On-line Bin Packing of Fragile Objects with Application in Cellular Networks

Wun-Tat Chan[1⋆], Francis Y. L. Chin[1⋆⋆], Deshi Ye[1], Guochuan Zhang[2⋆⋆⋆], and Yong Zhang[1]

[1] Department of Computer Science, The University of Hong Kong, Hong Kong.
{wtchan, chin, yedeshi, yzhang}@cs.hku.hk
[2] Department of Mathematics, Zhejiang University, Hangzhou 310027, China.
zgc@zju.edu.cn

Abstract. We study a specific bin packing problem which arises from the channel assignment problems in cellular networks. In cellular communications, frequency channels are some limited resource which may need to share by various users. However, in order to avoid signal interference among users, a user needs to specify to share the channel with at most how many other users, depending on the user's application. Under this setting, the problem of minimizing the total channels used to support all users can be modeled as a specific bin packing problem as follows: Given a set of items, each with two attributes, weight and fragility. We need to pack the items into bins such that, for each bin, the sum of weight in the bin must be at most the smallest fragility of all the items packed into the bin. The goal is to minimize the total number of bins (i.e., the channels in the cellular network) used. We consider the on-line version of this problem, where items arrive one by one. The next item arrives only after the current item has been packed, and the decision cannot be changed. We show that the asymptotic competitive ratio is at least 2. We also consider the case where the ratio of maximum fragility and minimum fragility is bounded by a constant. In this case, we present a class of online algorithms with asymptotic competitive ratio at most of $1.7r$, for any $r > 1$.

Keywords: Bin packing; Channel assignment; On-line algorithm.

1 Introduction

In a typical cellular network, various users communicate with the base station using various frequency channels. While in a CDMA (code division multiple access) system, since each channel has a capacity much larger than the bandwidth requirement of a single user, it is possible to allow many users to share the same channel. Therefore to maximize the bandwidth utility in such system, one

⋆ The research was supported in part by Hong Kong RGC Grant HKU5172/03E
⋆⋆ The research was supported in part by Hong Kong RGC Grant HKU7142/03E.
⋆⋆⋆ The research was supported in part by NSFC (10231060).

straightforward approach is to assign as many users to a single channel. However such assignment may result in interference between the users on the same channel, which affects the quality of communication. There exists a trade-off between the bandwidth utilization and communication quality. Roughly speaking, the fewer the users to share a common channel, the better the communication quality of each user. The requirements on communication quality may differ from users depending on the applications the users are running. To quantify the quality of communication of a user on a channel, we use a measure called the signal to noise ratio (SNR) [7, 9, 10]. Suppose there are M users communicating with a central base station at the same time with the same channel. If user i transmits with power p_i, then the signal received by the base station is $s_i = p_i g_i$, where g_i is the gain on the channel for user i. The SNR of user i is given by

$$\frac{s_i}{N_0 + \sum_{1 \leq j \leq M, j \neq i} s_j}, \tag{1}$$

where N_0 is the background noise power, e.g., the receiver thermal noise, which is assumed a constant. Generally speaking, the larger the value of SNR the better the communication quality. Suppose that each user specifies a minimum SNR that the user can tolerate. The objective of the channel assignment problem (CAP) is to assign the users to channels so that the SNR of each user of each channel is better than the minimum SNR the user specifies.

