Packing to Angles and Sectors

Piotr Berman Jieun Jeong Shiva Prasad Kasiviswanathan Bhuvan Urgaonkar
Department of Computer Science and Engineering
Pennsylvania State University
e-mail: {berman,jijeong,kasivisw,bhuvan}@cse.psu.edu

Abstract

In our problem we are given a set of customers, their positions on the plane and their demands. Geometrically, the area covered by a directional antenna with parameters $\alpha, \rho, \bar{r}$ is a set of points with polar coordinates (r, θ) such that $r \leq \bar{r}$ and $\alpha \leq \theta \leq \alpha + \rho$. Given these inputs, the capacitated covering problem considered here is to cover all the customers with the minimum number of directional antennas such that the demands of customers assigned to an antenna stays within a bound. We consider two settings of this capacitated cover problem arising in wireless networks. In the first setting where the antennas have variable angular range, we present an approximation algorithm with ratio 3. For the setting where the angular range of antennas is fixed, we improve this approximation ratio to 1.5.

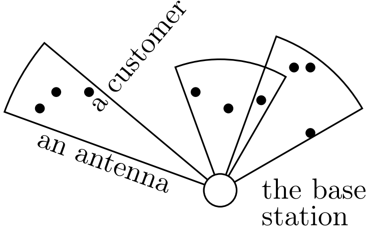
These results also apply for a related problem of bin packing with deadlines. In this problem we are are given a set of items, each with a weight, arrival time and deadline, and we want to pack each item into a bin after it arrives but before its deadline. The objective is to minimize the number of bins used. We present a 3-approximation algorithm for this problem, and 1.5-approximation algorithm for the special case when each difference between a deadline and the corresponding arrival time is the same.

1 Introduction

In the classic bin packing problem (see [6] for a survey) we are given a set of objects $\{1, \ldots, n\}$, each with a *weight* d_i, and the goal is to have a partition into as few parts as possible such that elements of each part can be placed in a single bin, *i.e.*, that have weights not exceeding some fixed bin capacity. On the other hand there are geometric covering problems, where we have a set of points in the plane and we want to cover them with minimum number of objects of some shape (see for e.g. [2, 5, 10]).

In this paper we consider problems in which a valid solution must form a valid packing and a valid cover. The shapes of sets we consider are those of possible ranges of directional antennas. Similar capacitated problems have been studied in the past in the context of set cover [17], vertex cover [4, 9], and rectangle stabbing [7]. In our case (as was in [9]), the capacities are *soft*. This means that multiple copies of a set (antenna) can be used with the multiplicities being counted in the size of solution.

These problems have become significant due to the increasing popularity of directional antennas in cellular and Wi-Fi communication networks [1, 3, 13, 15, 18]. Unlike conventional antennas, which transmit in all directions simultaneously, these directional antennas can transmit along a narrow beam in a particular direction. Additionally, with directional antennas, independent communication between nodes can take place in parallel, even if they are within range of each other [15]. Directional antennas yield higher throughput, lower interference, and better energy-efficiency [1, 13, 18].



In this paper, we consider challenges introduced in optimal positioning of directional antennas in infrastructure-based wireless networks (such as cellular networks or IEEE 802.11 WLANs). In our setting we have a base station that is connected to a wide-area network infrastructure, and we want to transfer information to and from a number of customers that can be reached by antennas placed on this base station. The base station functions as a gateway relaying traffic to and from wireless subscribers. The customers using devices like laptops, PDAs may have different demands on the transmission capacity, e.g., they may subscribe to connections with various bandwidth. The customers have fixed positions and we can choose how to orient our antennas. Each customer is assigned to an antenna (demands are unsplittable).

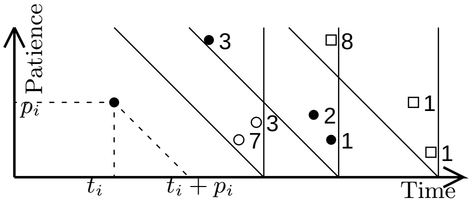
In the MinAntVar problem the antennas are programmable. We can choose the angular range of an antenna, so when the range is narrower it can reach further. We have a limit on the total bandwidth demand that can be assigned to an antenna and we want to minimize the number of antennas. We present an approximation algorithm for MinAntVar with ratio 3.

If the angular range of antennas are fixed, the problem becomes basically 1-dimensional. We minimize the number of antennas, problem MinAnt, and for a fixed number of antennas we also consider the dual objective of minimizing the maximum demand assigned to a single antenna, problem MinAntLoad. The problem of MinAntLoad is useful in situations where the antenna capacity is not rigid. For both problems we present approximation algorithms with ratio 1.5.

