Consolidating Virtual Machines with Dynamic Bandwidth Demand in Data Centers

Meng Wang*, Xiaoqiao Meng†, and Li Zhang†
* School of ECE, Cornell University, Ithaca, NY 14853, USA. Email: mw467@cornell.edu
†IBM T.J. Watson Research Center. 19 Skyline Drive, Hawthorne, NY 10532, USA. Email: {xmeng,zhangli}@us.ibm.com

***Abstract*—Recent advances in virtualization technology have made it a common practice to consolidate virtual machines(VMs) into a fewer number of servers. An efficient consolidation scheme requires that VMs are packed tightly, yet receiving resources commensurate with their demands. On the other hand, measurements from production data centers show that the network bandwidth demands of VMs are dynamic, making it difficult to characterize the demands by a fixed value and to apply traditional consolidation schemes. In this work, we capture VM bandwidth demand by random variables following probabilistic distributions. We study how VMs should be consolidated with bandwidth limit imposed by network devices such as Ethernet adapters and edge switches. We formulate a Stochastic Bin Packing problem and propose an online packing algorithm by which the number of servers required is within $(1+\epsilon)(\sqrt{2}+1)$ of the optimum for any $\epsilon > 0$. In a special case that there are finite number of possibilities for the means of the random variables, the number of servers required by our algorithm is within $(\sqrt{2}+1)$ of the optimum. In addition, we use numerical experiments to evaluate the proposed consolidation algorithm and observe 30% server reduction compared to several benchmark algorithms.**

I. Introduction

Virtualization technologies promise opportunities for modern data centers to host applications on shared infrastructure. Now data center operators can create a large number of virtual machines (VMs) for different workload requests. Each VM is provisioned with certain amount of computing resources commensurate with workload requirements. The operators can then consolidate all the VMs into a smaller number of physical servers, with the goal of minimizing the total required server number. Such a VM consolidation procedure is the key factor for a data center to achieve economy of scale.

Traditional VM consolidation schemes are concerned with CPU, memory and disk I/O [13][23][26][29][31][32]. In these schemes, the operators need to estimate a *deterministic* amount of resources required by the future workload; such an estimate is usually made from either an understanding of the underlying applications or a forecast using historical workload patterns. The actual amount of allocated resources to VMs are based on the estimate. In the consolidation phase, all such VMs with fixed size must be packed on servers with known capacity limit. In this way, the consolidation phase becomes the classical bin packing problem and can be solved by many heuristics.

Compared to the capacity limit of CPU, memory and disk I/O, network bandwidth limit has not been well studied. In fact, with an explosive growth of data center traffic, network bandwidth limit becomes more and more critical, and such limit exists at multiple consolidation levels. E.g., at the server level, due to the shared Ethernet adapters, the aggregate traffic rate for VMs placed on the same server must not exceed the adapter capacity. At the chassis/rack level, the aggregate rate must not exceed the capacity of the end-of-row/top-of-rack switch. Thus it is imperative to find efficient consolidation methods that respect the limit imposed by various network devices.

One can easily apply the traditional VM consolidation schemes to handle network bandwidth limit. Nevertheless, this might raise a fundamental drawback as recent measurement studies show that, in production data centers, network traffic are highly volatile and bursty[2][19][27]. It thus becomes difficult for the traditional schemes to make a reliable, deterministic estimate of bandwidth demand. Although one can take a conservative strategy by making an estimate much higher than the actual traffic rate, it leads to over-provisioning and resource waste.

In this work, instead of using deterministic values, we use random variables to characterize the future bandwidth usage. The random variables follow certain distributions that are estimated from either historical traffic rates or forecasting algorithms. Such a probabilistic characterization can better represent the uncertainty of the future bandwidth demand. With VM size being a random variable, the VM consolidation issue is formulated into a NP-hard Stochastic Bin Packing problem (SBP), which states that items with sizes following a probabilistic distribution must be packed into bins such that the number of bins used is minimal and the chance for violating any bin size is below a threshold. Similar problems are studied in [11][22], assuming the random variables all have Bernoulli-type distributions, Poisson distributions, or exponential distributions. In our work, all random variables have Gaussian distributions, and we propose an approximation algorithm and prove its asymptotical worst-case performance bound. In our algorithm, random variables are divided into groups according to their means and variances. The random variables are then assigned to different bins based on which groups they belong to. We prove the worst-case performance ratio of our algorithm is within $(1+\epsilon)(\sqrt{2}+1)$ for any $\epsilon > 0$. In a special case where there are finite number of possibilities of the mean value, the worst-case performance ratio is within $\sqrt{2}+1$. We run numerical experiments to evaluate performance of the proposed algorithm. The results show that our algorithm saves up to 30% of required servers compared with several

benchmark schemes.

The rest of the paper is organized as follows. Section II discusses the related work. Section III elaborates on problem formulation. Section IV and V present algorithms and bound analysis for a special case and generic cases respectively. Section VI considers a more general case with variable-sized bins. Section VII uses numerical experiments to evaluate the proposed algorithms. Section VIII concludes the paper.

II. Related work

A. Virtual Machine Consolidation

VM consolidation in Compute Clouds has been tackled in both commercial products and research work. Representative products include VMWare's Distributed Resource Scheduler (DRS) [33] and IBM Server Planning Tool (SPT) [12]. These tools use server utilization data to dynamically assign resources to servers based on various policies. In the research literature, most work [1][18][26][28][34] formulate VM consolidation as an optimization problem with the number of required servers as the objective. Such an optimization problem is often associated with constraints imposed by server capacity, service-level-agreement, legal, etc. To solve the optimization problem, these work use various algorithms with a nature similar to these bin packing heuristics such as First Fit Decreasing (FFD) [1].

While these products and research work consider CPU, memory or disk, they do not take into account the network capacity limit imposed by network devices. On the methodology aspect, these work denote the resource demand of VMs by a deterministic value. In contrast, in this work we model the VM resource demand as a random variable which better captures the dynamics of bandwidth usage in data centers.

B. Bin Packing

In the classical bin packing problem, a list L with real numbers between 0 and 1 must be assigned to bins with unit capacity such that the sum of the numbers in each bin is no more than 1 and the number of bins used is minimal. Let OPT denote the minimum. For a packing algorithm B, let $B(L)$ denote the number of bins used for L, then the worst case performance ratio of this algorithm is defined as $\lim_{OPT\to\infty} \sup_L [B(L)/OPT]$. Since finding the optimal packing involves solving an NP-complete partition problem [10][20], various heuristic algorithms [4][14][15][16][35] with performance guarantee have been proposed. One simple algorithm First Fit Decreasing (FFD) has a worst case performance ratio of $\frac{11}{9}$ [10][36], and is later shown to be tight [7]. MFFD [17], a variance of FFD can have a ratio of $\frac{71}{60}$.

These algorithms require to first sort the items, and are not applicable for an online bin packing problem. An online algorithm is required to process L in the same order as given. First Fit [10] is an simple online algorithm with a performance ratio of $\frac{17}{10}$, and a refined version [35] has a ratio of $\frac{5}{3}$. Next Fit [4] is another popular online algorithm which has a worst case performance ratio of 2 and takes $O(n)$ time and $O(1)$ space. First Fit (FF), on the other hand, takes $O(n \log n)$ time and $O(n)$ space. [24] proposes a HARMONIC algorithm which has a worst case performance ratio of 1.692. [35] shows that no online algorithm has a performance ratio less than $\frac{3}{2}$, and this lower bound is later improved to 1.53635 [6][25].

Our problem formulation is the same as the Stochastic Bin Problem in [11][22]. Given a positive constant α, the goal is to use a minimum number of bins to pack the items such that the violation probability of each bin is at most α. [22] consider the case that item sizes have a Bernoulli-type distribution, and the worst-case performance ratio of their algorithm is $O(\sqrt{\frac{\log p^{-1}}{\log\log p^{-1}}})$. [11] provide a polynomial time approximation algorithm for the cases that item sizes have Poisson or exponential distributions, a quasi-polynomial approximation scheme with running time polynomially on n and $\log 1/p$ for Bernoulli-distributed items.