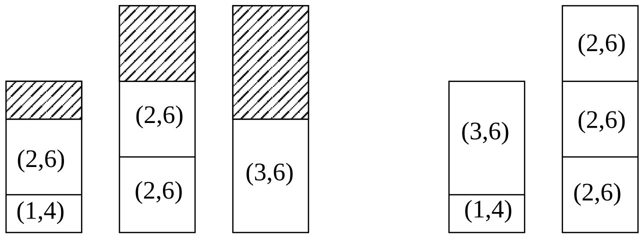
To tackle the CAP, Bansal, Liu and Zankar [1] defined the problem of bin packing with fragile objects (BPFO). The traditional bin packing problem [2, 3] is defined as follows. A sequence of (ordinary) items is given where each item has weight in $(0, 1]$. The goal is to pack all items into a minimum number of bins so that the total item weight in each bin is no more than 1. For BPFO, besides weight w_i, each (fragile) item is associated a *fragility* f_i. In each bin, the total weight of the items in the bin must be no more than the minimum fragility among the items in the bin. Precisely, suppose k items $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ are packed into the same bin. The packing is *feasible* if

$$w_{i_1} + w_{i_2} + \ldots + w_{i_k} \leq \min\{f_{i_1}, f_{i_2}, \ldots, f_{i_k}\}.$$

The goal of BPFO is to find a feasible packing of all items with the minimum number of bins.

It can be shown that BPFO can model CAP. The frequency channels can be regarded as bins and the users as the items. For each user i, we let the amount of power received at the base station s_i be the weight w_i of item i. Suppose user i requires a minimum SNR of value β_i. From the SNR expression 1, we can see that user i can tolerate at most $s_i\beta_i - N_0$ power from other users on the same channel and still maintain the communication quality. We let $f_i = s_i + s_i\beta_i - N_0$. Then, CAP can be seen as a special case of BPFO. In the rest of the paper, we will focus on solving the BPFO.

Suppose an item is denoted as (w, f) where w is the weight and f is the fragility of the item. Fig. 1 gives an example of two feasible ways of packing the items $(1, 4), (2, 6), (2, 6), (2, 6)$ and $(3, 6)$. One (non-optimal) solution is to pack the first two items into one bin. The sum of weights is 3, which is less than the minimum fragility 4. Then the next two items are packed into one bin and the final item into another bin (see Fig. 1(a)). In the optimal solution, one can pack the first item and the last item into one bin and the other items into another bin (see Fig. 1(b)).



(a) Non-optimal (b) Optimal

Fig. 1. Two feasible ways of packing the items $(1, 4), (2, 6), (2, 6), (2, 6), (3, 6)$.

Bansal, Liu and Zankar [1] studied the offline version BPFO, i.e., the set of items is given in the beginning. In this paper, we consider the on-line version of BPFO where the items arrive one by one, and the weight and fragility of an item are not known until the item arrives. This on-line version of BPFO also models the on-line version of CAP where the users arrive one by one and the power and minimum SNR of user are not known until the user arrives. For the on-line BPFO, when an item arrives, it must be assigned to one of the bins before the next item becomes known. Our goal is still to find a feasible packing of all items which minimize the number of bins.

The classical bin packing problems have been studied extensively. It is well known that the problem is NP-hard [6]. For other results in bin packing, the readers are referred to the survey papers [2–4]. For the offline BPFO, Bansal, Liu and Zankar [1] gave a 2-approximation algorithm, i.e., the algorithm can always pack the items with at most two times the number of bins used in the optimal solution. They also showed that no algorithm can achieve an approximation ratio of less than $3/2$ unless $P = NP$. Then they gave an algorithm that uses at most the same number of bins as the optimum solution but the sum of item weight in each bin can be at most twice the minimum fragility of that bin.

Competitive analysis: To evaluate the on-line algorithms presented in this paper, we adopt the conventional measure of *competitive ratio*. For any input sequence I, let $A(I)$ and $OPT(I)$ denote the number of bins used by the on-line algorithm A and an optimal offline algorithm, respectively. The on-line algorithm A has a competitive ratio c if there exists a constant h such that

$$A(I) \leq c \cdot OPT(I) + h$$

for any input sequence I. In such case, we also say that A is a *c-competitive* on-line algorithm. The *asymptotic competitive ratio* R_A^∞ of A is defined by

$$R_A^\infty = \limsup_{m\to\infty} R_A^m \quad \text{where} \quad R_A^m = \max\{A(I)/OPT(I) \mid OPT(I) = m\}.$$

For simplicity we may use just A and OPT instead of $A(L)$ and $OPT(L)$ if the context is clear.