Our approximation results are provably tight for the MinAnt problem, because approximating it within a ratio better than 1.5 (even asymptotically) would imply solving Partition (for a definition refer Garey and Johnson [8]) in polynomial time. We can place a group of customers very close to each other with demands that form an instance of Partition: covering that group with 2 antenna sets means finding the positive answer to Partition; we can replicate this group m times, so that no antenna set contains customers from different groups. As a result, if for every case that can be solved with $2m$ antennas we can find a solution with fewer than $3m$ antennas we can solve Partition.

Our packing problems have another motivation. Suppose that a store wants to deliver some ordered goods. Assume that each ordered good has a weight d_i, arrival time t_i and it has to be shipped latest by $t_i + p_i$ (say that p_i is the *patience* of a customer). Given a capacity of a single shipment (truckload, container, etc.) we want to minimize the number of shipments. We call this problem BinSchedule.

If every p_i is equal to some p, then we have a problem that is essentially identical to MinAnt; shipment at time t can be used for customers with $t - p \leq t_i \leq t$. The only difference is that time is not circular, which makes inductive arguments easier. If the values of p_i's are arbitrary, we can easily adapt our algorithm for MinAntVar to provide approximate solutions. As a result, we can approximate BinSchedule to within a factor 1.5 in the case of fixed patience parameter and to within a factor 3 in the case of variable patience parameter.



Coordinates of an item i encode t_i & p_i. Each shipment has a capacity of 10. The numbers indicate the weights. The items that are to be shipped together must be enclosed by an angle.

Organization: In Section 2 we present problem definitions and notations. In Section 3 we present a simple scheme for transforming any algorithm for the set cover problem to an algorithm for the corresponding capacitated set cover problem. This provides simpler (but weaker) approximation algorithms for our problems. In Section 4 we describe a 3-approximation algorithm for the problem of MinAntVar. The basic idea here is to first construct a cover (of customers) with some special properties, and to then use these properties for efficient packing. In Section 5 we describe 1.5-approximation algorithms for the problems of MinAnt and MinAntLoad. The basic idea here is to show the existence of a sufficiently good solution in which customer assignments to antennas are constrained to some special form. In turn, a best solution in this form can be found by using dynamic programming (Appendix D).

2 Preliminaries

This section presents formal description of the problems, notions, and notations which will be used throughout this paper.

2.1 Problem Definitions

A point (r, θ) is equivalent to a Cartesian point $(r \cos\theta, r \sin\theta)$. Equality of angles is understood modulo 2π, *i.e.*, $\alpha = \beta$ means that for a certain integer q we have $\alpha = \beta + 2q\pi$. In each problem under consideration, our input contains a set of customers $U = \{1, \ldots, n\}$, customer i is located at point (r_i, θ_i) and has demand in $(0, 1]$. For a set of customers S, we define its *load* as $d(S) = \sum_{i \in S} d_i$.

Let $\rho(r)$ denote a decreasing function that describes the trade-off between the radius of the antenna range and its angular width. In the case of variable range antennas define

$$\mathcal{R}(\bar{r}, \alpha) = \{j \in U : \ r_j \leq \bar{r} \text{ and } \theta_j = \alpha + \beta \text{ with } 0 \leq \beta \leq \rho(\bar{r})\}$$

as the range of an antenna with parameters $\bar{r}, \alpha$. A valid solution to the MinAntVar problem is a partition $U = S_1 \cup \cdots \cup S_K$ such that for $k = 1, \ldots, K$ we have: $d(S_k) \leq 1$ and $S_k \subseteq \mathcal{R}(\bar{r}, \alpha)$ for some $\bar{r}, \alpha$. The objective is to minimize K, which is the number of different antennas we deploy to satisfy the demands of all customers.

When antennas are of fixed angular range (ρ is a constant), the range definition can be simplified. We now define range of an antenna as

$$\mathcal{R}(\alpha) = \{j \in U : \ \theta_j = \alpha + \beta \text{ with } 0 \leq \beta \leq \rho\}.$$

In this setting we consider two problems for both of which a valid solution is a partition $U = S_1 \cup \cdots \cup S_K$ such that for $k = 1, \ldots, K$ we have: $S_k \subseteq \mathcal{R}(\alpha)$ for some α. In problem MinAnt we require that $d(S_k) \leq 1$ and we minimize K. In problem MinAntLoad we fix K and allow overloading of the antennas. The objective here is to minimize the maximum load on any antenna, *i.e.*, we minimize $\max_k d(S_k)$. Note that solving the cover problem allows to find out when K is too small to give any feasible solution. For both problems we present approximation algorithms with ratio 1.5. Following definitions of Hochbaum *et al.* [11], MinAntLoad also provides a 1/2-dual approximation algorithm for MinAnt.