III. Problem Formulation

In this section, we present a model of VM consolidation with network bandwidth constraint imposed by network devices. We first introdece the Stochastic Bin Packing (SBP) in Section III-A. We then compare SBP to the classical bin packing problem in Section III-B and illustrate its special properties in Section III-C.

A. Stochastic Bin Packing

We consider a scenario in which n VMs with known bandwidth demands must be consolidated onto a number of servers (or chassis, racks). We assume an online consolidation senario that each VM should be consolidated once a request is made. We assume servers have identical capacity limit. This corresponds to the common case that server configurations are homogeneous in the same data center. Heterogeneous cases will be discussed in Section VI. Let x_i denote the normalized bandwidth demand of VM i, the n-VM consolidation problem is to pack a list of items $L = (x_1, x_2, ..., x_n)$ into bins with unit capacity. Instead of assuming x_i a fixed value in the classical bin packing problem, here we assume x_i follows a probabilistic distribution. The distributions of the sizes of different items i and j are independent, and the probability distribution of their total size $x_i + x_j$ is the convolution of the two probability distributions of x_i and x_j. In the classical bin packing problem, a packing strategy is feasible if for each bin, the total item size in that bin does not exceed one. Here we define a packing strategy *feasible* if for each bin, the probability that the unit capacity is exceeded is no greater than a given constant $\alpha \in (0, 1)$. Our goal is to find a feasible packing strategy that uses the least number of bins. We formally describe the Stochastic Bin Packing problem as following:

Given a list of items $L = (x_1, x_2, ..., x_n)$, where x_i's are independent random variables, what is the minimum number of unit capacity bins needed to pack all the items, such that for each bin, the probability that its capacity is exceeded is no greater than a given constant $\alpha \in (0, 1)$?

Different from previous assumptions that x_i follows a Bernoulli distribution, a Poisson distribution or an exponential

distribution [11][22], in this paper we assume that x_i independently follows a normal distribution $\mathcal{N}(\mu_i, \sigma_i^2)$. In reality server consolidation usually occurs at weekly or monthly timescale. At such a large timescale, we believe the VM bandwidth usage can be approximated by a Gaussian distribution. We assume the mean μ_i is positive and the standard deviation σ_i is small enough compared with μ_i, thus the probability of x_i being negative is very small and negligible.

If $\sigma_i = 0$ for all i, then $x_i = \mu_i$ and the problem is reduced to the classical NP-hard one-dimensional bin packing problem. Clearly SBP is NP-hard.

For a packing strategy, let U^j denote the set of indices of the items that are packed in bin j. Since $x_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ and x_i and x_j are independent if $i \neq j$, then the total size of the items in a bin follows normal distribution with mean $\sum_{i\in U^j} \mu_i$ and variance $\sum_{i\in U^j} \sigma_i^2$. For the normal distribution, we have the following fact,

Lemma 1. *Given $x \sim \mathcal{N}(\mu, \sigma^2)$ with $\mu \in (0,1)$ and $\alpha \in (0,1)$, then Prob$[x > 1] \leq \alpha$ holds if and only if $\mu + \Phi^{-1}(1-\alpha)\sigma \leq 1$, where the a **quantile function** Φ^{-1} is the inverse of the cdf Φ of $\mathcal{N}(0,1)$.*

Proof: Since $x \sim \mathcal{N}(\mu, \sigma^2)$, then Prob$[x > \mu + \Phi^{-1}(1-\alpha)\sigma] = \alpha$ from definition. Thus Prob$[x > 1] \leq \alpha$ if and only if $\mu + \Phi^{-1}(1-\alpha)\sigma \leq 1$. ■

Given the violation probability α, let $\beta := \Phi^{-1}(1-\alpha)$. Thus, from Lemma 1, we say a packing strategy is *feasible* for a given α if and only if $\sum_{i\in U^j} \mu_i + \beta\sqrt{\sum_{i\in U^j} \sigma_i^2} \leq 1$ for all bin j. In this paper, we only consider the case that $\alpha \leq 0.5$ and $\beta \geq 0$. In order to make sure that there exists a feasible packing for a given α, throughout this paper we assume that each item should be able to be packed into one bin, i.e. $\mu_i + \beta\sigma_i \leq 1, \quad \forall i$.

B. SBP v.s. Deterministic Packing

Because $\sum_{i\in U^j} \mu_i + \beta\sqrt{\sum_{i\in U^j} \sigma_i^2} \leq \sum_{i\in U^j} (\mu_i + \beta\sigma_i)$, if we use constant $\mu_i + \beta\sigma_i$ as the size of item i, and reduce the problem to the classical bin packing, obviously any feasible packing strategy for the classical bin packing is also feasible for SBP. Therefore,

Observation 1. *The optimal number of bins used in SBP is no greater than the optimum of a classical bin packing problem with $\mu_i + \beta\sigma_i$ as the size for item i.*

Since SBP only requires $\mu_i + \mu_j + \beta\sqrt{\sigma_i^2 + \sigma_j^2} \leq 1$ when item i and item j are packed in one bin, the required bins by simply fitting into the classical bin packing problem may be much larger than the optimum of SBP. We provide an example to illustrate this difference.

Example 1 Let $n = 160$ and $x_i \overset{i.i.d.}{\sim} \mathcal{N}(\frac{2}{41}, (\frac{3}{164})^2)$ for all i. $\alpha = 0.0013$, and $\beta = \Phi^{-1}(1-\alpha) = 3$.

Since $\mu_i = \frac{2}{41}$, $\sigma_i = \frac{3}{164}$ for all i, and $16 \times \frac{2}{41} + 3 \times \sqrt{16} \times \frac{3}{164} = 1$, then each bin can pack exactly 16 items. For 160 items, the optimal strategy is to pack them into 10 bins.

If we use $\mu_i + \beta\sigma_i$ as the size of item i and treat it as a classical bin packing problem, since $\frac{2}{41} + 3 \times \frac{3}{164} = \frac{17}{164}$, then each bin can pack at most 9 items. Thus, the best strategy is to pack 9 items in each bin, which results in using 18 bins.

In general, if we directly fit into the classical bin packing problem with sizes $\mu_i + \beta\sigma_i$, even if we could solve the NP-hard problem and find the minimum bin number, this value might still be much larger than the number of bins that are indeed needed for SBP.

C. Equivalent Sizes Vary under Different Packing

In fact, for items that can be packed in one bin with U as the set of indices, we have

$$1 \geq \sum_{i\in U} \mu_i + \beta\sqrt{\sum_{i\in U} \sigma_i^2} = \sum_{i\in U} \left(\mu_i + \frac{(\beta\sigma_i)^2}{\beta\sqrt{\sum_{i\in U} \sigma_i^2}} \right). \quad (1)$$

Thus, we can view $\mu_i + \frac{(\beta\sigma_i)^2}{\beta\sqrt{\sum_{i\in U} \sigma_i^2}}$ as the "*equivalent size*" of item i, which in general depends on other items packed together. In a special case that $\sigma_i = 0$ or $\beta = 0$, the equivalent size reduces to μ_i, which does not depend on the size of other items and coincides with the classical case. However, when $\sigma_i \neq 0$ and $\beta \neq 0$, the equivalent size changes if the items that are packed in the same bin changes. For example, if item i is the only one in a bin, then its equivalent size achieves the maximum, which is $\mu_i + \beta\sigma_i$. If item i is packed in the same bin with some item having a non-zero variance, then the equivalent size is strictly less than $\mu_i + \beta\sigma_i$.

Note that in the classical bin packing problem, the number of bins used is indeed the sum of the item sizes plus the sum of the residual capacity of each bin. Since the item size is fixed, we only need to pack each bin in a compact way so as to reduce the residual capacity of each bin. However, in our problem setup, reducing the residual capacity does not necessarily reduce the number of bins used since the equivalent size can change. Let us first consider a simple example.

Example 2. Let $n = 35$, $\alpha = 0.0013$, $\beta = 3$. There are three types of items:(1) $x_i \overset{i.i.d.}{\sim} \mathcal{N}(1/6, 1/324)$, $\forall i = 1, ..., 4$, (2) $x_i \overset{i.i.d.}{\sim} \mathcal{N}(1/12, 1/1296)$, $\forall i = 5, ..., 13$, and (3) $x_i = 1/3 - \sqrt{3}/6$, $\forall i = 14, ..., 35$.

