w)$. Let $\mathcal{U}^j(t)$ be the set of all newly served jobs. A job l in $\mathcal{L}^j(t) \cup \mathcal{U}^j(t)$ can be denoted by a 3-tuple (k_l, t_l, h_l), which means the job is the h_l-th type k_l job that received service at time slot t_l, $1 \le h_l \le U_{k_l}^j(t_l)$. Job scheduling satisfies the following constraint, *i.e.*, the total number of available instances for cloud j should be larger than the total VMs of cloud j's running jobs,

$$\sum_{k:m_k=m} \sum_{w=1}^{w_k-1} g_k \cdot U_k^j(t-w) \le \sum_{i=1}^{J} r_m^{ji}(t), \\ 1 \le j \le J, 1 \le m \le M. \tag{2}$$

For each type of job, we use a queue $Q_k^j(t)$ to denote the workload of job type k at cloud j, here the workload refers to the total remaining service time slots needed for one type of jobs. The cloud federation guarantees to bound the service response time within a delay of d time slots, i.e., the time span from when the job arrives to when it starts to run on scheduled VMs. When a job's service response time can not be bounded within d, it is dropped. Let $D_k^j(t) \in [0, D_k^{j(max)}]$ denote the number of type-k jobs dropped by cloud j at time slot t. $D_k^{j(max)}$ is the maximum value of $D_k^j(t)$. Let α_k^j be the penalty to drop one such job.

The dynamic of $Q_k^j(t)$ is as follows:

$$Q_k^j(t+1) = \max\{Q_k^j(t) - \sum_{w=0}^{w_k-1} U_k^j(t-w) - w_k \cdot D_k^j(t), 0\} \\ + w_k \cdot A_k^j(t), 1 \le j \le J, 1 \le k \le K. \tag{3}$$

Job scheduling satisfies the following SLO constraint:

Each job is either served or dropped (subject to a penalty) before the maximum response delay d. (4)

We apply the ϵ-persistent queue technique [] to guarantee constraint (4). The delay can be guaranteed through the stability of virtual queues:

$$Z_k^j(t+1) = \max\{Z_k^j(t) + 1_{Q_k^j(t)>0} \cdot [\epsilon_k - \sum_{w=0}^{w_k-1} U_k^j(t-w)] - w_k \cdot D_k^j(t) - 1_{Q_k^j(t)=0} \cdot U_k^{max}, 0\}, \quad 1 \le j \le J, 1 \le k \le K. \tag{5}$$

C. VM placement

Hence, cloud j could place at most $r_m^{ji}(t)$ type m VMs in cloud i. For each individual running job in cloud j, $l \in \mathcal{L}^j(t) \cup \mathcal{U}^j(t)$, let $I_{l,s}^i(t)$ be the indicator whether the VM $s \in [1, g_{k_l}]$ is placed in cloud i or not at time slot t.

$$I_{l,s}^i(t) = 1 \text{ if instance } s \text{ is placed in cloud } i \text{ at time slot } t.$$
$$I_{l,s}^i(t) = 0 \text{ if not.} \tag{6}$$

The number of leftover type m VMs that are placed by cloud j in cloud i is $\sum_{l \in \mathcal{L}_m^j} \sum_{s=1}^{g_{k_l}} I_{l,s}^i(t), 1 \le i \le J$, $1 \le m \le M$. It can not exceed $r_m^{ji}(t)$,

$$\sum_{l \in \mathcal{L}_m^j} \sum_{s=1}^{g_{k_l}} I_{l,s}^i(t) \le r_m^{ji}(t), 1 \le i, j \le J, 1 \le m \le M. \tag{7}$$

Any VM from cloud j's jobs is placed,i.e.,

$$\sum_{i=1}^{J} I_{l,s}^i(t) = 1, l \in \mathcal{L}^j \cup \mathcal{U}^j, 1 \le s \le g_{k_l}, 1 \le j \le J. \tag{8}$$

D. Data traffic among clouds

Let us consider the data traffic among different cloud providers in the federation due to the placement and migration of VMs. There are two parts of traffic: one is the data traffic among different VMs of the same job; the other is the traffic induced by VM migrations.

