MDP Modeling of Resource Provisioning in Virtualized Content-Delivery Networks

Ali A. Haghighi, Shahram Shah Heydari, Shahram ShahbazPanahi
University of Ontario Institute of Technology (UOIT), Oshawa, Canada

***Abstract*—In this paper a Markov decision process (MDP) model for virtualized content delivery networks is proposed. We use stochastic optimization to assign cloud site resources to each user group. We propose how quality of experience (QoE) can be included in the modeling and optimization. We then present an optimal solution for a constraint-free version of the problem, and show the improvement in accumulated revenue when our optimization model is used. A sub-optimal algorithm is proposed that would reduce the complexity of the problem. Simulation results are presented to support merits of the proposed algorithm.**

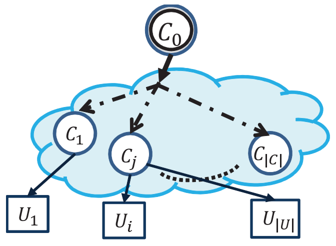

Fig. 1. The cluster graph model of the virtualized CDN.

I. Introduction

Resource provisioning in a cloud-based or virtualized content delivery network (CDN) has some inherent implementation features including 1) significant initialization time, 2) high overhead for updating assignment and 3) limited leased time of resources [1]. These features accompanied with *abrupt* demand variations of user groups, motivate us to reconsider conventional CDN resource management that merely is based on a snapshot of demand values (see e.g., [1–3]). Resource assignment in virtualized CDN can be considered as replica placement and/or allocating virtual bandwidth, storage and/or virtual machine at each cloud site. Inclusion of statistical behavior such as average demand values in resource assignment policies can be useful in term of saving resources but it is still not effective enough because future decrease or increase in demands will lead to either waste in resources or QoS penalties and significant bandwidth cost due to a root redirection. Therefore, it would be highly profitable to have a scheme for virtualized CDN resource provisioning that, in addition to the snapshot of network parameters, considers future demand variations into account as well in cost/revenue analysis. This would be particularly useful in networks where the variation in demand can be statistically characterized or follows a traceable memory based model. The above reasons lead us to a Markov decision process (MDP) based modeling of virtualized CDN and stochastic optimization for resource assignment. Stochastic optimization based on MDP modeling is performed by dynamic programming (DP) which is usually a recursive algorithm that guarantees the global optimum and can be implemented in the network through a lookup table.

MDP modeling for resource assignment has been used in other network types in the past. In particular, this method has been used in wireless sensor networks to manage recharging or harvest energy at nodes [4–6].

In virtualized CDN, research on MDP modeling and optimization is at early stages [7], [8], where it is primarily used for dynamic allocation of cloud resources for network function virtualization (NFV) components. In particular, [8] studies time slot allocation on a resource to an NFV component, while state transition probabilities are modeled as the reliability of NFV resources which are estimated via a machine learning based method.

In general, considering stochastic optimization based on MDP modeling allows better resource assignment in virtualized CDN by taking statistics of future network states and expectation of future rewards into account. In our model, we also propose a QoE index that is provided by each user group after a resource assignment. QoE index denotes the level of satisfaction of each user group and defined by quantized levels.

In the paper, we use t for time instant, Mathcal Font ($\mathcal{A}$) for set, $|.|$ for set cardinality, bold capital letter for matrix, bold letter for vector, and underline for quantisation value. Also, $[.]_{(i,j)}$ is the element (i,j) of its matrix argument, and $[.]_j$ is the element j of its vector argument.

II. System model and MDP formulation

We consider a groups of users, $\mathcal{U} = \{i\}_{i=1}^{|\mathcal{U}|}$ located in $|\mathcal{U}|$ abstracted locations, by contents originated from a source cloud site C_0 using a set of $|\mathcal{C}|$ surrogate cloud sites, $\mathcal{C} = \{j\}_{j=1}^{|\mathcal{C}|}$ (see Fig. 1). User group i and cloud site j are labeled by U_i and C_j, respectively. Each user group U_i experiences D_{ij} time unit as delay to be served by cloud site C_j. We assume a single cloud provider and a single content size. In the multicast clustered graph model in Fig. 1, each U_i is served by a root site C_0 through cloud site(s) C_j with an efficient usage of cloud site resources in order to maximize total revenue defined later on. The bandwidth or the communication capacity of each C_j is upper bounded to B_j.

Each MDP modeling includes state space and action space modeling, reward definition and transition probability of states which we explain each of them in the following.

A. State space

Demand rate for each user group U_i at time instant t is denoted by $d_i(t)$ that takes one of the quantisation values

$$d_i(t) \in \mathcal{D},\ \mathcal{D} = \{\underline{d}_k\}_{k=1}^{|\mathcal{D}|}. \tag{1}$$

Also,

$$\mathbf{d}_t = (d_1(t), \dots, d_{|\mathcal{U}|}(t)),\ \mathbf{d}_t \in \mathcal{D}^{|\mathcal{U}|}. \tag{2}$$

We consider demands as time dependent random variables.