Since $4 \times \frac{1}{6} + 3 \times \sqrt{4} \times \frac{1}{18} = 1$, then the first four items can fit into one bin. One can check that the next nine items can exactly fit into one bin. Since $22 \times (\frac{1}{3} - \frac{\sqrt{3}}{6}) \approx 0.9825 < 1$, then the rest twenty-two items can be packed in one bin. This strategy uses three bins and there is no residual capacity in two bins. One can also check that this strategy is also optimal since that there is no way to pack all the items into two bins.

If we pack two items in the first type and four items in the second type and one item in the third type together, one can check that they can just fit into one bin with no residual capacity. If two bins are packed in this way, then we have one item in the second type and twenty items in the third type left, which cannot fit into one bin. Therefore, even though the first two bins are fully packed, we need four bins in this strategy.

Algorithm 1 Group packing algorithm for items with means choosing from a finite set

Initial: $t_{kh} = 0$ for all k,h

1 Each item i ($i = 1, ..., n$) belongs to a group G_{kh} for some $k \in \{1, ..., m\}$, $h \in \{1, ..., W_k\}$

2 If item i fits into the current G_{kh} bin, the pack it.

3 Otherwise, $t_{kh} = t_{kh} + 1$, open a new G_{kh} bin to pack item i, and make it the current G_{kh} bin.

As shown in the example, it is the variation of equivalent size that leads to different packing results. Since the equivalent size is larger in the second strategy than the first one, even though there is no residual capacity, we cannot fit them into three bins. Therefore, in order to reduce the number of bins, we need to (1) reduce the residual capacity and (2) reduce the equivalent size at the same time while packing items.

Since the equivalent size of item i depends on other items, we provide its upper bound and lower bound that only depend on μ_i and σ_i. From (1), we have $\beta\sqrt{\sum_{i\in U}\sigma_i^2} \leq 1 - \sum_{i\in U}\mu_i \leq 1-\mu_i$, therefore, $\mu_i + \frac{(\beta\sigma_i)^2}{\beta\sqrt{\sum_{i\in U}\sigma_i^2}} \geq \mu_i + \frac{(\beta\sigma_i)^2}{1-\mu_i}$, where the righthand side only depends on μ_i and σ_i. Let

$$g(\mu, \sigma) := \mu + (\beta\sigma)^2/(1-\mu), \tag{2}$$

then we have

$$g(\mu_i, \sigma_i) \leq \mu_i + \frac{(\beta\sigma_i)^2}{\beta\sqrt{\sum_{i\in U}\sigma_i^2}} \leq \mu_i + \beta\sigma_i. \tag{3}$$

In later discussion, we will use the two bounds in (3) to characterize the equivalent size of item i.

IV. A Special Case: finite possibilities of means

In this section we consider a special case that the means of all items are from m distinct values e_1, e_2,...,e_m with $e_k \in (\eta, 1)$ ($\eta > 0$). While m is a constant, n, the number of items, can grow to infinity. We will provide an online packing algorithm, and prove that it can achieve a worse case performance ratio of $\sqrt{2}+1$. The result is slightly better than in the general scenario in Section V. Besides, the algorithm and the analysis technique used here are the basis for the general scenario.

We first describe the algorithm (see Algorithm 1). In the algorithm, we divide items into different groups according to their means and variances, we then pack the items in the same group by Next Fit [4]. Specifically, we keep one active bin for each group. If the current item cannot fit into the active bin, we open a new bin and make it active. For items with the same mean e_k ($k = 1, ..., m$), the standard deviation is no greater than $(1 - e_k)/\beta$ from previous assumptions. We divide them into $W_k (= \lfloor 1/e_k \rfloor \leq \lfloor 1/\eta \rfloor)$ groups according to their variances. If $\mu_i = e_k$, and $\sigma_i \in (\frac{1-(h+1)e_k}{\beta\sqrt{h+1}}, \frac{1-he_k}{\beta\sqrt{h}}]$ ($h = 1, ..., W_k - 1$), then we say item i belongs to group G_{kh}. If $\mu_i = e_k$, and $\sigma_i \in [0, \frac{1-W_k e_k}{\beta\sqrt{W_k}}]$, then item i belongs to group G_{kW_k}.

Let t_{kh} be the number of bins used by Algorithm 1 to pack the items in group G_{kh}, then the total number of bins used is $B := \sum_{k=1}^{m}\sum_{h=1}^{W_k} t_{kh}$. Since the item classification can be done in $O(\log(m\lfloor 1/\eta \rfloor))$ time, Algorithm 1 runs in $O(n\log(m\lfloor 1/\eta \rfloor))$ time. For given m and η, it takes $O(n)$ time. Since there are at most $m\lfloor 1/\eta \rfloor$ active bins at any time, the algorithm takes $m\lfloor 1/\eta \rfloor$ storage space to store active bins.

Now we derive the worst case performance ratio of Algorithm 1. If $W_k \geq 2$, note that since for each item i that belongs to G_{kh} ($h = 1, ..., W_k - 1$) we have $\mu_i = e_k$, and $\sigma_i \in (\frac{1-(h+1)e_k}{\beta\sqrt{h+1}}, \frac{1-he_k}{\beta\sqrt{h}}]$, then the total size of the items in one bin for group G_{kh} follows normal distribution with mean $\mu = de_k$ and variance $\sigma \in (\frac{\sqrt{d}(1-(h+1)e_k)}{\beta\sqrt{h+1}}, \frac{\sqrt{d}(1-he_k)}{\beta\sqrt{h}}]$, where d is the number of items in a bin. From Lemma 1, for a feasible packing with violation probability α, we want $\mu + \beta\sigma \leq 1$. If $d = h$, we have $\mu + \beta\sigma \leq he_k + (1 - he_k) = 1$, and if $d = h+1$, we have $\mu+\beta\sigma > (h+1)e_k+(1-(h+1)e_k) = 1$. Therefore there are exactly h items in each bin except the last one for group G_{kh} ($h = 1, ..., W_k-1$). For group G_{kW_k}, since each item has mean e_k and variance no greater than $\frac{1-W_k e_k}{\beta\sqrt{W_k}}$, then there are at least W_k items in each bin except the last one. Thus, we have the following key observation,

Observation 2. *In the resulting packing strategy of Algorithm 1, there are exactly h items in each bin except the last one of group G_{kh} ($h = 1, ..., W_k - 1$), and at least W_k items in each bin except the last one of group G_{kW_k} for $K = 1, ..., m$.*

From Observation 2, in the resulting packing strategy, for group G_{kh} ($k = 1, ..., m$, $h = 1, ..., W_k-1$), each item (except those in the last bin) takes up exactly $1/h$ of a bin. And for group G_{kW_k}, each item (except those in the last bin) takes up at most $1/W_k$ of a bin. Thus, like in [24] we can define an **effective occupation** $f(\mu, \sigma)$ as follows

$$f(\mu, \sigma) = \frac{1}{h} \quad \text{if} \quad \frac{1-(h+1)\mu}{\beta\sqrt{h+1}} < \sigma \leq \frac{1-h\mu}{\beta\sqrt{h}} \quad \text{for some } h \in \{1, ..., \lfloor 1/\mu \rfloor\} \tag{4}$$

$f(\mu_i, \sigma_i)$ can be roughly interpreted as the fraction of a bin that is occupied by each item i in the packing strategy of Algorithm 1. In fact, the effective occupation $f(\mu, \sigma)$ is very important for the performance analysis of the algorithms in this paper. Given violation probability α, let OPT denote the number of bins used by the optimal strategy. Define $\mathcal{S}$ to be the set of all the possible item subsets that can fit into one bin such that the violation probability for the bin capacity is at most α, i.e.,

$$\mathcal{S} := \{U \subset \{1, 2, ..., n\} \mid \sum_{i\in U}\mu_i + \beta\sqrt{\sum_{i\in U}\sigma_i^2} \leq 1\}. \tag{5}$$

Now we can state a general lemma which will be used both here and in Section V for algorithm analysis.