Our results: Let k be the *fragility ratio*, which is the ratio between maximum fragility and minimum fragility. We show that no on-line algorithm can have asymptotic competitive ratio less than 2 when k can be arbitrary large. We also show that the asymptotic competitive ratio of an any-fit algorithm, which include the classical first-fit and best-fit, is at least k. For the case that k is bounded by a constant, we develop a class of on-line algorithms that achieves the asymptotic competitive ratio $1.7r$ for any $r > 1$.

The remainder of this paper is organized as follows. In Section 2 we show that when the fragility ratio can be arbitrary large there is no on-line algorithm that is better than 2-competitive. In Section 3 we consider that the fragility ratio is bounded by a constant k. Section 3.1 shows that the asymptotic competitive ratio of an any-fit algorithm cannot be better than k. Section 3.2 gives a new class of algorithms and shows that its asymptotic competitive ratio $1.7r$ for any $r > 1$.

2 Unbounded Fragility Ratio

In this section we assume that the fragility ratio k can be arbitrary large. We show that the asymptotic competitive ratio is at least 2 for any on-line algorithm.

Theorem 1. *For the problem of packing fragile objects, no on-line algorithm has a asymptotic competitive ratio less than 2.*

Proof. First, we give an adversary of at most 6 items. We show that for any on-line algorithm the competitive ratio of packing these items is at least 2. Then we argue that the adversary can be extended to a long sequence that the optimal solution needs an arbitrary large number of bins.

The adversary begins with two items a_1 and a_2, where $a_1 = (\varepsilon, 1/4 + \varepsilon)$ and $a_2 = (1/4, 1)$, for $0 < \varepsilon < 1/8$. In the following, we exhaust all the ways how an on-line algorithm A can pack these two items and the subsequent items and show that the competitive ratio is at least 2.

Case 1. If the items a_1 and a_2 are packed into two different bins, the adversary stops. Clearly, $A = 2$ and $OPT = 1$.

Case 2. Consider the items a_1 and a_2 are packed into the same bin, namely B_1. Since the smaller fragility among these two items is $1/4+\varepsilon$, which is the same as the total weight. B_1 is considered full in this case. The adversary gives another two items a_3 and a_4, where $a_3 = (\varepsilon, 1/4 + \varepsilon)$ and $a_4 = (1/4, 1)$, respectively.

Case 2.1. Consider the items a_3 and a_4 are packed into the same bin. Then adversary has the final two items $a_5 = (1/4 - \varepsilon, 1/4 + \varepsilon)$ and $a_6 = (1/4, 1)$ arrive, which obviously cannot be packed into one bin. So the algorithm needs to use 4 bins in total, i.e., $A = 4$. On the other hand, the optimal offline algorithm can pack a_1, a_3 and a_5 into one bin, and the remaining items into another bin, thus $OPT = 2$.

Case 2.2. Consider the items a_3 and a_4 are packed into two different bins, namely B_2 and B_3, respectively. In this case, the adversary gives an item $a_5 = (\varepsilon, 1/2+\varepsilon)$.

Case 2.2.1. If the item a_5 is packed together with item a_3, i.e., into bin B_2. The adversary gives the last item $a_6 = (1/4-\varepsilon, 1/4+\varepsilon)$. Since a_6 cannot fit into the existing three bins, B_1, B_2 and B_3, it has to be packed into a new bin. Therefore, $A = 4$. On the other hand, the optimal offline algorithm can pack a_1, a_3 and a_6 into one bin, and a_2, a_4 and a_5 into another bin. Then $OPT = 2$.