We assume a quality of experience index value, $q_i(t)$, is fed back from each U_i to the network manager in each time instant t. Also,

$$q_i(t) \in \mathcal{Q},\ \mathcal{Q} = \{\underline{q}_k\}_{k=1}^{|\mathcal{Q}|}, \tag{3}$$

and

$$\mathbf{q}_t = (q_1(t), \dots, q_{|\mathcal{U}|}(t)),\ \mathbf{q}_t \in \mathcal{Q}^{|\mathcal{U}|}. \tag{4}$$

Note that the system is considered to be casual meaning $\mathbf{q}_t$ is available at network manager based on resource provisioning made at time instant $t-1$.

Therefore, the state of the network in time instant t is as follows

$$\mathbf{s}_t = (d_1(t), \dots, d_{|\mathcal{U}|}(t), q_1(t), \dots, q_{|\mathcal{U}|}(t)),\ \mathbf{s}_t \in \mathcal{S}, \tag{5}$$

where $\mathcal{S}$ is the state space and $\mathcal{S} \subseteq (\mathcal{Q} \times \mathcal{D})^{|\mathcal{U}|}$.

B. Action space

Let $x_i(t),\ x_i(t) \in \mathcal{C},\ i = 1, \dots, |\mathcal{U}|$, be the assignment variable implying the cloud site which serves demand $d_i(t)$. For example, $x_1(t) = 3$ means $d_1(t)$ is served by C_3. So, we denote the action vector in time instant t, $\mathbf{x}_t$ by

$$\mathbf{x}_t = (x_1(t), \dots, x_{|\mathcal{U}|}(t)). \tag{6}$$

In each state $\mathbf{s}_t$, an action $\mathbf{x}_t$ is selected from the corresponding action set $\mathcal{A}^{\mathbf{s}_t}$ defined by

$$\begin{aligned}\mathcal{A}^{\mathbf{s}_t} = \big\{\mathbf{x}_t | \sum_{i=1}^{|\mathcal{U}|} d_i(t) I(x_i(t) = j) \le B_j, \\ j \in \mathcal{C},\ \mathbf{x}_t \in \mathcal{C}^{|\mathcal{U}|}\big\},\end{aligned} \tag{7}$$

where B_j is the bandwidth limit of C_j and $I(.)$ is an indicator function and is equal to one if its argument is true and zero, otherwise. The action space is defined as

$$\mathcal{A} = \bigcup_{\mathbf{s}_t \in \mathcal{S}} \mathcal{A}^{\mathbf{s}_t}. \tag{8}$$

C. Reward

Both profit and cost are included in the reward function. Reward in each time instant t is a function of state and action, $R(\mathbf{s}_t, \mathbf{x}_t)$, defined as follows,

$$R(\mathbf{s}_t, \mathbf{x}_t) = \sum_{i \in \mathcal{U}} w_i^p d_i(t) + w_i^q q_i(t) \tag{9}$$

$$- \sum_{i \in \mathcal{U}, j \in \mathcal{C}} P_j^b d_i(t) I(x_i(t) = j) \tag{10}$$

where the first summation reflects profit and included QoE of all users, w_i^p is profit per demand and w_i^q is a normalization factor. Second summation is the bandwidth cost used in each cloud site and P_j^b is bandwidth unit price.

D. Transition probabilities

Transition probability in each time instant t, denoted by $p_t(\mathbf{s}'|\mathbf{s}, \mathbf{x})$, is the probability of changing network state from the state $\mathbf{s}$ in time t, to the state $\mathbf{s}'$ in time $t+1$, given an action $\mathbf{x}$. In other words, we have

$$\begin{aligned} p_t(\mathbf{s}'|\mathbf{s}, \mathbf{x}) &= \Pr(\mathbf{s}_{t+1} = \mathbf{s}' | \mathbf{s}_t = \mathbf{s}, \mathbf{x}_t = \mathbf{x}) && (11)\\ &= \Pr(\mathbf{d}_{t+1} = \mathbf{d}' | \mathbf{d}_t = \mathbf{d}, \mathbf{x}_t = \mathbf{x}) && (12)\\ &\times \Pr(\mathbf{q}_{t+1} = \mathbf{q}' | \mathbf{q}_t = \mathbf{q}, \mathbf{x}_t = \mathbf{x}), && (13)\end{aligned}$$

where demand and QoE vectors are considered to be independent. We assume transition probabilities do not depend on time that means *stationary* stochastic process is considered. In the following we analyze (12) and (13).

We can assume user group demands are independent of each other and of the actions in each state, which leads to

$$\Pr(\mathbf{d}_{t+1} = \mathbf{d}' | \mathbf{d}_t = \mathbf{d}, \mathbf{x}_t = \mathbf{x}) \tag{14}$$

$$= \prod_{i \in \mathcal{U}} \Pr(d_i(t+1) = d_i' | d_i(t) = d_i). \tag{15}$$

So having demands transition probability matrix of each user i denoted by $\mathbf{P}_i^d$, $[\mathbf{P}_i^d]_{(m,n)} := \Pr(d_i(t+1) = \underline{d}_n | d_i(t) = \underline{d}_m)$ is sufficient to obtain (15). Note that $\sum_{n=1}^{|\mathcal{D}|} [\mathbf{P}_i^d]_{(m,n)} = 1$.