Lemma 2. *Given α, a list of items $L = (x_1, x_2, ..., x_n)$ with $x_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ are packed into B bins. Let U^j be the set of indices of items in bin j. If there exists function f such that*

for every bin j except Q bins (Q does not depend on n),

$$\sum_{i\in U^j} f(\mu_i,\sigma_i) \geq 1, \tag{6}$$

and let $r^* := \max_{U\in S}\sum_{i\in U} f(\mu_i,\sigma_i)$, where $\mathcal{S}$ is defined in (5), then we have

$$B \leq r^* OPT + Q. \tag{7}$$

Moreover, when OPT go to infinity,

$$\lim_{OPT\to\infty} B/OPT \leq r^* \tag{8}$$

Proof: Since (6) holds for $B-Q$ bins, then $B-Q \leq \sum_{i=1}^n f(\mu_i,\sigma_i)$. It follows from the definition of r^* that $\sum_{i=1}^n f(\mu_i,\sigma_i) \leq r^* OPT$. Then we have (7) by combing the above two inequalities. (8) follows if we let both sides of (7) divide OPT and let OPT go to infinity. ■

Since it is hard to directly compute r^*, we will next give an upper bound which is also an upper bound for the worst-case performance ratio of Algorithm 1. Since $\sum_{i\in U}\mu_i + \beta\sqrt{\sum_{i\in U}\sigma_i^2} \leq 1$ for every U in $\mathcal{S}$, we know

$$\begin{aligned} r^* &\leq \max_{U\in S} \frac{\sum_{i\in U} f(\mu_i,\sigma_i)}{\sum\limits_{i\in U}\mu_i + \beta\sqrt{\sum_{i\in U}\sigma_i^2}} \leq \max_{U\in S}\frac{\sum_{i\in U} f(\mu_i,\sigma_i)}{\sum_{i\in U} g(\mu_i,\sigma_i)} \\ &\leq \max_{U\in S}\max_{i\in U}\frac{f(\mu_i,\sigma_i)}{g(\mu_i,\sigma_i)} = \max_i \frac{f(\mu_i,\sigma_i)}{g(\mu_i,\sigma_i)}, \end{aligned} \tag{9}$$

where g is defined in (2), the second inequality follows from (3) and the third holds since f and g are always positive.

Now consider the effective occupation f defined in (4). For group G_{kh} ($k=1,...,m$, $h=1,...,W_k-1$), Algorithm 1 uses t_{kh} bins, and for each bin j except the last one, one can easily check that $\sum_{i\in U^j} f(\mu_i,\sigma_i)=1$. In group G_{kW_k}, we have that $\sum_{i\in U^j} f(\mu_i,\sigma_i)\geq 1$ for each bin $j\in\{1,..,t_{kW_k}-1\}$. Thus, the assumption of Lemma 2 is satisfied with $Q\leq\sum_{k=1}^m W_k$, where $\sum_{k=1}^m W_k$ accounts for the total number of possibly unfilled bins in all the groups. Therefore, from Lemma 2 and (9), and $e_k\geq\eta\ \forall k$, we have

$$B \leq r^* OPT + \sum_{k=1}^m W_k \leq \max_i \frac{f(\mu_i,\sigma_i)}{g(\mu_i,\sigma_i)} OPT + m\lfloor\frac{1}{\eta}\rfloor. \tag{10}$$

Now the question is how to estimate $\max_i \frac{f(\mu_i,\sigma_i)}{g(\mu_i,\sigma_i)}$. We state an important lemma for $\frac{f(\mu,\sigma)}{g(\mu,\sigma)}$.

Lemma 3. *For all $h=1,...,\lfloor 1/\mu\rfloor$, we have*

$$\max_{\mu\in(0,1),\sigma\in[0,\frac{1-h\mu}{\beta\sqrt{h}}]} \frac{f(\mu,\sigma)}{g(\mu,\sigma)} < \lambda(h) := \left(2h^2(\sqrt{\frac{h}{h+1}}-1)+h\right)^{-1}, \tag{11}$$

where f and g are defined in (4) and (2), and $\lambda(h)$ strictly decreases as h increases. Specially,

$$\max_{\mu+\beta\sigma\leq 1}\left(f(\mu,\sigma)/g(\mu,\sigma)\right) < \sqrt{2}+1. \tag{12}$$

Proof: We study $f(\mu,\sigma)/g(\mu,\sigma)$ in three different cases.

(i) If $\mu>1/2$, then $f(\mu,\sigma)=1$, and $g(\mu,\sigma)=\mu+\sigma^2/(1-\mu)>1/2$. Therefore, $f(\mu,\sigma)/g(\mu,\sigma)<2$.

(ii) If $\mu\leq 1/2$ and $\sigma\in(\frac{1-(h+1)\mu}{\beta\sqrt{h+1}},\frac{1-h\mu}{\beta\sqrt{h}}]$ for some h belongs to $\{1,2,...,\lfloor 1/\mu\rfloor-1\}$, then $f(\mu,\sigma)=1/h$. Therefore,

$$\begin{aligned} \frac{f(\mu,\sigma)}{g(\mu,\sigma)} &< \left(h(\mu+\frac{(1-(h+1)\mu)^2}{(h+1)(1-\mu)})\right)^{-1} \\ &= \left(h^2(1-\mu)+\frac{h^3}{(h+1)(1-\mu)}+h-2h^2\right)^{-1} \\ &\leq (2h^2(\sqrt{h/(h+1)}-1)+h)^{-1}, \end{aligned} \tag{13}$$

where the last inequality holds from the fact that $h^2(1-\mu)+\frac{h^3}{(h+1)(1-\mu)}\geq 2h^2\sqrt{\frac{h}{h+1}}$, and the equality holds when $\mu=1-\sqrt{h/(h+1)}$. Therefore, for $h=2,...,\lfloor 1/\mu\rfloor$, we have

$$\max_{\mu}\max_{\sigma\in(\frac{1-(h+1)\mu}{\beta\sqrt{h+1}},\frac{1-h\mu}{\beta\sqrt{h}}]}\left(f(\mu,\sigma)/g(\mu,\sigma)\right) < \lambda(h).$$

One can check that $\lambda(h)>0$ and $\lambda'(h)<0$ for all h. Then $\lambda(h)$ strictly decreases as h increases. When $h=1$, $\lambda(h)$ achieves the maximum value $\sqrt{2}+1$. When h goes to infinity, $\lambda(h)$ goes to $\frac{4}{3}$.

(iii) If $\mu\leq 1/2$, and $\sigma\leq\frac{1-\lfloor 1/\mu\rfloor\mu}{\beta\sqrt{\lfloor 1/\mu\rfloor}}$, then $f(\mu,\sigma)=1/\lfloor 1/\mu\rfloor$, and $g(\mu,\sigma)\geq\mu$. Therefore, $\frac{f(\mu,\sigma)}{g(\mu,\sigma)}\leq\frac{1}{\mu\lfloor 1/\mu\rfloor}$.

Let $D:=\lfloor 1/\mu\rfloor$, then $D\geq 2$. Besides, $1-\mu<D\mu\leq 1$. Thus, $1/D\geq\mu$, and $0\leq 1-D\mu<\mu$. We claim that

$$D\mu > (D-1)^2(1-\mu)+\frac{(D-1)^3}{D(1-\mu)}+(D-1)-2(D-1)^2. \tag{14}$$

Since the righthand side of (14) minus its lefthand side is

$$\begin{aligned} & ((D-1)\mu^2-(1-D\mu)^2)/(1-\mu)+(1/D-\mu) \\ >\ & ((D-1)\mu^2-\mu^2)/(1-\mu)+(1/D-\mu) \\ =\ & (D-2)\mu^2/(1-\mu)+(1/D-\mu)\geq 0, \end{aligned}$$

where the first inequality holds from $1-D\mu<\mu$, and the second inequality holds from $D\geq 2$, $\mu\leq 1$ and $1/D\geq\mu$. Therefore (14) holds. From (14) we have

$$\begin{aligned} D\mu >\ & 2\sqrt{\tfrac{(D-1)^2(1-\mu)(D-1)^3}{D(1-\mu)}}+(D-1)-2(D-1)^2 \\ =\ & 2(D-1)^2(\sqrt{(D-1)/D}-1)+D-1. \end{aligned}$$

Thus,

$$\frac{f(\mu,\sigma)}{g(\mu,\sigma)}\leq\frac{1}{\mu\lfloor 1/\mu\rfloor} < \left(2(D-1)^2(\sqrt{(D-1)/D}-1)+D-1\right)^{-1}.$$

Combing cases (i), (ii) and (iii), we conclude that

$$\max_{\mu\in(0,1),\sigma\in[0,\frac{1-h\mu}{\beta\sqrt{h}}]}(f(\mu,\sigma)/g(\mu,\sigma)) < \lambda(h),$$

where $h=1,...,\lfloor 1/\mu\rfloor$. When $h=1$, we get (12). ■

Now we are ready to present our first main result.