Case 2.2.2. If the item a_5 is packed together with item a_4, i.e., into bin B_3. The adversary gives the last item $a_6 = (1/4 + \varepsilon, 1)$. Since a_6 cannot fit into the existing three bins, B_1, B_2 and B_3, it has to be packed into a new bin. Therefore, $A = 4$. On the other hand, the optimal offline algorithm can pack items a_1, a_3, a_5 into one bin and items a_2, a_4, a_6 into another bin. Then $OPT = 2$.

Case 2.2.3. If the item a_5 is packed into a new bin, then the adversary stops. We have $A = 4$ but the optimal offline algorithm can use two bins, i.e., $OPT = 2$.

From all above cases, we show that $R_A \geq 2$. To bound the asymptotic competitive ratio, we can give an item sequence of length arbitrary large, so that the number of bins required by the optimal solution is arbitrary large. Let I_1 denote the sequence of items given by the adversary above (with at most 6 items). We define an arbitrary long sequence, composed by subsequence, $I_1, I_2, \ldots, I_m$, where I_i is obtained from I_{i-1} as follows. Since the fragility can be arbitrary large, we can make a copy of the sequence I_{i-1} and consider it as I_i but with the weight and the fragility of the items to be scaled up by the a ratio so that the minimum weight of items in I_i is larger than the maximum fragility of the items in I_{i-1}. In that case, any item in I_i cannot be packed into the bins containing items of I_{i-1} and the optimal solution needs to use as many bins as the length of the sequence, which is arbitrary large. Together with the fact that for every subsequence I_i the on-line algorithm uses at least two times the number of bins of the optimal solution for I_i, we have the asymptotic competitive ratio for any on-line algorithm at least 2. □

3 Bounded Fragility Ratio

In this section we assume that the fragility ratio k is a given constant. We first analyze the any-fit algorithm and then develop some new algorithms.

3.1 Any-fit Algorithms

We consider a common type of algorithms, called the any-fit algorithms. An any-fit algorithm packs an item into a non-empty bin if the bin can accommodate the item, otherwise the item is packed into an empty bin. There are special cases of the any-fit algorithm, e.g., first-fit and best-fit which differ in the way of choosing which non-empty bin to pack the item. In the following theorem, we show that an any-fit algorithm has an asymptotic competitive ratio at least k.

Theorem 2. *The asymptotic competitive ratio of an any-fit algorithm cannot be better than k.*

Proof. The adversary sequence consists of $2kn$ items: $a, b, a, b, \ldots$, where $a = (1 - 1/kn, k)$ and $b = (1/kn, 1)$. An any-fit algorithm packs each pair items a and b into one bin. Thus, the number of bins used by the any-fit algorithm is kn. For the optimal solution, one can pack k items a together into one bin, and all the kn items a_2 in one bin. Then $OPT = n + 1$. The theorem follows as n can be arbitrary large. $\square$

3.2 Items Divided by Classes

We define a general method to pack the items by first dividing the items into *classes*. A predefined parameter $r > 1$ is used. An item with fragility f belongs to the class s for an integer s if $f \in [r^s, r^{s+1})$. As the fragility ratio is at most k, the number of classes is at most $\lceil \log_r k \rceil$. We only pack items of the same class into the same bin. The bin which stores an item of class s is denoted as a class s bin. The general method is described as follows. When a new item with fragility f arrives, we find an integer s such $r^s \leq f < r^{s+1}$. If there is a class s non-empty bin that the item can fit in, then pack the item into one of those bins. Otherwise, pack the item into an empty bin.

We give two specific implementations of the general method, namely the ANF_r and AFF_r. They differ in the way of packing items of the same class. In short, ANF_r makes use of next-fit and AFF_r makes use of first-fit. Precisely, for ANF_r, it maintains at most one *active* non-empty bin only for each class. If the item cannot fit in the active bin, the bin is closed and the item is packed into an empty bin which becomes the only active bin in the class. It is clear that next-fit takes $O(1)$ time to pack each item and $O(n)$ time to pack a sequence of n item. For AFF_r, according to the order in which the bins are used, it finds the first non-empty bin in the class where the new item can fit it and packs the item into that bin. However, in the worst case it takes $\Theta(n \log n)$ time to pack a sequence of n items [8].