In our model, we assume QoE of each user group U_i, chiefly depends on the experienced delay by U_i, defined as

$$\bar{D}_i(t) = \sum_{j \in \mathcal{C}} I(x_i(t) = j) D_{ij}. \tag{16}$$

Thus, for (13), QoE is influenced by action. We assume for a given action, QoE level does not depend on the previous QoE levels. In other words,

$$\begin{aligned}&\Pr(\mathbf{q}_{t+1} = \mathbf{q}' | \mathbf{q}_t = \mathbf{q}, \mathbf{x}_t = \mathbf{x})\\ &= \Pr(\mathbf{q}_{t+1} = \mathbf{q}' | \mathbf{x}_t = \mathbf{x})\\ &= \prod_{i \in \mathcal{U}} \Pr\left(q_i(t+1) = q' \middle| x_i(t) = x_i\right)\\ &= \prod_{i \in \mathcal{U}} \Pr(q_i(t+1) = q' | \bar{D}_i(t) = \bar{D}_i),\end{aligned} \tag{17}$$

where $\bar{D}_i(t)$ is given in (16). For each term $\Pr(q_i(t+1) = q' | \bar{D}_i(t) = \bar{D}_i)$ in (17) we assume QoE feedbacks follow the following probability mass function (pmf)

$$\Pr(q_i(t) = \underline{q}_{|\mathcal{Q}|-k+1}, \bar{D}_i) = \begin{cases} M\alpha, D_{k-1}^b < \bar{D}_i < D_k^b, \\ \alpha, \quad \text{otherwise}, \end{cases} \tag{18}$$

where M, $M > 1$ is a design parameter, $D_k^b,\ k = 1, \dots, |\mathcal{Q}|$ are fixed thresholds and $D_0^b = 0$. Also, summation over all cases in (18) is 1 which yields α to be

$$\alpha = \frac{1}{|\mathcal{Q}| + M - 1}.$$

Using the pmf (18), (17) is obtained. The considered model for QoE feedback says for example for very low delay value $\bar{D}_i < D_1^b$, probability of receiving the highest QoE level, i.e., $q_{|\mathcal{Q}|}$, is the highest value $M\alpha$ while the probability of receiving the other feedback levels are α.

III. Resource Optimization Formulation

A. Objective function

A policy is a rule that determines a decision for the given state of the system and is denoted by π [9]. Each action $\mathbf{x}_t$ is a function of state of the system denoted by $\mathbf{x}_t = A_t^\pi(\mathbf{s}_t)$ where $A_t^\pi : \mathcal{S} \rightarrow \mathcal{A}^{\mathbf{s}}$. For the starting state $\mathbf{s}_0 = \mathbf{s}$ and the policy π, infinite horizon discounted reward is defined as follows

$$J_\gamma^\pi(\mathbf{s}) = \mathbb{E}_{\mathbf{s}_t}^\pi \left(\sum_{t=0}^{\infty} \gamma^t R\big(\mathbf{s}_t, \mathbf{x}_t\big) \Big| \mathbf{s}_0 = \mathbf{s} \right), \tag{19}$$

where expectation is over all possible sequence of states that the policy π is used. Also, $\gamma \in (0,1)$ is a discounting factor. For the given starting state $\mathbf{s}_0$ our objective is to find the maximum discounted reward in infinite horizon scenario and the corresponding best policy. In other words,

$$J^*(\mathbf{s}) = \max_{\pi \in \Pi} J_\gamma^\pi(\mathbf{s}). \tag{20}$$

Note that for $\gamma = 0$, only the first term in (19) is considered in the optimization, i.e., $J_\gamma^\pi(\mathbf{s}) = R\big(\mathbf{s}_0, \mathbf{x}_0\big),\ \mathbf{s}_0 = \mathbf{s}$, which is equivalent to the conventional myopic approach where only the current *observed* state, $\mathbf{s}_0 = \mathbf{s}$ is considered in the optimization. We consider myopic scenario in the simulation results as a benchmark.

B. Dynamic programming

The optimization problem (20) is known as infinite horizon discounted reward problem. A family of schemes known as dynamic programming (DP) can be used to solve stochastic problems that are based on MDP models. There are different DP algorithms to solve the optimization problem (20), such as value iteration, policy iteration, linear programming based, and the hybrid schemes. Value iteration is the most suitable scheme due to the simplicity in implementation. In Algorithm 1 we present value iteration algorithm to solve (20). Note that n is the iteration number and not time index. Also, π^n is the policy used to obtain v^n, and $||v||$ is the largest absolute value defined as follows

$$||v|| = \max_{\mathbf{s}} |v(\mathbf{s})|. \tag{21}$$

The algorithm is run to obtain the best policy π^ϵ and the objective value v^ϵ that is an approximation of $J^*(\mathbf{s})$ with the tolerance ϵ, i.e., $|v^\epsilon - J^*(\mathbf{s})| < \epsilon$.

