Online Algorithms for Joint Application-VM-Physical-Machine Auto-Scaling in a Cloud

Yang Guo
Bell Labs, Alcatel-Lucent
Holmdel, NJ 07733
yang.guo@alcatel-lucent.com

Alexander L. Stolyar
Bell Labs, Alcatel-Lucent
Murray Hill, NJ 07974
stolyar@research.bell-labs.com

Anwar Walid
Bell Labs, Alcatel-Lucent
Murray Hill, NJ 07974
anwar@research.bell-labs.com

ABSTRACT

We develop *shadow routing* based online algorithms for the joint problem of application-to-VM and VM-to-PM assignments in a cloud environment. The asymptotic optimality of the shadow algorithm is proved and the performance is evaluated by simulations.

Categories and Subject Descriptors

G.1.6 [**NUMERICAL ANALYSIS**]: Optimization—*Convex programming, Linear programming*; G.3 [**PROBABILITY AND STATISTICS**]: Probabilistic algorithms, Queueing theory

Keywords

Cloud, VM choice and placement, Shadow routing

1. INTRODUCTION

A Cloud data-center typically consists of a large number of physical machines (PMs). Virtualization technology enables cloud service providers to offer customers virtual machines (VMs) to run their applications. VMs can be configured and allocated dynamically to meet demand variations. The cloud service providers may further wish to "pack" the allocated VMs into a smaller number of PMs so as to minimize operational costs and save energy. The design of efficient algorithms for the application-to-VM and VM-to-PM assignments is a challenging research problem.

Considerable prior works exist on cloud resource management with the focus on either the application-to-VM assignments or VM-to-PM assignments, e.g., [1,2,4]. In this paper, we consider the *joint* assignment of applications to VMs and VMs to PMs. In our framework, we explicitly consider (i) the time-varying demand load for applications; (ii) the VM type(s) an application may require; and (iii) the physical machines and their resource configurations.

We employ the *shadow routing* approach, adopting a specially constructed virtual queueing system, which in essence dynamically produces and maintains a solution to the underlying optimization problem that guides the actual assignment algorithm. The advantage of shadow routing approach is that it is simple and adaptive: no need to know a priori, or explicitly measure, the application request arrival rates; if the arrival rates change, the algorithm adapt automatically. Yet, as we show, the algorithm is asymptotically optimal. All these features are confirmed by our simulation experiments.

SIGMETRICS'14, June 16–20, 2014, Austin, Texas, USA.
ACM 978-1-4503-2789-3/14/06.
http://dx.doi.org/10.1145/2591971.2592035.

2. MODEL AND PROBLEM STATEMENT

There are several applications, indexed by $i \in \mathcal{I} = \{1, \ldots, I\}$, and several classes of VMs, indexed by $j \in \mathcal{J} = \{1, \ldots, J\}$. At any given time, each application i employs a group of VMs, denoted by a vector $m_i = (m_{ij}, j = 1, \ldots, J)$ to provide the service, where m_{ij} is the number of VMs of class j assigned to application i. User service requests for application i arrive at the rate λ_i. A service request for application i is assigned to one of the VMs of application i and is serviced by that VM. The average service time of an application i request is $1/\mu_i$. A VM of class j can service an interger number $w_{ij} \geq 0$ of concurrent application i requests.

Each class j VM needs several computing resources of different types when it is served; namely, the amount $a_{jk} > 0$ of resources $k = 1, \ldots, K$. The DC contains $\beta > 0$ physical machines (PM), each of which has the amount $A_k > 0$ of resource $k \in \mathcal{K}_\ell$. If a class j VM is instantialized in the DC, it is placed into one of the PMs, where the amounts a_{jk} of resources are allocated (if they are still available at that specific PM). This means, in particular, that a PM in the DC can simultaneously serve the numbers of different VM class types given by a vector $s = (s_j,\ j = 1, \ldots, J)$ if

$$\sum_i s_j a_{jk} \leq A_k, \quad \forall k \in \mathcal{K}_\ell. \tag{1}$$

Vectors s (with non-negative integer components) satisfying this condition we call *(feasible) configurations* of a PM in the DC. The set of configurations which are not dominated by convex combinations of other configurations is denoted by S.

Denote by $\phi_s \geq 0$ the fraction of PMs in the DC that are used in the configuration $s \in S$. We want an algorithm that allocates fractions ϕ_s, and numbers of VMs, m_{ij}, such that ideally they are close to the optimal solution of the following linear program:

$$\min_{\{m_{ij}\},\{\phi_s\},\rho} \rho, \tag{2}$$

subject to

$$m_{ij} \geq 0, \ \forall (i,j), \quad \phi_s \geq 0, \ \forall s, \tag{3}$$

$$\lambda_i/\mu_i \leq \sum_j w_{ij} \cdot m_{ij}, \quad \forall i, \tag{4}$$

$$\sum_i m_{ij} \leq \sum_{s \in S} s_j \phi_s \beta, \quad \forall j, \tag{5}$$

$$\sum_{s \in S} \phi_s = \rho. \tag{6}$$

Note that in this LP the variables ϕ_s and m_{ij} are non-negative *real* numbers; they have the meaning of *average* values (of the corresponding "true" variables), such that the *average* system workload can be handled. The LHS of (4) is the aggregate (average) workload for application i, while the RHS of (4) is the aggregate (average) service capacity allocated this application. In the constraint (5) $\phi_s \beta$ is the (average) number of PMs used in configuration s and, therefore, the RHS is the (average) number of j-VMs that can be served by PMs in all configurations. The meaning of ρ is the average fraction of the utilized PMs. (The LP makes sense even if the optimal ρ is greater than 1 – if this is the case, the DC is overloaded.)