In the following we give a tight analysis of the performance of ANF_r.

Lemma 1. *The asymptotic competitive ratio of algorithm ANF_r cannot be better than $2r$.*

Proof. We give an adversary as follows. There are $2rn$ pairs of items $(1/2, r)$ and $(\varepsilon, 1)$ arriving, where $\varepsilon > 0$ is a sufficiently small number. After that, a sequence of items with small weights and increasing fragilities arrives. Precisely, the whole item sequence is

$$\underbrace{(1/2, r), (\varepsilon, 1), (1/2, r), (\varepsilon, 1), \ldots, (1/2, r), (\varepsilon, 1)}_{4rn \text{ items}}, (\varepsilon, r^2), (\varepsilon, r^3), \ldots, (\varepsilon, k).$$

It can be seen that $ANF_r = 2rn + T - 1$ where $T \leq \log_r k$ is the number of classes. For an optimal solution, it can pack $2r$ items $(1/2, r)$ into one bin and all the items with weight ε into another bin. Hence, $OPT = n + 1$. The asymptotic competitive ratio of algorithm ANF_r is $(2rn + T - 1)/(n + 1)$, which is arbitrary close to $2r$ as n and hence OPT are arbitrary large. ☐

Before we analyze the upper bound on the asymptotic competitive ratio of ANF_r, we give a lower bound on the number of bins to pack a set of items in the following lemma.

Lemma 2. *Given a set of n items (w_i, f_i) for $1 \leq i \leq n$, the n items cannot be packed with less than $\sum_{i=1}^{n} w_i/f_i$ bins.*

Proof. For any packing of the n items, consider one of the bins. Suppose that t items $(w_{j_1}, f_{j_1}), \ldots, (w_{j_t}, f_{j_t})$ are packed into the bin. We have $\sum_{i=1}^{t} w_{j_i} \leq \min_{i=1}^{t}(f_{j_i})$ and thus $\sum_{i=1}^{t} (w_{j_i}/f_{j_i}) \leq \sum_{i=1}^{t} w_{j_i}/\min_{i=1}^{t}(f_{j_i}) \leq 1$. Counting all the bins in the packing, we can see that the number of bins used is at least $\sum_{i=1}^{n} w_i/f_i$.

Theorem 3. *The asymptotic competitive ratio of algorithm ANF_r is no more than $2r$.*

Proof. Consider an item sequence I. We create another item sequence I' based on I. For each item (w, f) in I, there is a corresponding item (w, f') in I', with the same weight but different fragility. Precisely, if $r^s \leq f < r^{s+1}$ for some integer s, i.e., the item of I in class s, then we have $f' = r^{s+1}$ and the corresponding item of I' is in class $(s + 1)$. It is easy to see that the number of bins required to pack the items in I' is no more than that for items in I, i.e., $OPT(I') \leq OPT(I)$, because each item has a larger fragility.

Let S and S' be the sets of integers that denote the classes required to divide the items in I and I', respectively. By the construction of I', we can see that for each $i \in S$, we have $i + 1 \in S'$. Let q_i and q'_i be the total weight of items in class i for I and I', respectively. We have $q_i = q'_{i+1}$ according to the construction of I'. By Lemma 2, we have $OPT(I') \geq \sum_{i \in S'} q'_i/r^i = \sum_{i \in S} q_i/r^{i+1}$.