C. Challenges

Main challenges of the optimization model that was presented include 1) very large state space and 2) large action space. In particular, exponential complexity of the computations with regard to the number of users is the main drawback of the implementation such that the size of the state/action space for a real CDN could not be implemented in terms of the required processing and memory.

For value iteration algorithm 1 we need to investigate $O = |\mathcal{S}|^2|\mathcal{A}|$ cases in each iteration that for the model is

$$O = |\mathcal{D} \times \mathcal{Q}|^{2|\mathcal{U}|}|\mathcal{C}|^{|\mathcal{U}|}.$$

Algorithm 1 Value Iteration

```
0- Initialization,
   • Fix $\epsilon > 0$.
   • $v^0(\mathbf{s}) \leftarrow 0,\ \forall \mathbf{s} \in \mathcal{S}$
   • $n \leftarrow 1$
1- For $\forall \mathbf{s} \in \mathcal{S}$ compute
       $v^n(\mathbf{s}) \leftarrow \max_{\mathbf{x} \in \mathcal{A}} \left( R(\mathbf{s},\mathbf{x}) + \gamma \sum_{\mathbf{s}' \in \mathcal{S}} \Pr(\mathbf{s}'|\mathbf{s},\mathbf{x}) v^{n-1}(\mathbf{s}') \right)$
2- If $||v^n - v^{n-1}|| < \epsilon(1-\gamma)/2\gamma$,
   • $v^\epsilon \leftarrow v^n$
   • $\pi^\epsilon \leftarrow \pi^n$
   else
   • $n \leftarrow n+1$
   • Go to 1.
3- Return $v^\epsilon$ and $\pi^\epsilon$.
```

For instant, even a small size network with $|\mathcal{D}| = 4$, $|\mathcal{Q}| = 3$, $|\mathcal{U}| = 4$, $|\mathcal{C}| = 3$, needs $O \sim 3.47\text{e}10$ investigations per iteration which is not feasible. Although the algorithm is not needed to be run in real time to provide the lookup table, a high volume of memory is required to store it. In the following we propose alternative solutions to tackle these challenges.

IV. Approximations and Suboptimal Algorithms

Here we address the challenges mentioned in Section III-C. We start with unlimited bandwidth scenario as an especial case of the problem and prove that the DP implementation can be performed by a significantly reduced complexity. Then, we deal with limited bandwidth scenario and propose a DP based algorithm to solve the problem.

A. Unlimited bandwidth and BW$_\infty$ algorithm

If the bandwidth of each cloud site is considered to be unlimited, meaning each cloud site is able to serve all user groups, we can assume that the resource provisioning of each user group is independent of the other groups. We define state for U_i as

$$\mathbf{s}_t^i = (d_i(t), q_i(t)), \mathbf{s}_t^i \in \mathcal{S}^i, \mathcal{S}^i = \mathcal{D} \times \mathcal{Q}, \tag{22}$$

where $\mathcal{S}^i$ is the corresponding state space. The action is $x_i(t),\ x_i(t) \in \mathcal{C}$ meaning the action space for U_i, denoted by $\mathcal{A}^i$, is $\mathcal{A}^i = \mathcal{C}$.

Reward for each U_i is written as

$$\begin{aligned} R^i(\mathbf{s}_t^i, x_i(t)) = {} & w_i^p d_i(t) + w_i^q q_i(t) \\ & - \sum_{j \in \mathcal{C}} P_j^b d_i(t) I(x_i(t) = j). \end{aligned} \tag{23}$$

Furthermore, transition probability for U_i is obtained as

$$\begin{aligned} & p_t^i(\mathbf{s}'|\mathbf{s}, x) = \Pr(\mathbf{s}_{t+1}^i = \mathbf{s}'|\mathbf{s}_t^i = \mathbf{s}, x_i(t) = x) \\ & = \Pr(d_i(t+1) = d'|d_i(t) = d) \Pr(q_i(t+1) = q'|x_i(t) = x) \\ & = \Pr(d_i(t+1) = d'|d_i(t) = d) \\ & \qquad \times \Pr(q_i(t+1) = q'|\bar{D}_i(t) = \bar{D}_i). \end{aligned} \tag{24}$$

The infinite horizon discounted reward as objective function for each U_i is as follows

$$J_\gamma^{\pi^i}(\mathbf{s}^i) = \mathbb{E}_{\mathbf{s}_t^i}^{\pi^i}\left(\sum_{t=0}^{\infty}\gamma^t R\big(\mathbf{s}_t^i, x_i(t)\big)\bigg|\mathbf{s}_0^i = \mathbf{s}^i\right), \tag{25}$$

where π^i is a selected policy for U_i. For the considered infinite bandwidth scenario, (25) and (19) satisfy the following equation

$$J_\gamma^{\pi}(\mathbf{s}) = \sum_{i\in\mathcal{U}} J_\gamma^{\pi^i}(\mathbf{s}^i), \tag{26}$$

where $\mathbf{s} = (\mathbf{s}^1, \ldots, \mathbf{s}^{|\mathcal{U}|})$ and π is the set of individual policies, π^i, of all user groups.