First, we consider the traffic due to traffic among VMs of jobs. The traffic transferring out of cloud i due to jobs of cloud j is $\sum_{l \in \mathcal{L}^j \cup \mathcal{U}^j} \sum_{s_2 != s_1} I_{l,s_1}^i \cdot (1 - I_{l,s_2}^i) \cdot T_{s_1,s_2}^l$.

The traffic transferring into cloud i due to jobs of cloud j is $\sum_{l \in \mathcal{L}^j \cup \mathcal{U}^j} \sum_{s_2 != s_1} I_{l,s_1}^i \cdot (1 - I_{l,s_2}^i) \cdot T_{s_2,s_1}^l$

Next, let us consider the traffic due to migration. We assume the VM downtime during migration is short and can be ignored. We also assume the bandwidth used for migration is large, and the migration time can be ignored compared to the time slot. Let T_m be the size of transferred data for migrating a type m instance. We consider the transferred data due to the VM migration. The migration traffic out of cloud i due to migration of jobs in cloud j can be calculated as $\sum_{l \in \mathcal{L}^j} \sum_{s=1}^{g_{k_l}} [I_{l,s}^i(t-1) - I_{l,s}^i(t)]^+ \cdot T_{m_{k_l}}$

The migration traffic into cloud i due to migration of jobs in cloud j is $\sum_{l \in \mathcal{L}^j} \sum_{s=1}^{g_{k_l}} [I_{l,s}^i(t) - I_{l,s}^i(t-1)]^+ \cdot T_{m_{k_l}}$.

Hence, the data volume transferring out of cloud i due to jobs of cloud j is:

$$G_{O,i}^j(t) = \sum_{l \in \mathcal{L}^j \cup \mathcal{U}^j} \sum_{s_2 != s_1} I_{l,s_1}^i \cdot (1 - I_{l,s_2}^i) \cdot T_{s_1,s_2}^l + \sum_{l \in \mathcal{L}^j} \sum_{s=1}^{g_{k_l}} [I_{l,s}^i(t-1) - I_{l,s}^i(t)]^+ \cdot T_{m_{k_l}}$$

The data volume transferring into cloud i due to jobs of cloud j is:

$$G_{I,i}^j(t) = \sum_{l \in \mathcal{L}^j \cup \mathcal{U}^j} \sum_{s_2 != s_1} I_{l,s_1}^i \cdot (1 - I_{l,s_2}^i) \cdot T_{s_2,s_1}^l + \sum_{l \in \mathcal{L}^j} \sum_{s=1}^{g_{k_l}} [I_{l,s}^i(t) - I_{l,s}^i(t-1)]^+ \cdot T_{m_{k_l}}$$

E. Cost minimization problem definition

We first formulate the cost minimization problem for one cloud provider in the federation.

Let us first consider the cost of cloud j in the federation. The total cost includes the operational cost for running instances and transferring data.

The time-averaged operational cost for running instances is:

$$C_1^j = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{\sum_{i=1}^{J} \sum_{m=1}^{M} \beta_m^i(t) \cdot r_m^{ji}(t)\}, 1 \le j \le J$$

The time-averaged network cost related to transferring data out and in is :

$$C_2^j = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{\sum_{i=1}^{J} [\lambda_O^i(t) \cdot G_{O,i}^j(t) + \lambda_I^i(t) \cdot G_{I,i}^j(t)]\}, \quad 1 \le j \le J.$$

The cost due to penalty of dropping jobs is

$$C_3^j = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{\sum_{k=1}^{K} \alpha_k^j \cdot D_k^j(t)\}$$

The cost minimization problem at cloud j can be formulated as follows:

$$\min C_1^j + C_2^j + C_3^j$$
$$\text{constraints } 1 - 8.$$

F. Social cost minimization

The total cost of the federation is the overall cost of the cloud federation:

$$\sum_{j=1}^{J} [C_1^j + C_2^j + C_3^j]$$

The social cost minimization problem is:

$$\min \sum_{j=1}^{J} [C_1^j + C_2^j + C_3^j]$$
$$\text{constraints } 1 - 8.$$

TABLE I
IMPORTANT NOTATIONS

J	total number of cloud providers in the federation.
M	types of instances.
N_m^j	number of servers running type m instances in cloud j.
H_m^j	number of type m instances a server in cloud j can host.
K	the number of all job types.
g_k	the number of VMs required by job type k.
w_k	the service time of job type k.
m_k	type of VMs required by job type k.
$\mathcal{A}_k^j(t)$	set of type k jobs arriving at t in cloud j.
$A_k^j(t)$	number of type k jobs arriving at t in cloud j.
$U_k^j(t)$	number of newly served type k jobs at t in cloud j.
Q_k^j	queue backlog for workload of type k jobs in cloud j.
$D_k^j(t)$	number of dropped type k jobs at t in cloud j.
d	maximum responsive delay.
$\mathcal{L}^j$	set of all leftover jobs from cloud j being served.
T^l	the traffic matrix of job l.
r_m^{ij}	number of type m instances cloud i buys from cloud j.
$I_{l,s}^i$	indicator whether instance s of job l is placed in cloud i or not.
T_m	the data size of migrating a type m instance.
β_m^j	operational cost of hosting one type m virtual machine.
λ_O^j	cost of transferring one volume of data out of cloud j.
λ_I^j	cost of transferring one volume of data into cloud j.

The variables include $U_k^j(t)$, i.e., the number of new type k jobs served at time t, $I_{l,s}^j(t)$, i.e., the instance s assignment indicator, $r_m^{ij}(t)$, i.e., the number of type m VMs cloud i buys from cloud j.

We summarize important notations in Table I for ease of reference.

II. ALGORITHM DESIGN

We present a dynamic algorithm for each cloud to provision instances, schedule jobs and place VMs in different cloud providers. The goal is to minimize the total cost of virtual machine running and data transferring.

A. Cost minimization algorithm

Our dynamic algorithm operates in a time slotted manner. Due to the elegant performance of Lyapunov optimization, we apply Lyapunov optimization framework to design an online algorithm. But we consider different time length of jobs. Each time slot, our algorithm minimizes the drift-plus-penalty in the Lyapunov optimization framework, the details for the derivation of one slot drift-plus-penalty can be seen in appendix (**??**):

$$\begin{aligned}\min & \sum_{j=1}^{J}\sum_{k=1}^{K}[Q_k^j(t)+Z_k^j(t)]\cdot \\ & [-\sum_{w=0}^{w_k-1}U_k^j(t-w)-w_k\cdot D_k^j(t)] \\ & +V\cdot\sum_{j=1}^{J}[\sum_{i=1}^{J}\sum_{m=1}^{M}\beta_m^i(t)\cdot r_m^{ji}(t) \\ & +\sum_{i=1}^{J}\lambda_O^i(t)\cdot G_{O,i}^j(t)+\sum_{i=1}^{J}\lambda_I^i(t)\cdot G_{I,i}^j(t)+\sum_{k=1}^{K}\alpha_k^j\cdot D_k^j(t)]\end{aligned}$$

Constraints$1-8$.

The above optimization problem can be decomposed into the following two sub-problems:

$$\begin{aligned}\min & V\sum_{j=1}^{J}\sum_{i=1}^{J}\sum_{m=1}^{M}\beta_m^i(t)\cdot r_m^{ji}(t)-\sum_{j=1}^{J}\sum_{k=1}^{K}[Q_k^j(t)+Z_k^j(t)]\sum_{w=0}^{w_k-1}U_k^j(t-w) \\ & +V\sum_{j=1}^{J}[\sum_{i=1}^{J}\lambda_O^i(t)\cdot G_{O,i}^j(t)+\sum_{i=1}^{J}\lambda_I^i(t)\cdot G_{I,i}^j(t)]\end{aligned}$$

constraints$(1)-(8)$.

$$\begin{aligned}\min & \sum_{j=1}^{J}\sum_{k=1}^{K}\{V\alpha_k^j-[Q_k^j(t)+Z_k^j(t)]w_k\}D_k^j(t) \\ & 0\le D_k^j(t)\le D_k^{j(max)}\end{aligned}$$

The solution for the second sub-problem can be presented as follows:

$$D_k^j(t)=\begin{cases}D_k^{j(max)}, & \text{if } Q_k^j(t)+Z_k^j(t)>\frac{V\alpha_k^j}{w_k};\\ 0, & \text{if } Q_k^j(t)+Z_k^j(t)\le\frac{V\alpha_k^j}{w_k}.\end{cases}$$

Now let us consider the solution for the first sub-problem: First, let us see what is the optimal strategy for the individual cloud j to schedule jobs under known r_m^{ji}'s, $1\le i\le J, 1\le m\le M$.