Theorem 1. *Given a finite set $\{e_1,...,e_m\}$ with $e_k\in(\eta,1)$ ($\eta>0$) for all $k=1,...,m$ and a violation probability $\alpha\leq 0.5$, a list $L=(x_1,...,x_n)$ with $x_i\sim\mathcal{N}(\mu_i,\sigma_i^2)$ and $\mu_i\in\{e_1,...,e_m\}$ for all $i=1,...,n$ needs to be packed into the*

least number of bins while for each bin, the probability that its capacity is exceeded does not exceed α. Then the number of bins B used by Algorithm 1 to pack L satisfies

$$B < (\sqrt{2}+1)OPT + m\lfloor 1/\eta \rfloor, \tag{15}$$

where OPT is the minimum number of bins needed. And the worst-case performance ratio satisfies

$$\lim_{n\to\infty} B/OPT \le \sqrt{2}+1. \tag{16}$$

Proof: (15) follows by combing (7), (10) and (12). When n goes to infinity, since the number of items that each bin could pack is at most $\lfloor 1/\eta \rfloor$, then OPT also goes to infinity. Since $m\lfloor 1/\eta \rfloor$ is also a constant, then (16) holds. ∎

Note that in Theorem 1, we only assume that $\sigma_i \ge 0$ and $\mu_i + \beta\sigma_i \le 1$ for all i. In fact, the upper bound of the worst-case performance ratio is achieved in the group that $\mu \le \frac{1}{2}$ and $\sigma \in (\frac{1-2\mu}{\sqrt{2}\beta}, 1-\mu]$. If no item belongs to this group, the upper bound can be reduced. Therefore, provided with further constraints on μ and σ such that no item belongs to groups with a large ratio $\frac{f(\mu,\sigma)}{g(\mu,\sigma)}$, we can indeed obtain a stronger result than that in Theorem 1.

Theorem 2. *Same assumption as in Theorem 1. If it further holds that for some $h \in \mathcal{Z}^+$, $h\mu_i + \beta\sqrt{h}\sigma_i \le 1$ for all i, then*

$$B < \lambda(h)OPT + m\lfloor 1/\eta \rfloor, \text{and } \lim_{n\to\infty} B/OPT \le \lambda(h),$$

where $\lambda(h) = (2h^2(\sqrt{h/(h+1)}-1)+h)^{-1}$.

Proof: It follows by combing (7), (9) and (11). ∎

The worst-case performance ratio $\lambda(h)$ strictly decreases as h increases, and $\lim_{h\to\infty}\lambda(h) = 4/3$. In practice, since $\mu_i > \eta$ for all item i, then h is no greater than $1/\eta$.

V. Generic scenarios

Section IV discussed the special case that the means of the items form a finite set whose cardinality does not change as the number of items increases. Here we consider the general case that every two items can have different means and different variances. In other words, we only assume that $\mu_i \in (\eta, 1)$, $\sigma_i \ge 0$, and $\mu_i + \beta\sigma_i \le 1$ for all i. For any $\epsilon > 0$, we provide an online packing algorithm whose worst case performance ratio is $(1+\epsilon)(\sqrt{2}+1)$.

We first introduce the algorithm. Same as Algorithm 1, the key idea is to divide items into different groups and pack items in the same group by Next Fit. In Algorithm 1, since there are only m possibilities for the mean, then we classify items with the same mean according to the variance such that for different groups, the number of items that one bin can pack is different. Here we come across the difficulty that each item could have a unique mean. Therefore we need to divide the items in a different and more cautious way.

We will define a strictly decreasing sequence $\{c_k\}$ to divide the means into intervals $(c_1, 1]$ and $(c_{k+1}, c_k]$ $(k \ge 2)$, and for items with means belong to $(c_{k+1}, c_k]$, we will use $\{d_h^k, h = 1, ..., m_k = \lfloor 1/c_k \rfloor\}$ to divide their standard deviations into intervals $[d_{m_k}^k, d_{m_k-1}^k]$ and $(d_h^k, d_{h-1}^k](h = 1, ..., m_k - 1)$. Then items with means in the same interval and standard deviations in the same interval belong to the same group. To ensure that the number of groups is finite, $(\eta, 1)$ should be covered by finite number of intervals we defined. Moreover, for interval $(c_{k+1}, c_k]$, the union of $[d_{m_k}^k, d_{m_k-1}^k]$ and $(d_h^k, d_{h-1}^k](h = 1, ..., m_k - 1)$ should cover $[0, (1-c_{k+1})/\beta]$, since $\mu_i + \beta\sigma_i \le 1$ for every item i.

We define a sequence c_k together with d_h^k $(h = 0, 1, ..., \lfloor 1/c_k \rfloor)$ as follows. Given $\epsilon > 0$, let $r := (1+\epsilon)(\sqrt{2}+1)$, then $c_1 = 1/r$, $m_1 = \lfloor 1/c_1 \rfloor$, and

$$d_h^1 = \frac{1-(h+1)c_1}{\beta\sqrt{h+1}}, \quad h = 0, 1, ..., m_1 - 1. \tag{17}$$

For $k \in \mathcal{Z}^+$ and $k \ge 2$,

$$p_h^k = \frac{1}{2}\Big(\frac{1}{hr} + 1 - \sqrt{(\frac{1}{hr}-1)^2 + 4(\beta d_h^{k-1})^2}\Big), \tag{18}$$

where $h = 1, ..., m_{k-1} - 1$, and $p_{m_{k-1}}^k = 1/rm_{k-1}$,

$$c_k = \max_{h\in\{1,...,m_{k-1}\}} p_h^k, \quad m_k = \lfloor 1/c_k \rfloor, \tag{19}$$

$$d_h^k = \frac{1-(h+1)c_k}{\beta\sqrt{h+1}}, \quad h = 0, 1, ..., m_k - 1. \tag{20}$$

We also define $m_0 = 1$, and $d_{m_k}^k = 0$ for all $k \ge 0$.

To show that this partition is valid, note that from (20), corresponding to an interval $(c_{k+1}, c_k]$, the union of $[d_{m_k}^k, d_{m_k-1}^k]$ and $(d_h^k, d_{h-1}^k](h = 1, ..., m_k - 1)$ is exactly $[0, (1-c_k)/\beta]$. Therefore, we only need to show that $(\eta, 1)$ could be covered by a finite number of intervals induced by $\{c_k\}$. Formally, we state the following lemma,

Lemma 4. *For a given $\epsilon > 0$, the sequence $\{c_k\}$ defined in (17)∼(20) satisfies $c_k \le c_{k-1}/(1+\epsilon)$, $\forall k \ge 2$. Thus, $\{c_k\}$ is strictly decreasing, and there exists $M \le -\log\big((\sqrt{2}+1)\eta\big)/(\log(1+\epsilon))$ such that $c_{\tilde{M}} \le \eta$, where $\tilde{M} = \max(1, M)$.*

Proof: If $k \ge 2$, from the definition we know $p_h^k < 1$ for all $h = 1, ..., m_{k-1}$. Then from (19) we have $c_k < 1$. Note that given $d_h^{k-1}(h = 1, ..., m_{k-1})$, the function $\pi(x) = x + (\beta d_h^{k-1})^2/(1-x)$ strictly increases on $[-\infty, 1)$. One can also check that $\pi(p_h^k) = 1/(hr)$ with p_h^k defined in (18). Thus, we have $x + (\beta d_h^{k-1})^2/(1-x) \ge 1/(hr)$ for all $x \in [p_h^k, 1)$, and $x + (\beta d_h^{k-1})^2/(1-x) < 1/hr$ for all $x \in [0, p_h^k)$.