Equation (26) says for unlimited bandwidth scenario we can use DP individually for each user group (see Algorithm 2). This reduced size of state/action space can significantly reduce complexity and speed up obtaining the optimal policy. In other words, using value iteration algorithm for each user group, we need to investigate the following number of cases in each iteration to reach the optimal policy

$$O = |\mathcal{U}| \times |\mathcal{S}^i|^2|\mathcal{A}^i| = |\mathcal{U}||\mathcal{D}\times\mathcal{Q}|^2|\mathcal{C}|.$$

For example, for the same simulation scenario in Section III-C, i.e., $|\mathcal{D}| = 4$, $|\mathcal{Q}| = 3$, $|\mathcal{U}| = 4$, $|\mathcal{C}| = 3$, we have $O =$ 1.7e3 which is significantly lower than that of the conventional model i.e., $O \sim 3.47\text{e}10$, mentioned before.

Algorithm 2 BW_∞.

```
1- For $\forall U_i, i \in \mathcal{U}$,
    • Perform Algorithm 1 independently for each $U_i$ using the corresponded inputs,
    • Save value, $\pi_i^\epsilon \leftarrow \pi^\epsilon$, $v_i^\epsilon \leftarrow v^\epsilon$
2- Return optimal value and policy, $(v^\epsilon, \pi^\epsilon)$;
    $v^\epsilon \leftarrow \sum_{i\in\mathcal{U}} v_i^\epsilon$, $\pi^\epsilon \leftarrow \{\pi_i^\epsilon\}_{i=1}^{\mathcal{U}}$.
```

B. Divide-and-Conquer algorithm

The proposed algorithm here is based on a family of algorithms named divide-and-conquer (DaQ). In this approach, the problem is recursively divided to some sub-problems with low enough complexity to be solved directly, then combined to give a solution for the main problem. DaQ is specially suitable for example for biconvex optimization problems (see e.g., [10]).

In the DaQ algorithm, we perform DP-based optimization in each round only for one variable while the other variables are fixed. Then, we repeat the algorithm until it is converged to a solution. We show Algorithm DaQ in Algorithm 3. We use π to denote the current policy i.e., the set of current actions within each round, V for objective function, superscript * for optimum value and subscript $_{(k)}$ for the index corresponded to the selected best user in round k.

In the algorithm, we start with a feasible, but not necessarily optimal, policy π_0 which can be, for example, assigning all U_i to the root cloud site C_0. In each round k of DaQ, 1) we perform DP to obtain the best action for each user group while the actions of the other user groups are fixed (Step 3), 2) we select the best user with maximum objective function (Step 4). In Step 5, we update the policy based on the selected best user. We repeat this process until all actions are selected. Note that Step 2 and the last action of Step 3 are to make sure the previous assigned actions in the earlier rounds, are not investigated again. In Step 7, we update the objective function and in Step 8 we check the convergence of the algorithm.

Note that updating π in each round means updating action space for the non-optimized actions.

Algorithm 3 DaQ.

```
0- Initials:
    • Input $\mathbf{s}_0$
    • Fix $\delta > 0$
    • Fix $\pi_0$
    • $n \leftarrow 0$
    • $V_n^* \leftarrow$ Objective function corresponding to $\mathbf{s}_0$ and $\pi_0$ (compute by Algorithm 1)
1- Assign loop variables
    • $\pi \leftarrow \pi_0$,
    • $k \leftarrow 0$
2- $i \leftarrow$ Minimum index of the set $\{U_1, \ldots, U_{|\mathcal{U}|}\}\backslash\{U_{(1)}, \ldots, U_{(k)}\}$
3- While $i \le |\mathcal{U}|$
    • For $\pi$, fix $x_j, \forall j \neq i, j \in \mathcal{U}$
    • Use Algorithm 1 to solve the problem for $x_i$ and
    • save the objective function for $\mathbf{s}_0$, $V_{ki} \leftarrow v^\epsilon$, and the corresponding argument/action.
    • $i \leftarrow i + 1$ (*)
    • If $i \in$ one of the indexes of the set $\{U_{(1)}, \ldots, U_{(k)}\}$,
      Go to the line (*).
4- Obtain maximum objective and the best action
    • $k \leftarrow k + 1$.
    • $V_k \leftarrow \max_i V_{ki}$,
    • $x_{(k)} \leftarrow \arg\max_i V_{ki}$
5- Update $\pi$ based on $x_{(k)}$
6- If $k < |\mathcal{U}|$,
      Go to **2**.
7- Update round values
    • $n \leftarrow n + 1$,
    • $V_n^* \leftarrow V_k$
8- If $|V_n^* - V_{n-1}^*| > \delta$,
    • $\pi_0 \leftarrow \pi$
    • Go to **1**.
9- Return $V_n^*$, $\pi$.
```

Note that we do not need to use the state space of the whole network defined in Section II-A in Step 3. This is because the actions of $|\mathcal{U}| - 1$ user groups are fixed. So, in order to obtain the best action for the single under investigation user group, we only need to search over its action space. This action space is determined based on the remained available bandwidth of cloud sites which depends on the fixed actions of the other users. Thus, to implement Step 3, at first, for the user groups with fixed action we obtain the objective function for each individual user separately using the fixed single action. Then, we obtain the objective function as well as the best action for the under investigation user group. Finally, we add all values of objective functions and save it as V_{ki}.