1) What types of jobs and how many should we schedule at each time slot for cloud j?

We minimize the following optimization problem:

$$\begin{aligned}\min & -\sum_{k=1}^{K}[Q_k^j(t)+Z_k^j(t)]\sum_{w=0}^{w_k-1}U_k^j(t-w)\\ \text{constraint: } & (2).\end{aligned}$$

To minimize the above objective function, we should schedule type $k_{j,m}^*$ jobs for type m instances:

$$k_{j,m}^*=argmax_{k:m_k=m}\frac{Q_k^j(t)+Z_k^j(t)}{g_k}, 1\le j\le J$$

The number of scheduled type $k_{j,m}^*$ jobs should be:

$$U_{k_{j,m}^*}^j(t)\le\frac{\sum_{i=1}^{J}r_m^{ji}(t)-\sum_{k:m_k=m}\sum_{w=1}^{w_k-1}g_k\cdot U_k^j(t-w)}{g_{k_{j,m}^*}}, \quad 1\le j\le J$$

Discussions: For each type of virtual machines, each cloud $j, 1\le j\le J$ determines one type of jobs to schedule at each time slot. The type of jobs scheduled for type m instances should be $k_{j,m}^*=argmax_{k:m_k=m}\frac{Q_k^j(t)+Z_k^j(t)}{g_k}$. And the number of scheduled type $k_{j,m}^*$ jobs should be $U_{k_{j,m}^*}^j(t)\le\frac{\sum_{i=1}^{J}r_m^{ji}(t)-\sum_{k:m_k=m}\sum_{w=1}^{w_k-1}g_k\cdot U_k^j(t-w)}{g_{k_{j,m}^*}}$.

The second question is:

2) Should cloud i provision its type m instances? If so, which cloud should cloud i provision its type m instances to and how many?

We substitute $U_{k_{j,m}^*}^j=\frac{\sum_{i=1}^{J}r_m^{ji}(t)-\sum_{k:m_k=m}\sum_{w=1}^{w_k-1}g_k\cdot U_k^j(t-w)}{g_{k_{j,m}^*}}$ into the original problem.

Under known $I^i_{l,s}(t)$, we choose r^{ji}_m through minimizing the following problem:

$$\min \sum_{j=1}^{J}\sum_{i=1}^{J}\sum_{m=1}^{M}[\beta^i_m(t) - \frac{Q^j_{k^*_{j,m}}(t) + Z^j_{k^*_{j,m}}(t)}{g_{k^*_{j,m}}}] \cdot r^{ji}_m(t) \quad (9)$$

constraints: (1)(7)

We define $j^*_m = argmax_{j\in[1,J]} \frac{Q^j_{k^*_{j,m}}(t)+Z^j_{k^*_{j,m}}(t)}{g_{k^*_{j,m}}}$. Then, when cloud i satisfies $\beta^i_m(t) - \frac{Q^j_{k^*_{j,m}}(t)+Z^j_{k^*_{j,m}}(t)}{g_{k^*_{j,m}}}|_{j=j^*_m} < 0$, cloud i provisions type m virtual machines at time slot t. Cloud i provisions all its unoccupied type m virtual machines to cloud j^*_m. $r^{j^*_m i}_m(t) = N^i_m H^i_m - \sum_{l\in\mathcal{L}_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t) + \sum_{l\in\mathcal{L}^{j^*_m}_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t)$, the number of type m instances newly provisioned to cloud j^*_m from cloud i is $N^i_m H^i_m - \sum_{l\in\mathcal{L}_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t)$, *i.e.*, cloud i provisions all available type m instances not occupied by leftover jobs to cloud j^*_m. When for instance type m, cloud i satisfies $\beta^i_m(t) - \frac{Q^j_{k^*_{j,m}}(t)+Z^j_{k^*_{j,m}}(t)}{g_{k^*_{j,m}}}|_{j=j^*_m} \geq 0$, $r^{j^*_m i}_m(t) = \sum_{l\in\mathcal{L}^{j^*_m}_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t)$, $1 \leq j \leq J$. For other clouds $j \neq j^*_m$, $r^{ji}_m = \sum_{l\in\mathcal{L}^j_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t)$.