From the definition of c_k in (19), we have for all $x \in [c_k, 1)$,

$$x + (\beta d_h^{k-1})^2/(1-x) \ge 1/(hr), \quad \forall h = 1, ..., m_{k-1}, \tag{21}$$

and for every $y \in [0, c_k)$, there exists some h such that $y + (\beta d_h^{k-1})^2/(1-y) < 1/(hr)$. Let $\gamma_k = c_{k-1}/(1+\epsilon)$, then in order to prove $c_k \le \gamma_k$ holds, we only need to prove that

$$\gamma_k + (\beta d_h^{k-1})^2/(1-\gamma_k) \ge 1/(hr), \quad \forall h = 1, ..., m_{k-1}. \tag{22}$$

If $c_{k-1} > 1/2$, then $m_{k-1} = 1$, and $d_1^{k-1} = 0$. Therefore,

$$\gamma_k + (\beta d_1^{k-1})^2/(1-\gamma_k) = c_{k-1}/(1+\epsilon) > 1/r. \tag{23}$$

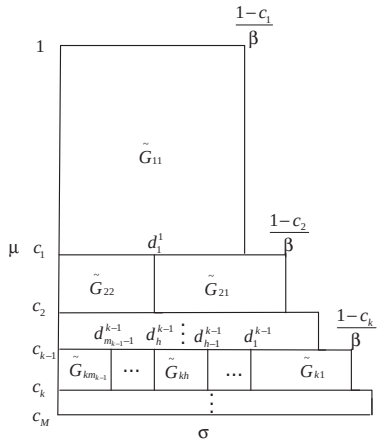

Fig. 1. Grouping items according to μ and σ

If $c_{k-1} \le 1/2$, then $1-\gamma_k = 1 - c_{k-1}/(1+\epsilon) = (1+\epsilon)(1-c_{k-1}) + \epsilon(\frac{2+\epsilon}{1+\epsilon}c_{k-1} - 1) \le (1+\epsilon)(1-c_{k-1})$. Therefore, for all $h = 1, ..., m_{k-1} - 1$,

$$\gamma_k + \frac{(\beta d_h^{k-1})^2}{1-\gamma_k} \ge \frac{1}{1+\epsilon}\big(c_{k-1} + \frac{(\beta d_h^{k-1})^2}{1-c_{k-1}}\big). \quad (24)$$

From (13) and the arguments following it, we know that

$$h\big(c_{k-1} + \frac{(\beta d_h^{k-1})^2}{1-c_{k-1}}\big) \ge 2h^2(\sqrt{\frac{h}{h+1}} - 1) + h \ge \sqrt{2} - 1, \quad (25)$$

for all $c_{k-1} \in (0, 1/2]$ and $h = 1, ..., m_{k-1} - 1$. Combing (24) and (25), we have for $h = 1, ..., m_{k-1} - 1$,

$$\gamma_k + (\beta d_h^{k-1})^2/(1-\gamma_k) \ge (\sqrt{2}-1)/(h(1+\epsilon)) = \frac{1}{hr}. \quad (26)$$

Note that since $c_{k-1} \le 1/2$, then $c_{k-1}m_{k-1} \ge c_{k-1}(1/c_{k-1} - 1) \ge 1/2 > \sqrt{2} - 1$, thus,

$$\gamma_k \ge \frac{c_{k-1}}{1+\epsilon} > \frac{\sqrt{2}+1}{(1+\epsilon)m_{k-1}} = \frac{1}{m_{k-1}r}. \quad (27)$$

Combing (23), (26) and (27), we have that (22) holds, therefore $c_k \le c_{k-1}/(1+\epsilon)$.Then, $c_k \le (1/(1+\epsilon))^{k-1}c_1 = (\sqrt{2}-1)(1+\epsilon)^{-k}$. If $\eta \ge \sqrt{2}-1$, then $c_1 \le \eta$. If $\eta < \sqrt{2}-1$, then $C_M \le \eta$. Therefore, $c_{\tilde{M}} \le \eta$. ■

If we further let $c_0 = 1$, then $(\eta, 1) \subset \cup_{k=1}^{\tilde{M}}(c_k, c_{k-1}]$ with $\tilde{M}$ defined in Lemma 4. Item i belongs to group $\tilde{G}_{km_{k-1}}$ if $\mu_i \in (c_k, c_{k-1}](k = 1, ..., \tilde{M})$, and $\sigma_i \in [0, d_{m_{k-1}-1}^{k-1}]$. It belongs to group $\tilde{G}_{kh}$ if $\mu_i \in (c_k, c_{k-1}]$ and $\sigma_i \in (d_h^{k-1}, d_{h-1}^{k-1}]$ $(h = 1, ..., m_{k-1} - 1)$. Since $k \le \tilde{M}$, and $m_k \le \lfloor 1/\eta \rfloor$ for all k, then there are at most $\tilde{M}\lfloor 1/\eta \rfloor$ groups. Figure 1 illustrates how to divide items according to the mean and the standard deviation. We briefly summarize the packing strategy in Algorithm 2. Since the classification takes $O(\log(\tilde{M}\lfloor 1/\eta \rfloor))$ time for each item, Algorithm 2 runs in $O(n\log(\tilde{M}\lfloor 1/\eta \rfloor))$ time. Since there are at most $\tilde{M}\lfloor 1/\eta \rfloor$ active bins at any time, the algorithm takes $\tilde{M}\lfloor 1/\eta \rfloor$ storage space to store active bins.

Algorithm 2 General group packing algorithm

1 Keep one active bin for each group $\tilde{G}_{kh}$.
2 For each item, decide which group it belongs to. If it fits into the current bin for that group, then pack it. Otherwise, open a new bin and make it the current bin for that group.

Let $\tilde{t}_{kh}$ denote the number of bins used to pack items in group $\tilde{G}_{kh}$, then the total number of bins used is $\tilde{B} := \sum_{k=1}^{\tilde{M}} \sum_{h=1}^{m_{k-1}} \tilde{t}_{kh}$. We have one main result as follows,

Theorem 3. *A list $L = (x_1, ..., x_n)$ with $x_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ $(\mu_i \in (\eta, 1))$ needs to be packed into the minimum of bins, such that for each bin, the probability that its size is exceeded does not exceed a given probability α. For any $\epsilon > 0$, Algorithm 2 with $\{c_k\}$ and $\{d_h^k\}$ defined in (17)∼(20) produces a feasible packing strategy with the number of bins $\tilde{B}$ satisfying*

$$\tilde{B} \le (1+\epsilon)(\sqrt{2}+1)OPT + \tilde{M}\lfloor 1/\eta \rfloor, \quad (28)$$

where OPT is the minimum number of bins needed. And the worst-case performance ratio satisfies

$$\lim_{n\to\infty} \tilde{B}/OPT \le (1+\epsilon)(\sqrt{2}+1). \quad (29)$$

Proof: Since for any item i in group $\tilde{G}_{kh}$, $\mu_i \le c_{k-1}$ and $\sigma_i \le d_{h-1}^{k-1}$, then the total size of h items in group $\tilde{G}_{kh}$ follows $\mathcal{N}(\mu, \sigma^2)$ with $\mu \le hc_{k-1}$ and $\sigma \le \sqrt{h}d_{h-1}^{k-1}$. Since $\mu + \beta\sigma \le hc_{k-1} + \beta\sqrt{h}d_{h-1}^k = 1$, then there are at least h items in one bin except the last bin for group $\tilde{G}_{kh}$. We define the *effective occupation* $\tilde{f}(\mu, \sigma)$ such that $\tilde{f}(\mu_i, \sigma_i) = \frac{1}{h}$ if item i belongs to $\tilde{G}_{kh}$ $(k = 1, ..., \tilde{M},\ h = 1, ..., m_{k-1})$. For each bin j of group $\tilde{G}_{kh}$ except the last bin, let U^j denote the set of indices of the items in this bin, then $\sum_{i\in U^j} f(\mu_i, \sigma_i) \ge 1$. Then from Lemma 2, we have

$$\tilde{B} \le \tilde{r}^* OPT + \tilde{M}\lfloor 1/\eta \rfloor, \quad (30)$$

where $\tilde{M}\lfloor 1/\eta \rfloor$ accounts for the total number of possibility unfilled bins in all the groups, and $\tilde{r}^* = \max_{U\in\mathcal{S}} \sum_{i\in U} \tilde{f}(\mu_i, \sigma_i)$ with $\mathcal{S}$ defined in (5). From (9), we have $\tilde{r}^* \le \max_i \frac{\tilde{f}(\mu_i,\sigma_i)}{g(\mu_i,\sigma_i)}$, where $g(\mu, \sigma)$ is defined in (2).