As mentioned before, the reward function is separable to reward functions of individual users. Also, as seen in Section IV-A, the bandwidth constraint is the main factor of dependence of user actions to each other. Thus, only one round is enough to reach a solution in DaQ algorithm. Note that this is not valid for more complicated reward functions such that multiple parameters make actions to be dependent.

Running speed of DaQ algorithm is also significantly better than the conventional approach i.e., solving the main problem by Algorithm 1. For each user at each round we need to investigate the following cases per iteration

$$O' = |\mathcal{D} \times \mathcal{Q}|^2 |\mathcal{C}|.$$

In each round k, $k = 0, \dots, |\mathcal{U}| - 1$ number of investigated cases is $O_k = (|\mathcal{U}| - k)O'$, and the total number of investigation per round is $O = \sum_k O_k$, which yields

$$O = \frac{1}{2}(|\mathcal{U}|^2 + |\mathcal{U}|)|\mathcal{D} \times \mathcal{Q}|^2 |\mathcal{C}|.$$

Thus, the complexity is a quadratic function of number of users which is much lower than the exponential complexity offered in the optimal algorithm 1. For example for the same setup considered in Section III-C we need about 4.32e3 investigations per iteration which is a reasonable value.

V. Performance results and comparisons

We performed simulations over a network including $|\mathcal{C}| = 3$ cloud sites and $|\mathcal{U}| = 4$ user groups with a geographical coordinates in an square area with 10e6 km^2 shown in Fig. 2. We assume inter node delay τ_{ij} and distance l_{ij} follows the rule $\tau_{ij} = \alpha l_{ij} + \beta$ [1], with $\alpha = 0.02$ and $\beta = 5$ time units. We mark each edge between each cloud site and user group by a delay index $D_{ij} \in \{1, 2, 3\}$ such that D_{ij} is 1 for $\tau < D_1^b$, $D_1^b = 10$, 2 for $D_1^b < \tau < D_2^b$, $D_2^b = 14$ and 3 for $\tau > D_2^b$.

For DP based algorithms we assume $\gamma = 0.9$, $\epsilon = 0.002$. Also, corresponding weights for demand and QoE used in reward function is assumed to be $\mathbf{w}^p := [w_1^p\ w_2^p\ w_3^p\ w_4^p] = [1\ 1\ 1\ 2], \mathbf{w}^q := [w_1^q\ w_2^q\ w_3^q\ w_4^q] = [1\ 1\ 1\ 1]$. We assume the price unit of cloud sites to be $\mathbf{P}^b := [P_1^b\ P_2^b\ P_3^b] = [.2\ .15\ .1]$ and corresponding bandwidths are $\mathbf{B} = [B_1\ B_2\ B_3] = [6\ 6\ 9]$. Four demand levels $\mathcal{D} = \{1,\ 2,\ 3,\ 4\}$ and Three QoE feedback levels $\mathcal{Q} = \{1,\ 2,\ 3\}$ are considered. We assume the same demand transition probability matrix for all users, i.e., $\mathbf{P}_i^d = \mathbf{P}^d$ with probability values $[\mathbf{P}^d]_{(m,n)}, n, m \in \{1, ..., |\mathcal{D}|\}$ are 0.2 for $n = m + 1$, 0.1 for $n = m - 1$, and 0 for $|m - n| \geq 2$. Note that $[\mathbf{P}^d]_{(m,m)} = 1 - \sum_{n \neq m} [\mathbf{P}^d]_{(m,n)}$.

For probability function of QoE feedback in (18) we assume $M = 3$, $\alpha = 0.2$.

In Fig. 3 we plot the accumulated rewards (AR) of DP based algorithms including value iteration (optimum) algorithm, DaQ algorithm and the solution in infinite bandwidth scenario, all normalized by the myopic performance. As it is seen, AR of DaQ performs acceptably close to the optimum algorithm and, for the considered parameters, outperforms 1.03 times higher than myopic AR performance. It means as we proceed during times the gap between ARs of DaQ and myopic are diverged. This 3 percentage gain of DaQ over myopic is affected by different system parameters. In Table I we study the influence of different parameters on the gain offered by DaQ over myopic scheme. We define the improvement AR gap of DaQ over myopic, denoted by $g_{\mathbf{DaQ}}$, as follows