Consider I and the packing of items in class i by ANF_r. Let b_i be the number of bins used by ANF_r for the items in class i. We have $ANF_r(I) = \sum_{i \in S} b_i$. We claim that $b_i \leq 2rq_i/r^{i+1} + 1$ for any $i \in S$. The claim is proved as follows. We say that two bins of the same class are "adjacent" if in the execution of ANF_r one of the bins is closed and immediately the other one is made active. For any two adjacent bins of the same class i, we can see that their total weight is at least r^i, otherwise all items in the two bins can be packed into one. This implies that on average, except the last one if b_i is odd, each bin stores items of total weight at least $r^i/2$. Therefore, except the last bin if b_i is odd, if we consider each bin has weight just $r^i/2$, the total weight of all bins is at most the total weight of items in class i. Precisely, we have $(b_i - 1)r^i/2 \leq q_i$, i.e., $b_i \leq 2rq_i/r^{i+1} + 1$.

As a result, we have

$$B_{ANF_r}(I) = \sum_{i \in S} b_i \leq \sum_{i \in S} (2rq_i/r^{i+1} + 1) \leq 2rB_{\mathrm{OPT}}(I') + |S| \leq 2rB_{\mathrm{OPT}}(I) + |S|$$

where $|S| \leq \log_r k$ is the number of classes. □

In the following we analyze the performance of AFF_r and give an upper bound on its asymptotic competitive ratio. Our analysis makes use of the function W defined in [5] and some lemma and theorem related to the classical bin packing [5]. The function $W : [0,1] \to [0, 8/5]$ is defined as follows:

$$W(\alpha) = \begin{cases} \frac{6}{5}\alpha & \text{for } 0 \leq \alpha \leq \frac{1}{6}, \\ \frac{9}{5}\alpha - \frac{1}{10} & \text{for } \frac{1}{6} < \alpha \leq \frac{1}{3}, \\ \frac{6}{5}\alpha + \frac{1}{10} & \text{for } \frac{1}{3} < \alpha \leq \frac{1}{2}, \\ \frac{6}{5}\alpha + \frac{4}{10} & \text{for } \frac{1}{2} < \alpha \leq 1. \end{cases}$$

Lemma 3 (Garey et al. [5]). *Given a set of numbers $S = \{w_1, w_2, \ldots\}$ with total at most 1, i.e., $\sum_{w_i \in S} w_i \leq 1$, we have $\sum_{w_i \in S} W(w_i) \leq 1.7$.*

Theorem 4 (Garey et al. [5]). *In the classical bin packing problem, given any sequence I of n items with weights $w_1, w_2, \ldots, w_n$, the number of bins used by first-fit, $FF(I)$, satisfies the inequality: $FF(I) \leq \sum_{1 \leq i \leq n} W(w_i) + 1$.*

From Lemma 3, we have a lower bound on the number of bins used in the classical bin packing.

Corollary 1. *In the classical bin packing problem, given any sequence I of n items with weights $w_1, w_2, \ldots, w_n$, the number of bins required to pack all items of I is at least $\sum_{1 \leq i \leq n} W(w_i)/1.7$.*

Theorem 5. *$AFF_r(I) \leq 1.7r \cdot OPT(I) + \log_r k$ for any item sequence I.*

Proof. Consider an item sequence I. We create a new item sequence I' so that for each item (w, f) in I, there is a corresponding item (w, f') in I' with $f' = r^s$ where $r^s \leq f < r^{s+1}$ for some integer s. It can be proved that $AFF_r(I) \leq AFF_r(I')$.

Consider a packing by AFF_r for items of I'. Let S and S' be the sets of integers that denote the classes required to divide the items in I and I', respectively. Note that $S = S'$ because for any item of I the corresponding item of I'

falls into the same class as the item of I. Let b'_i be the number of bins used for the items in class i and let $T^i = \{w^i_1, w^i_2, \ldots\}$ be the weights of items in class i. As $S = S'$, the definition of the set T^i is the same for both I and I'. Since all items in class i have the same fragility r^i, we can apply Theorem 4 and have an upper bound on b'_i, i.e., $b'_i \leq \sum_{w^i_j \in T^i} W(w^i_j/r^i) + 1$. Summing up for all classes, we have