Discussions: For any cloud i satisfying $\beta^i_m(t) - \frac{Q^j_{k^*_{j,m}}(t)+Z^j_{k^*_{j,m}}(t)}{g_{k^*_{j,m}}}|_{j=j^*_m} \leq 0$, it provisions all available type m instances not occupied by leftover jobs to cloud j^*_m, $j^*_m = argmax_{j\in[1,J]} \frac{Q^j_{k^*_{j,m}}(t)+Z^j_{k^*_{j,m}}(t)}{g_{k^*_{j,m}}}$. $r^{j^*_m i} = N^i_m H^i_m - \sum_{l\in\mathcal{L}_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t) + \sum_{l\in\mathcal{L}^{j^*_m}_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t)$.

3) Then let us consider the virtual machine placement for jobs. The placement of virtual machines can be decomposed into two parts: the first part is the placement or migration of virtual machines from leftover jobs; the second part is the placement of virtual machines from newly served jobs.

For any type m instances, the placement of virtual machines from leftover jobs can be solved through the following optimization problem:

$$\begin{aligned}
\min & \sum_{j=1}^{J}\sum_{i\in I_{j^*_m}}\sum_{m=1}^{M}[\frac{Q^{j^*_m}_{k^*_{j^*_m,m}} + Z^{j^*_m}_{k^*_{j^*_m,m}}}{g_{k^*_{j^*_m,m}}} - \frac{Q^j_{k^*_{j,m}}(t) + Z^j_{k^*_{j,m}}(t)}{g_{k^*_{j,m}}}] \cdot \sum_{l\in\mathcal{L}^j_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t) \\
& + \sum_{j=1}^{J}\sum_{i\notin I_{j^*_m}}\sum_{m=1}^{M}[\beta^i_m(t) - \frac{Q^j_{k^*_{j,m}}(t) + Z^j_{k^*_{j,m}}(t)}{g_{k^*_{j,m}}}] \cdot \sum_{l\in\mathcal{L}^j_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t) \\
& + \sum_{j=1}^{J}\sum_{i=1}^{J} V\lambda^i_O(t)[\sum_{l\in\mathcal{L}^j}\sum_{s_2!=s_1} I^i_{l,s_1}\cdot(1-I^i_{l,s_2})\cdot T^l_{s_1,s_2} \\
& + \sum_{l\in\mathcal{L}^j}\sum_{s=1}^{g_{k_l}}[I^i_{l,s}(t-1) - I^i_{l,s}(t)]^+ \cdot T_{m_{k_l}}] \\
& + \sum_{j=1}^{J}\sum_{i=1}^{J} V\lambda^i_I(t)[\sum_{l\in\mathcal{L}^j}\sum_{s_2!=s_1} I^i_{l,s_1}\cdot(1-I^i_{l,s_2})\cdot T^l_{s_2,s_1} \\
& + \sum_{l\in\mathcal{L}^j}\sum_{s=1}^{g_{k_l}}[I^i_{l,s}(t) - I^i_{l,s}(t-1)]^+ \cdot T_{m_{k_l}}]
\end{aligned}$$

constraint: $\sum_{l\in\mathcal{L}_m}\sum_{s=1}^{g_{k_l}} I^i_{l,s}(t) \leq N^i_m H^i_m$

The placement of virtual machines from newly served jobs can be served through the following optimization problem:

$$\begin{aligned}
\min & \sum_{j=1}^{J}\sum_{i=1}^{J} V\lambda^i_O(t)\sum_{l\in\mathcal{U}^j}\sum_{s_2!=s_1} I^i_{l,s_1}\cdot(1-I^i_{l,s_2})\cdot T^l_{s_1,s_2} \\
& + \sum_{j=1}^{J}\sum_{i=1}^{J} V\lambda^i_I(t)\sum_{l\in\mathcal{U}^j}\sum_{s_2!=s_1} I^i_{l,s_1}\cdot(1-I^i_{l,s_2})\cdot T^l_{s_2,s_1}
\end{aligned}$$

(Solve the above two optimization problems to get $I^i_{l,s}(t)$. Will continue to work on how to solve them heuristically.)