$$g_{\text{DaQ}} = \frac{\text{AR of DaQ}}{\text{AR of myopic}} - 1 \tag{27}$$

In Table I, the 1st row is the result presented in Fig. 3. In the other rows, we only change the parameter presented in the second column and fix the others. As it is seen, increasing $\mathbf{w}^q$, γ and/or decreasing $\mathbf{w}^q$ improve the gain. Also, as we increase dependency of QoE feedback level with the delay of each assigned cloud site at 7th row we improve the gain. However, as we decrease the the system memory at 8th row the gain is decreased. At the last row of the table we applied the changes at rows 3, 5, 7 and, around 15% improvement gain of DaQ over myopic is observed. In Fig. 4 we study the influence of bandwidth constraint on g_{DaQ}. We set a bandwidth for each cloud site and linearly increase all constraint with a same trend. In each trend we can see loosing bandwidth constraint decreases g_{DaQ}. Furthermore, we can see that for some bandwidth constraint $\mathbf{B} = [5\ 9\ 5]$ the gap is significant. This comes from the fact that a highly user-assigned cloud site needs more bandwidth to serve the users. In Fig. 5 we added three more cloud sites with the same bandwidth constraint equal to 5, to the network in Fig. 2, i.e., $|\mathcal{C}| = 6$, $|\mathcal{U}| = 4$, and compare the performance of DaQ with the one in Fig. 3. As it is seen, by increasing the number of cloud sites AR performance improvement of dynamic programming based schemes over conventional myopic ones is increased.

VI. Conclusion

We modeled a vitualized CDN using MDP model, and performed stochastic optimization to assign user groups to surrogate sites for a virtualized CDN. We proposed user groups can contribute in the assignment by QoE feedback they send. We proved that for real size network with more than 10 user groups the conventional solution is not possible due to an exponential complexity w.r.t the number of user groups. Then, we proposed two algorithms which a significantly low complexity. The first one was for a special case of bandwidth unlimited and the second one, referred to DaQ, solves the main problem with a small gap w.r.t the optimal solution in the performance. Complexity analysis made for all algorithms and the effect of different parameters are discussed through simulations.

Acknowledgment

This work was supported by a Collaborative Research and Development Grant from NSERC and Ericsson Canada.

References

[1] M. Hu, J. Luo, Y. Wang, and B. Veeravalli, "Practical resource provisioning and caching with dynamic resilience for cloud-based content distribution networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 8, pp. 2169–2179, Aug. 2014.

[2] C. Papagianni, A. Leivadeas, and S. Papavassiliou, "A cloud-oriented content delivery network paradigm: Modeling and assessment," *IEEE Trans. Dependable Secure Comput.*, vol. 10, no. 5, pp. 287–300, Sep. 2013.

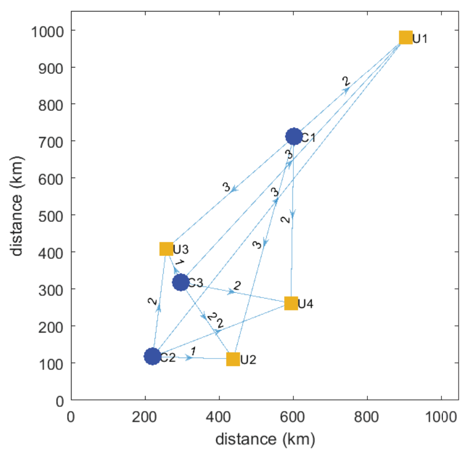

Fig. 2. Network topology with $|\mathcal{C}| = 3$ cloud sites and $|\mathcal{U}| = 4$ user groups).

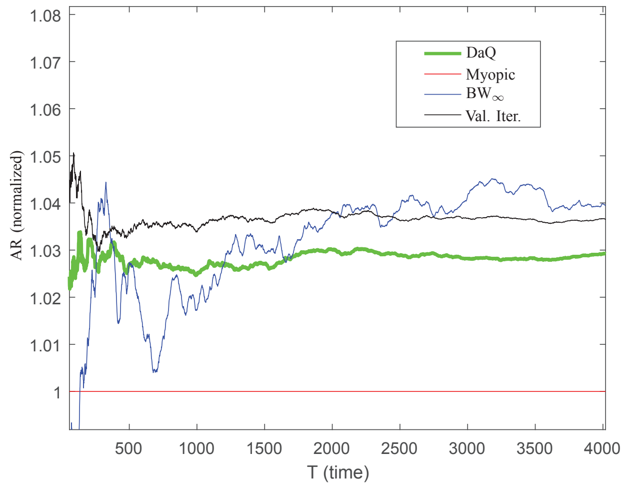

Fig. 3. AR versus time instants for value iteration (optimum) algorithm, DaQ algorithm and infinite bandwidth-constraint scenario, all normalized by the myopic AR performance ($B = [6\ 6\ 9]$, $|\mathcal{C}| = 3$, $|\mathcal{U}| = 4$).

[3] F. Wang, J. Liu, and M. Chen, "CALMS: Cloud-assisted live media streaming for globalized demands with time/region diversities," in *IEEE Int. Conf. Computer Commun. (INFOCOM)*, Mar. 2012, pp. 199–207.