$$
\begin{aligned}
AFF_r(I') = \sum_{i \in S'} b'_i &\leq \sum_{i \in S'} \sum_{w^i_j \in T^i} W(w^i_j/r^i) + |S'| \\
&\leq r \cdot \sum_{i \in S'} \sum_{w^i_j \in T^i} W(w^i_j/r^{i+1}) + |S'| \\
&\leq r \cdot \sum_{(w,f) \in I} W(w/f) + |S| \qquad (2)
\end{aligned}
$$

because $S = S'$ and for each class i item $(w, f) \in I$ we have $f < r^{i+1}$ and the function W is monotonic increasing.

We define a classic bin packing problem instance I^c by the input sequence I. For each item (w, f) in I, we create an item in I^c with weight w/f. The number of bins used by the optimal packing for I^c, $OPT^c(I^c)$, is at most the number of bins used by the optimal packing for I, i.e., $OPT^c(I^c) \leq OPT(I)$, because we can always transform the latter solution to a former one. By Corollary 1, we have $1.7 \cdot OPT^c(I^c) \geq \sum_{(w,f) \in I} W(w/f)$. Together with Inequality 2, we have

$$AFF_r(I) \leq AFF_r(I') \leq 1.7r \cdot OPT^c(I^c) + |S| \leq 1.7r \cdot OPT(I) + |S|$$

where $|S| \leq \log_r k$ is the number of classes. □

Remarks: Suppose that the items of the input sequence consist of only d distinct fragilities. We can modify our algorithm AFF to group only the items of the same fragility to the same class and then apply first-fit to pack the items. If $d < k$, then AFF in this case has a slightly better performance in terms of the additive constant, i.e., $AFF \leq 1.7r \cdot OPT + d$.

References

1. N. Bansal, Z. Liu, and A. Sankar. Bin-packing with fragile objects. In R. A. Baeza-Yates, U. Montanari, and N. Santoro, editors, *IFIP TCS*, volume 223 of *IFIP Conference Proceedings*, pages 38–46. Kluwer, 2002.
2. E. G. Coffman, Jr., G. Galambos, S. Martello, and D. Vigo. Bin pakcing approximation algorithms: Combinatorial analysis. In D.-Z. Du and P. M. Pardalos, editors, *Handbook of Combinatorial Optimization*. Kluwer Academic Publishers, 1998.
3. E. G. Coffman, Jr., M. R. Garey, and D. S. Johnson. Bin packing approximation algorithms: A survey. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 46–93. PWS, 1996.

4. J. Csirik and G. J. Woeginger. On-line packing and covering problems. In A. Fiat and G. J. Woeginger, editors, *On-line Algorithms—The State of the Art*, volume 1442 of *Lecture Notes in Computer Science*, pages 147–177. Springer, 1996.
5. M. R. Garey, R. L. Graham, D. S. Johnson, and A. C.-C. Yao. Resource constrained scheduling as generalized bin packing. *J. Combin. Theory, Ser. A*, 21:257–298, 1976.
6. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, San Francisco, 1979.
7. S. A. Grandhi, J. Zander, and R. Yates. Constrained power control. *Wireless Personal Communications*, 1(4):257–270, 1994.
8. D. S. Johnson. Fast algorithms for bin packing. *J. Comput. Syst. Sci.*, 8:272–314, 1974.
9. A. Sampath, P. S. Kumar", and J. M. Holtzman. Power control and resource management for a multimedia cdma wireless system. In *The Sixth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'95)*, volume 1, pages 21–25, 1995.
10. B. E. Shrader, R. S. Karlsson, L. L. H. Andrew, and J. Zander. Power-control-based band allocation in multiband cdma. In *IEEE Global Telecommunications Conference (GLOBECOM '01)*, volume 6, pages 3306–3310, 2001.