III. Game theory analysis

In the previous section, we study how to minimize the total cost of the cloud federation under the condition that individual clouds' resource allocations are under the central control. There exists conflicts among resource allocations to individual clouds. When one cloud obtains the resource, other clouds will not be able to use it. Hence, we can use a game to model the resource allocation among individual clouds. Each individual cloud intends to minimize its individual cost.

We could try to answer the following three questions:

1. Does the Nash equilibrium exist?

2. How to calculate the Nash equilibrium?

3. What is the price of anarchy?

We first prove that each individual cloud could apply the Lyapunov optimization framework. The individual cloud's decisions based on the Lyapunov optimization under any others' strategies can achieve close-to-optimal performance (will write it down in appendix).

The one shot optimization problem for any individual cloud j is:

$$\begin{aligned}
\min & \sum_{k=1}^{K}[Q^j_k(t) + Z^j_k(t)]\cdot[-\sum_{w=0}^{w_k-1} U^j_k(t-w) - w_k\cdot D^j_k(t)] \\
& + V\cdot[\sum_{i=1}^{J}\sum_{m=1}^{M}\beta^i_m(t)\cdot r^{ji}_m(t) \\
& + \sum_{i=1}^{J}\lambda^i_O(t)\cdot G^j_{O,i}(t) + \sum_{i=1}^{J}\lambda^i_I(t)\cdot G^j_{I,i}(t) + \sum_{k=1}^{K}\alpha^j_k\cdot D^j_k(t)]
\end{aligned}$$

Constraints$(1)(2)(6)-(8)$.

The interactions are only among the competitions among VM provisioning, i.e., r^{ji}_m's, $1 \leq j,i \leq J, 1 \leq m \leq M$. The job scheduling and VM placement are separate. Hence, we consider how individual cloud buys VMs from other clouds.

The VM buying problem can be separated by dual decomposition method. The primal problem is problem (9). We can apply dual decomposition method to solve (9) distributedly. Define the partial Lagrangian:

$$
\begin{aligned}
L(\mathbf{r}_m^{ji},\nu) &= \sum_{j=1}^{J}\sum_{i=1}^{J}\sum_{m=1}^{M}[\beta_m^i(t) - \frac{Q_{k_{j,m}^*}^j(t) + Z_{k_{j,m}^*}^j(t)}{g_{k_{j,m}^*}}]\cdot r_m^{ji}(t) \\
&+ \sum_{j=1}^{J}\sum_{i=1}^{J}[V\lambda_O^i(t)G_{O,i}^j(t) + V\lambda_I^i G_{I,i}^j(t)] \\
&+ \sum_{j=1}^{J}\sum_{m=1}^{M}\nu_m^j(\sum_{i=1}^{J} r_m^{ij}(t) - N_m^j\cdot H_m^j) \\
&= \sum_{j=1}^{J}\{\sum_{i=1}^{J}\sum_{m=1}^{M}[\beta_m^i(t) - \frac{Q_{k_{j,m}^*}^j(t) + Z_{k_{j,m}^*}^j(t)}{g_{k_{j,m}^*}}]\cdot r_m^{ji}(t) \\
&+ \nu_m^i r_m^{ji}(t) + \sum_{i=1}^{J}[V\lambda_O^i(t)G_{O,i}^j(t) + V\lambda_I^i G_{I,i}^j(t)]\} \\
&- \sum_{j=1}^{J}\sum_{m=1}^{M}\nu_m^j N_m^j H_m^j
\end{aligned}
$$

The subproblem for cloud j is:

$$
\begin{aligned}
&\min \sum_{i=1}^{J}\sum_{m=1}^{M}[\beta_m^i(t) + \nu_m^i(t) - \frac{Q_{k_{j,m}^*}^j(t) + Z_{k_{j,m}^*}^j(t)}{g_{k_{j,m}^*}}]r_m^{ji}(t) \\
&+ \sum_{i=1}^{J}[V\lambda_O^i(t)G_{O,i}^j(t) + V\lambda_I^i G_{I,i}^j(t)]
\end{aligned}
$$

constraint(6)(7)(8)