2.2 Notations

A sequence of angles $\alpha_1, \ldots, \alpha_k$ is circularly ordered if for a certain β we have $0 \leq \alpha_1 + \beta \leq \cdots \leq \alpha_k + \beta < 2\pi$. Define for customer i

$$\begin{aligned} R(i) &= \{j \in U: \ r_j \leq r_i \text{ and } \theta_j = \theta_i + \beta \text{ with } 0 \leq \beta \leq \rho(r_i)\}, \\ R^{rev}(i) &= \{j \in U: \ r_j \leq r_i \text{ and } \theta_j = \theta_i - \beta \text{ with } 0 \leq \beta \leq \rho(r_i)\}. \end{aligned}$$

Define $DoubleR(i) = R^{rev}(i) \cup R(i)$. Intuitively $R(i)$ consists of all customers whose angular separation from i is at most $\rho(r_i)$ in the counter-clockwise direction and $R^{rev}(i)$ represents the same in the clockwise direction.

Note that when antennas have fixed angular range, the radial distance of customers becomes insignificant. To reflect this, we modify definitions of $R(i)$ and $R^{rev}(i)$: we replace function $\rho(r_i)$ with constant ρ and drop the condition $r_j \leq r_i$.

We say customer i has a *large* demand if $d_i > 1/2$, otherwise i has a *small* demand. It will be convenient to use notation $L = \{i \in U: \ d_i > 1/2\}$ for the set of customers with large demands.

3 Capacitated Set Cover

A set system $(\mathcal{X}, \mathcal{R})$ is a set $\mathcal{X}$ with a collection of subsets $\mathcal{R}$. A set cover is a sub-collection of $\mathcal{C} \subseteq \mathcal{R}$ whose union is $\mathcal{X}$. Set cover problem is that of finding a cover $\mathcal{C}$ of minimum cardinality. In capacitated covering problem every element in $\mathcal{X}$ has a weight associated with it. We want to find a minimum cardinality covering such that every set P used in the solution must be a subset of some $Q \in \mathcal{R}$ and sum of weights of the elements in P is within some bound.

Suppose that an algorithm for the (uncapacitated) set cover gives us cover $\mathcal{C}$. We assign each element to one of the sets that contains it, subsequently we apply First-Fit method [6] of bin packing on each set. One can see that if the sum of weights of the elements is k times the bin capacity, we create at most $|\mathcal{C}| + 2k$ bins. Moreover, this transformation takes only linear time. This yields the following theorem

Theorem 3.1. *Let $(\mathcal{X}, \mathcal{R})$ be a set system. If there exists a c-approximation algorithm for the cover problem over the set system, then there exists a $c+2$-approximation algorithm for the corresponding capacitated cover problem.*

In the case of BinSchedule with fixed customer patience, the underlying sets are line intervals. The constraint matrix of the standard ILP formulation [16] for set cover has, for such sets, the property of consecutive ones (and thus total unimodularity [14]). Therefore the underlying set cover can be solved exactly with a greedy method. The case of MinAnt is almost the same, because its sets yield a constraint matrix with a similar property of circular ones (See [12]). By Theorem 3.1, these problems therefore have simple algorithms with approximation ratio of 3. However, in the case of of MinAntVar (or BinSchedule with variable customer patience), the related integer program has an *integrality gap*, *i.e.*, it has fractional solutions that have lower value of the objective function than the best integral solution. An example establishing this is formulated in Appendix A.

```
$\mathcal{C} \leftarrow I \leftarrow \varnothing$
while $\mathcal{C} \neq U$
    $i \leftarrow$ element of $(U \setminus \mathcal{C})$ with maximum $r_i$
    $I \leftarrow I \cup \{i\}$
    $\mathcal{C} \leftarrow \mathcal{C} \cup DoubleR(i)$
```

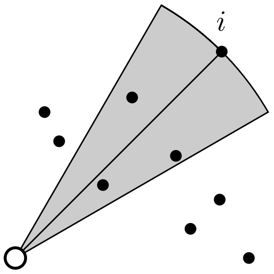
Figure 1: Creating independent sets in Phase 1 of the algorithm for MinAntVar

4 Covering with Variable Range Antennas

Due to Theorem 3.1, a 4-approximation algorithm exists for MinAntVar as the associated covering problem has a simple 2-approximation (Phase 1). In this section we describe an algorithm (divided over three phases) that achieves a 3-approximation for MinAntVar. All the proofs omitted in this section can be found in Appendix B.