For any item i in group $\tilde{G}_{kh}$, $(k = 1, ..., \tilde{M},\ h = 1, ..., m_{k-1})$, we have $\mu_i > c_k$ and $\sigma_i \ge d_h^{k-1}$, thus,

$$g(\mu_i, \sigma_i) = \mu_i + (\beta\sigma_i)^2/(1-\mu_i) \ge c_k + (\beta d_h^{k-1})^2/(1-c_k)$$
$$= c_k + \big(1 - (h+1)c_{k-1}\big)^2/\big((h+1)(1-c_k)\big) \ge 1/(hr),$$

where the last inequality holds from (21). Then

$$\tilde{f}(\mu_i, \sigma_i)/g(\mu_i, \sigma_i) \le 1/r = (1+\epsilon)(\sqrt{2}+1). \quad (31)$$

Since for every item i, (31) holds, then

$$r^* \le \max_i \big(\tilde{f}(\mu_i, \sigma_i)/g(\mu_i, \sigma_i)\big) \le (1+\epsilon)(\sqrt{2}+1). \quad (32)$$

Therefore, (28) follows from (30) and (32). Since given ϵ, $\tilde{M}\lfloor 1/\eta \rfloor$ is a constant, and OPT goes to infinity as n goes to infinity, then (29) follows.

■

We remark that if we define $\{c_k\}$ as $c_k = c_{k-1}/(1+\epsilon)$ instead of (19), and modify Algorithm 2 accordingly, Theorem 3 still holds. However, compared with the current algorithm with (19), the items are divided into more groups if we make this change, which could lead to more unfilled bins. Thus, Algorithm 2 with $\{c_k\}$ and $\{d_{kh}\}$ defined in (17)$\sim$(20) has a better practical performance.

In the analysis of Algorithm 1 and Algorithm 2, since it is hard to compute r^* and $\tilde{r}^*$ directly, we use their upper bounds $\max_i \frac{f(\mu_i,\sigma_i)}{g(\mu_i,\sigma_i)}$ (for r^*) and $\max_i \frac{\tilde{f}(\mu_i,\sigma_i)}{g(\mu_i,\sigma_i)}$ (for $\tilde{r}^*$) instead. However, since these upper bounds are in general loose, the worst-case performance ratio we obtained are also not tight.

VI. Extension to variable-sized bins

Here we extend to a more general case that there are $q > 1$ kinds of bins with different capacities $0 < \alpha_1 < \alpha_2 < ... < \alpha_q = 1$, and there are an inexhaustible supply of bins of each size. Then for a given list L of items, and a given violation probability, we want to minimize the total capacities of the bins used to pack L. Let OPT_v denote this minimum total capacity. We also consider packing L using only unit-capacity bins like what we discussed previously and let OPT denote the minimum number of unit-capacity bins needed. Then one can easily check that $OPT_v \leq OPT$.

Theorem 1 established the relation of the number bins B used by Algorithm 1 and OPT in (15) and (16). Theorem 3 established the relation of the number of bins $\tilde{B}$ used by Algorithm 2 and OPT in (28) and (29). Here we claim that (15-16) and (28-29) still hold if we replace OPT with OPT_v. In other words, even if the optimal strategy can choose bin size for each bin, the worst-case performance ratio of implementing Algorithm 1 and Algorithm 2 with unit-capacity bins do not change. This result follows from the following lemma,

Lemma 5. *Given α, a list of items $L = (x_1, x_2, ..., x_n)$ with $x_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ are packed into B bins. Let U^j be the set of indices of item in bin j. If there exists function f such that for every bin j except Q bins (Q does not depend on n), $\sum_{i\in U^j} f(\mu_i,\sigma_i) \geq 1$, and g is defined in (2), then*

$$B \leq \max_i \big(f(\mu_i,\sigma_i)/g(\mu_i,\sigma_i)\big)OPT_v + Q,$$

Proof: Note that $B - Q \leq \sum_{i=1}^n f(\mu_i,\sigma_i)$ from assumption. Since the maximum possible bin size is 1, then $g(\mu_i,\sigma_i)$ is still a lower bound of the equivalent size of item i, i.e. (3) still holds. Thus, $\sum_{i=1}^n g(\mu_i,\sigma_i) \leq OPT_v$. Then, $B \leq \sum_{i=1}^n f(\mu_i,\sigma_i) + Q \leq \max_i \frac{f(\mu_i,\sigma_i)}{g(\mu_i,\sigma_i)}(\sum_{i=1}^n g(\mu_i,\sigma_i)) + Q \leq \big(f(\mu_i,\sigma_i)/g(\mu_i,\sigma_i)\big)OPT_v + Q$. ■

Then using the same analysis in Section IV and combining Lemma 3 and Lemma 5, we conclude that

$$B < (\sqrt{2}+1)OPT_v + m\lfloor 1/\eta \rfloor,$$

where B is the number of bins used by Algorithm 1. One can compare this result with (15). Similarly, for the number of bins $\tilde{B}$ used by Algorithm 2, following the analysis in Section V and Lemma 5, we have

$$\tilde{B} \leq (1+\epsilon)(\sqrt{2}+1)OPT_v + \tilde{M}\lfloor 1/\eta \rfloor.$$

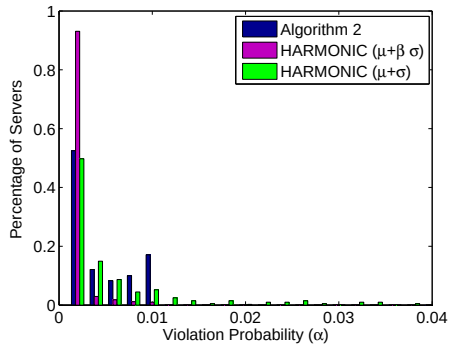

Fig. 2. Violation probability on physical machines in different strategies

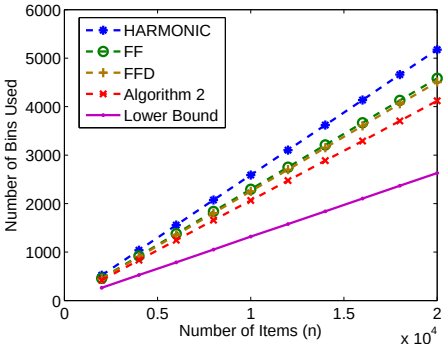

Fig. 3. Number of bins used by different algorithms

Therefore, the worst case performance bounds we obtained for Algorithm 1 and Algorithm 2 still hold for the general problem of packing random variables with variable-sized bins.

VII. Numerical Results

We use traffic dataset from global operational data centers. Details of this dataset are provided in [27]. We compute the mean and variance of the traffic rates for about 9K VMs in a six-hour period. In our simulations, we assume these VMs are consolidated onto servers equipped with 1 Gbps Ethernet card. The capacity violation probability is $\alpha = 0.01$, thus we have $\beta = 2.3263$. The number of servers used by the proposed group packing algorithm, i.e., Algorithm 2, is 421. For comparison purposes, we also implement the HARMONIC algorithm [24] (with M=12), a popular online algorithm for the classical bin packing problem. To guarantee the actual violation probability does not exceed α, we use $\mu_i + \beta\sigma_i$ as the bandwidth requirement for VM i when HARMONIC is tested. In this test, the number of servers used is 609. Comparing the proposed algorithm and HARMONIC, the proposed one reduces the required servers by 30%. If we are less conservative and use $\mu_i + 1.2\sigma_i$ as the bandwidth requirement for VM i, then the number of servers used is 402, which is comparable to the number used by Algorithm 2. However, the violation probability in this case exceeds α for some servers. We then draw 10000 samples from the obtained distributions for all

the VMs and calculate the empirical violation probability for each server in differen schemes. As shown in Figure 2, both our algorithm and HARMONIC with $\mu + \beta\sigma$ as VM size can guarantee that for each server, the violation probability does not exceed 0.01, while in the HARMONIC case with $\mu+1.2\sigma$ as VM size, the violation probability exceeds 0.01 in about 17 percents of the servers.