IV. Algorithm Performance

Appendix

Let $\Theta^j(t) = (\mathbf{Q}^j, \mathbf{Z}^j)$ be the vector of actual queues and virtual queues in cloud j. $\mathbf{\Theta}(t) = (\Theta^1(t), \Theta^2(t), ..., \Theta^J(t))$. The Lyapunov function of $\mathbf{\Theta}(t)$ is:

$$
L(\Theta(t)) = \frac{1}{2}\sum_{j=1}^{J}\sum_{k=1}^{K}[Q_k^j(t)^2 + Z_k^j(t)^2] \tag{10}
$$

The one slot drift is

$$
\begin{aligned}
&\Delta(\Theta(t)) \le B + \sum_{j=1}^{J}\sum_{k=1}^{K} Q_k^j(t)\cdot \\
&[w_k\mathbb{E}[A_k^j(t)] - \sum_{w=0}^{w_k-1} U_k^j(t-w) - w_k\cdot D_k^j(t)] + \sum_{j=1}^{J}\sum_{k=1}^{K} Z_k^j(t)\cdot \\
&[1_{Q_k^j(t)>0}(\epsilon_k - \sum_{w=0}^{w_k-1} U_k^j(t-w)) - w_k D_k^j(t) - 1_{Q_k^j(t)=0}U_k^{max}]
\end{aligned}
$$

B is a constant.

The drift plus penalty is:

$$
\begin{aligned}
&\Delta(\Theta(t)) + V\cdot\sum_{j=1}^{J}\{\sum_{i=1}^{J}\sum_{m=1}^{M}\beta_m^i(t)\cdot r_m^{ji}(t) \\
&+ \sum_{i=1}^{J}[\lambda_O^i(t)\cdot G_{O,i}^j(t) + \lambda_I^i(t)\cdot G_{I,i}^j(t)] + \sum_{k=1}^{K}\alpha_k^j\cdot D_k^j(t)\} \\
&\le B + \sum_{j=1}^{J}\sum_{k=1}^{K} Q_k^j(t)\cdot[w_k\mathbb{E}[A_k^j(t)] - \sum_{w=0}^{w_k-1} U_k^j(t-w) - w_k\cdot D_k^j(t)] \\
&+ \sum_{j=1}^{J}\sum_{k=1}^{K} Z_k^j(t)\cdot[(\epsilon_k - \sum_{w=0}^{w_k-1} U_k^j(t-w)) - w_k D_k^j(t)] \\
&+ V\cdot\sum_{j=1}^{J}[\sum_{i=1}^{J}\sum_{m=1}^{M}\beta_m^i(t)\cdot r_m^{ji}(t) \\
&+ \sum_{i=1}^{J}\lambda_O^i(t)\cdot G_{O,i}^j(t) + \sum_{i=1}^{J}\lambda_I^i(t)\cdot G_{I,i}^j(t) + \sum_{k=1}^{K}\alpha_k^j\cdot D_k^j(t)]
\end{aligned}
$$

We minimize the right-hand-side of the above drift-plus-penalty expression.

$$
\begin{aligned}
&\min \sum_{j=1}^{J}\sum_{k=1}^{K}[Q_k^j(t) + Z_k^j(t)]\cdot \\
&[-\sum_{w=0}^{w_k-1} U_k^j(t-w) - w_k\cdot D_k^j(t)] \\
&+ V\cdot\sum_{j=1}^{J}[\sum_{i=1}^{J}\sum_{m=1}^{M}\beta_m^i(t)\cdot r_m^{ji}(t) \\
&+ \sum_{i=1}^{J}\lambda_O^i(t)\cdot G_{O,i}^j(t) + \sum_{i=1}^{J}\lambda_I^i(t)\cdot G_{I,i}^j(t) + \sum_{k=1}^{K}\alpha_k^j\cdot D_k^j(t)]
\end{aligned}
$$

Constraints$1-8$.