3. SHADOW ROUTING BASED SCHEME

Shadow algorithm: The algorithm maintains the virtual queues Q_i and V_j, and uses positive parameters c_1, c_2, satisfying

$$c_1 > \max_i 1/\mu_i, \quad c_2 > c_1/\beta. \tag{7}$$

There are additional (small) parameters $\eta > 0$ and $\theta > 0$, and parameter $\alpha > 0$.

Upon each new service request arrival, say of type i to be specific, the algorithm does the following (in sequence):

1. $Q_i(t) := Q_i(t) + 1/\mu_i$.
2. Set $b_{\ell j} := 0$ for all (ℓj). Compute

$$j* \in \arg\max_j [\alpha w_{ij} Q_i - (1/\beta) V_j], \tag{8}$$

and if

$$\alpha w_{ij*} Q_i - (1/\beta) V_{j*} \geq 0 \tag{9}$$

do

$$Q_i := \max\{Q_i - c_1 w_{ij*}, 0\}, \quad V_{j*} := V_{j*} + c_1/\beta, \quad b_{ij*} := 1.$$

3. Set $b_s = 0$ for all $s \in S$. Find configuration vector σ such that

$$\sigma \in \arg\max_{s \in S} \sum_{j \in \mathcal{J}} s_j V_j. \tag{10}$$

If condition

$$\eta \sum_j \sigma_j V_j \geq 1 \tag{11}$$

holds, set $b_\sigma := 1$. Update virtual queues $V_j(t)$ as follows:

$$V_j := \max\{V_j - \sum_{s \in S} s_j b_s, 0\}, \quad \forall j \in \mathcal{J}.$$

4. Update variables $\bar{p}_{\ell j}$ and $\bar{\phi}_s$.

$$\bar{p}_{\ell j} := (1-\theta)\bar{p}_{\ell j} + \theta b_{\ell j}, \quad \forall (\ell, j),$$

$$\bar{\phi}_s := (1-\theta)\bar{\phi}_s + \theta b_s, \quad \forall s.$$

End of Algorithm

The Shadow algorithm is an instance of *Greedy Primal-Dual* (GPD) algorithm [5], applied to a specially constructed two-stage virtual queueing system [3].

PROPOSITION 1. *Suppose all system parameters and all Shadow algorithm parameters, except maybe η, are fixed rational numbers. Assume that condition (7) holds. Suppose the input flows are Poisson, with fixed rates λ_i. Consider a sequence of systems with parameter $\eta \to 0$. Then, for any η, the virtual queueing process is a positive recurrent countable discrete-time Markov chain. Moreover, stationary distributions of the processes are such that the following holds. Denote by $\bar{\phi}_s^{(\eta)}$ the steady-state probability that configuration s is chosen in (10) and condition (11) holds for it. Similarly, let $\bar{p}_{ij}^{(\eta)}$ be the steady-state probability that the arriving request is of type i, index j is chosen in (8) and condition (9) holds. Then, as $\eta \to 0$, the sequence of vectors $(\{\bar{\phi}_s^{(\eta)}\}, \{\bar{p}_{ij}^{(\eta)}\})$ is such that its any limiting point $(\{\bar{\phi}_s\}, \{\bar{p}_{ij}\})$ satisfies (using notation $\lambda = \sum_i \lambda_i$)*

$$\lambda c_1 \bar{p}_{ij} = m_{ij}, \quad \forall (i,j),$$

$$\lambda c_2 \bar{\phi}_s = \phi_s, \quad \forall \ s \in S,$$

where vector $(\{\phi_s\}, \{m_{ij}\})$ is an optimal solution of LP (2)-(6).

Proposition 1 shows that the values of $\bar{\phi}_s$ and $\bar{p}_{ij}$ dynamically "maintained" by the Shadow algorithm are proportional to the optimal values ϕ_s and m_{ij}, respectively. This allows us to devise very simple dynamic algorithms [3] for assignining an arriving applications to VM types, and assigning VMs to PMs; both algorithms have access to the current values of $\bar{\phi}_s$ and $\bar{p}_{ij}$.

The evaluation of Shadow algorithm in realistic scenarios is presented in [3]. (There we also show how to set the algorithm parameters.) The evaluations show very good accuracy of the algorithm as well as its great robustness and adaptability.

4. REFERENCES

[1] Auto Scaling - Amazon Web Services, *http://aws.amazon.com/documentation/autoscaling/*

[2] Y. Guo, A. L. Stolyar, A. Walid. Shadow-routing based dynamic algorithms for Virtual Machine placement in a network cloud. *INFOCOM-2013*.

[3] Y. Guo, A. L. Stolyar, A. Walid. Online Algorithms for Joint Application-VM-Physical-Machine Auto-Scaling in a Cloud. Bell Labs Technical Memo, 2014. http://ect.bell-labs.com/who/stolyar/publications/gsw-sigm-14-full.pdf

[4] T. Lorido-Botran, J. Miguel-Alonso and J. Lozano. Auto-scaling Techniques for Elastic Applications in Cloud Environments. Technical Report EHU-KAT-IK-09-12, University of the Basque Country, 2012.

[5] A. L. Stolyar. Maximizing Queueing Network Utility subject to Stability: Greedy Primal-Dual Algorithm. *Queueing Systems*, Vol. 50, pp. 401-457 (2005).