Next, we generate random samples and pack them with four different algorithms: Algorithm 2, HARMONIC, FFD and FF. We increase the number of items to pack from 2000 to 20000. Given $\alpha = 0.0228$ and $\beta = 2$, we let the distribution of the size of item i follow $\mathcal{N}(\mu_i, \sigma_i^2)$ with μ_i and σ_i satisfying $\mu_i + \beta\sigma_i \leq 1$. Algorithm 2 packs the distributions, while HARMONIC, FFD and FF take $\mu_i + \beta\sigma_i$ as the size of item i. We also find a lower bound of the minimum number of bins needed (OPT) as follows. We use FFD to pack items with $g(\mu_i, \sigma_i)$ as the size of item i and let B be the number of bins used. Since the minimum number of bins needed $g(\mu_i, \sigma_i)$ $(i = 1, ..., n)$, denoted by OPT_g, is less than OPT, and the FFD algorithm guarantees that $B \leq \frac{11}{9}OPT_g + 1$([36]), then $\frac{9}{11}(B-1) \leq OPT_g \leq OPT$, and $\frac{9}{11}(B-1)$ is a lower bound of OPT. Figure 3 compares the number of bins used by the four algorithms. The result for each n is averaged over 1000 runs. All the algorithms can guarantee that the violation probability does not exceed 0.0228. The result shows that the number of bins used varies significantly: Algorithm 2 uses the least among the four, and is within 1.6 times optimum.

VIII. Conclusion

Most traditional VM consolidation schemes only consider CPU, memory and disk constraints, and solve a bin packing type of problem by making a deterministic estimate of resource demands. This paper studies VM consolidation problem when network devices in data centers impose bandwidth constraints. Because of the dynamic nature in data center traffic, we formulate the VM consolidation as a novel Random Variable Packing problem which models the bandwidth demands of VMs as probabilistic distributions. Such a probabilistic approach better captures the uncertainty in network bandwidth demand than the traditional deterministic model. We propose an approximation algorithm and prove its worst-case performance ratio of $(1+\epsilon)(\sqrt{2}+1)$ for any $\epsilon > 0$. The ratio is further improved to $\sqrt{2}+1$ in special cases. We demonstrate with numerical experiments that the proposed algorithm saves many servers without violating the server capacity constraints. Due to the generality of the defined RVP problem and the algorithm, the results in this work can apply to the VM consolidation problem for other resource types and beyond. SBP with general item size distributions is interesting to explore.

References

[1] Y. Ajiro and A. Tanaka, "Improving packing algorithms for server consolidation," in *Proceedings of the International Conference for the Computer Measurement Group (CMG)*, 2007.

[2] T. A. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding data center traffic characteristics," in *ACM SIGCOMM WREN workshop*, 2009.

[3] R. E. Burkard and G. Zhang, "Bounded space on-line variable-sized bin packing," *Acta Cybern.*, vol. 13, no. 1, pp. 63–76, 1997.

[4] E. G. Coffman, Jr., M. R. Garey, and D. S. Johnson, "Approximation algorithms for bin packing: a survey," pp. 46–93, 1997.

[5] J. Csirik, "An on-line algorithm for variable-sized bin packing," *Acta Inf.*, vol. 26, no. 9, pp. 697–709, 1989.

[6] D.J.Brown, "A lower bound for on-line one-dimensional bin packing algorithms," *Tech. Rep. No. R-864*, 1979.

[7] G. Dósa, "The tight bound of first fit decreasing bin-packing algorithm is FFD(I)=(11/9)OPT(I)+6/9," in *Combinatorics, Algorithms, Probabilistic and Experimental Methodologies*, 2007.

[8] D. K. Friesen and M. A. Langston, "A storage-size selection problem," *Inf. Process. Lett.*, vol. 18, no. 5, pp. 295–296, 1984.

[9] ——, "Variable sized bin packing," *SIAM Journal on Computing*, vol. 15, no. 1, pp. 222–230, 1986.

[10] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co Ltd, January 1979.

[11] A. Goel and P. Indyk, "Stochastic load balancing and related problems," *40th IEEE Annual Symposium on Foundations of Computer Science*, vol. 0, p. 579, 1999.

[12] IBM, "Server Planning Tool, http://www-304.ibm.com/jct01004c/systems/support/tools/systemplanningtool/."

[13] IBM WebSphere CloudBurst, "http://www-01.ibm.com/software/webservers/cloudburst/."

[14] D. S. Johnson, A. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham, "Worst-case performance bounds for simple one-dimensional packing algorithms," *SIAM Journal on Computing*, vol. 3, no. 4, pp. 299–325, 1974.

[15] D. S. Johnson, *Ph.D. dissertation*. MIT, Cambridge, Mass., June 1973.

[16] ——, "Fast algorithms for bin packing," *Journal of Computer and System Sciences*, vol. 8, no. 3, pp. 272 – 314, 1974.

[17] D. S. Johnson and M. R. Garey, "A 71/60 theorem for bin packing," *Journal of Complexity*, vol. 1, no. 1, pp. 65 – 106, 1985.

[18] J.Shahabuddin, A.Chrungoo, V.Gupta, S.Juneja, S.Kapoor, and A.Kumar, "Stream-packing: Resource allocation in web server farms with a QOS guarantee," in *HiPC*, 2001.

[19] S. Kandula, S. Sengupta, A. Greenberg, and P. Patel, "The nature of datacenter traffic: Measurements and analysis," in *ACM IMC*, 2009.

[20] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum Press, 1972, pp. 85–103.

[21] N. G. Kinnerseley and M. A. Langston, "Online variable-sized bin packing," *Discrete Applied Mathematics*, vol. 22, no. 2, pp. 143 – 148, 1989.

[22] J. Kleinberg, Y. Rabani, and E. Tardos, "Allocating bandwidth for bursty connections," in *Proc. STOC*, 1997, pp. 664–673.

[23] Lanamark Suite, "http://www.lanamark.com/."

[24] C. C. Lee and D. T. Lee, "A simple on-line bin-packing algorithm," *J. ACM*, vol. 32, no. 3, pp. 562–572, 1985.

[25] F. M. Liang, "A lower bound for on-line bin packing," *Information Processing Letters*, vol. 10, pp. 76 – 79, 1980.

[26] S. Mehta and A. Neogi, "Recon: a tool to recommend dynamic server consolidation in multi-cluster data centers," in *NOMS*, 2008.

[27] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *IEEE INFOCOM*, 2010.

[28] N.Bobroff, A.Kochut, and K.Beaty, "Dynamic placement of virtual machines for managing SLA violations," in *Integrated Network Management*, 2007.

[29] Novell PlateSpin Recon, "http://www.novell.com/products/recon/."

[30] S. S. Seiden, "An optimal online algorithm for bounded space variable-sized bin packing," *SIAM Journal on Discrete Mathematics*, vol. 14, no. 4, pp. 458–470, 2001.

[31] VMware Inc., "VMware Capacity Planner, http://www.vmware.com/products/capacity-planner/."

[32] ——, "VMWare vCenter CapacityIQ, http://www.vmware.com/products/vcenter-capacityiq/."

[33] ——, "Resource Management with VMware DRS," VMware Inc., Whitepaper, 2006.

[34] W.Leinberger, G.Karypis, and V.Kumar, "Multi-capacity bin packing algorithms with applications to job scheduling under multiple constraints," in *ICPP*, 1999.

[35] A. C.-C. Yao, "New algorithms for bin packing," *J. ACM*, vol. 27, no. 2, pp. 207–227, 1980.
[36] M. Yue and L. Zhang, "A simple proof of the inequality MFFD(L)$\leq$ 71/60 OPT(L) + 1, L for the MFFD bin-packing algorithm," *Acta Mathematicae Applicatae Sinica*, vol. 11, no. 3, pp. 318–330, 1995.
[37] G. Zhang, "Worst-case analysis of the ffh algorithm for online variable-sized bin packing," *Computing*, vol. 56, no. 2, pp. 165–172, 1996.