[4] Y. Osais, F. Yu, and M. St-Hilaire, "Optimal management of rechargeable biosensors in temperature-sensitive environments," in *IEEE Veh. Technol. Conf.*, Mar. 2010, pp. 1–5.

[5] M. Nourian, A. S. Leong, and S. Dey, "Optimal energy allocation for Kalman filtering over packet dropping links with imperfect acknowledgments and energy harvesting constraints," *IEEE Trans. Autom. Control*, vol. 59, no. 8, pp. 2128–2143, Aug. 2014.

[6] M. A. Alsheikh, D. T. Hoang, D. Niyato, H. Tan, and S. Lin, "Markov decision processes with applications in wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1239–1267, 2015.

[7] N. Bartolini, F. Lo, and P. Petrioli, "Optimal dynamic replica placement in content delivery networks," in *IEEE Int. Conf. Net.*, 2003, pp. 125–130.

[8] R. Shi, J. Zhang, W. Chu, Q. Bao, X. Jin, C. Gong, Q. Zhu, C. Yu, and S. Rosenberg, "MDP and machine learning-based cost-optimization of dynamic resource allocation for network function virtualization," in *IEEE Int. Conf. Service Comput.*, 2015, pp. 66–73.

[9] W. B. Powell, *Approximate dynamic programming: Solving the curses of dimensionality*, 2nd ed. Wiley, 2011, vol. 9.

[10] S. S. A. Minasian and R. S. Adve, "Energy harvesting cooperative

TABLE I
EFFECT OF DIFFERENT PARAMETERS OVER g_{DaQ}.

-	**Changed parameter**	$g_{\textbf{DaQ}} \times 100\%$
0	No change (Fig. 3)	3
1	$\mathbf{P}^b = [.11\ .10\ .09]$	4
2	$0.1 \times \mathbf{w}^p$	6
3	$10 \times \mathbf{w}^p$	1
4	$0.1 \times \mathbf{w}^q$	0.5
5	$10 \times \mathbf{w}^q$	7
6	$\gamma = 0.1$	1
7	QoE feedback in (18) with $M = 18,\ \alpha = 0.05$	7
8	$[\mathbf{P}^d]_{(m,n)} = \begin{cases} 0.5, & m = 1,4, \lvert m-n\rvert \leq 1, \\ 0.3, & m = 2,3, \lvert m-n\rvert = 1, \\ 0, & \lvert m-n\rvert \geq 2 \end{cases}$	2
9	3 & 5 & 7	15

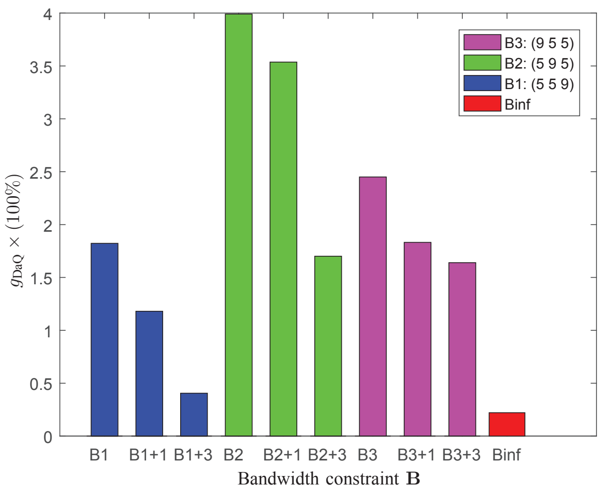

Fig. 4. Improvement gain for the AR of DaQ over myopic for different bandwidth parameters (rewards are accumulated over 4e3 time instants).

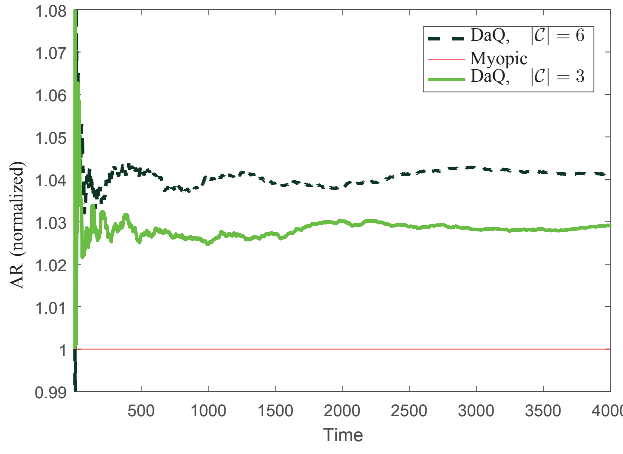

Fig. 5. AR versus time instants in different network sizes for DaQ algorithms normalized by their corresponding myopic performances ($|\mathcal{U}| = 4$).

communication systems," *IEEE Trans. Wireless Commun.*, vol. 13, no. 11, pp. 6118–6131, Nov. 2014.