4.1 Phase 1 of the Algorithm — Independent Set

We start by forming a (uncapacitated) cover of the set of all customers with antenna ranges, we will also create an independent set of customers I; more precisely, we will assure that no two different customers in I may belong to the same antenna range. The method is described in Fig. 1. Note that similar covering for BinSchedule can be constructed by using two shipments to cover all the customer points that can be covered with the least patient customer.



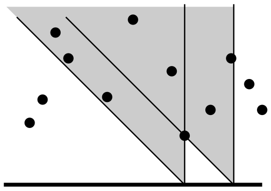
Antennas $R(i)$ and $R^{rev}(i)$ for the furthest customer i.

The ranges $R^{rev}(i)$ and $R(i)$ are defined in such a way that if some customer j has $r_j \leq r_i$ and i, j can be together in an antenna range, then $j \in R^{rev}(i) \cup R(i) = DoubleR(i)$. Thus it is easy to see that I is indeed an independent set of customers, so we need at least $|I|$ antenna ranges to cover all customers, while we have found a cover that uses $2|I|$ antenna ranges.

It will be convenient to duplicate customers in set I, so for $i \in I$ we create customer $n + i$ with $(r_{n+i}, \theta_{n+i}, d_{n+i}) = (r_i, \theta_i, 0)$. Then we make a substitution to make U and I disjoint:

$$I \leftarrow \{n + i : \ i \in I\}.$$

We call I the set of independents and use independent j to indicate $j \in I$.



Two shipments for the least patient customer.

4.2 Phase 2 of the Algorithm — Maximum Flow

We define a directed network as follows: there are 4 layers of nodes:

layer 0:	r,	the source,
layer 1:	U,	the set of customers,
layer 2:	I,	the set of independents,
layer 3:	s,	the sink.

Arcs and its capacities are defined as follows for $i \in U, j \in I$:

(r, i) always exists,	$c_{r,i} = d_i$,
(i, j) exists iff $i \in DoubleR(j)$,	$c_{i,j} = d_i$,
(j, s) always exists,	$c_{j,s} = 1$.

In this network we find a maximum flow f and a corresponding cut C (defined as the set of vertices containing r, but not s) with the minimum capacity. We normalize f and C to assure certain desirable properties that will be used to define the partition of the customers and to analyze the approximation ratio.

Given a flow f and network arc (u, v), let $f_{u,v}$ denote the net flow from u to v. We denote value of the flow f by $|f|$. Let $Partial$ be the set of undirected edges $\{a, b\}$ such that for a network arc (a, b) we have $0 < f_{a,b} < c_{a,b}$. Our first normalization criterion is that $Partial$ is acyclic.

Lemma 4.1. *Given a valid flow f, we can in polynomial time find the flow f' such that (a) values of f' and f are the same ($|f| = |f'|$), (b) Partial defined by f' is acyclic, and (c) if for any network arc (u, v) we have $f_{u,v} = 0$ or $f_{u,v} = c_{u,v}$, then $f_{u,v} = f'_{u,v}$.*

The process described by Lemma 4.1 will be called *acyclic normalization*. We will say that flow f is *bad* if there exists network nodes g,h,j and arcs $(g, j), (h, j)$ such that g is a large demand customer ($d_g > 1/2$), h is a small demand customer ($d_h \leq 1/2$), j is an independent, $0 < f_{g,j} < d_g$, and $f_{r,h} = 0$. Our second normalization criterion is that there always exists a flow that is not bad.

Lemma 4.2. *Given a valid flow f, we can in polynomial time find the flow f' such that (a) values of f' and f are the same ($|f| = |f'|$), (b) Partial defined by f' is acyclic, and (c) f' is not bad.*

Now we normalize the cut C. If an arc (i, j) exists and $j \notin C$, then we insist that $i \notin C$. Otherwise we remove i from C, and then arcs of the form $(i, j'), j' \in I$ cannot anymore contribute to the capacity of C. While now arc (r, i) makes the contribution of d_i to the capacity of C, and at least one arc (i, j) did before, we can improve our criterion (decrease the number of arcs violating the condition) with no increase to the capacity of C.

4.3 Phase 3 of the Algorithm — Assignment of Customers

Let f be a maximum flow in the network such that: (a) $Partial$ defined by f is acyclic, and (b) f is not bad. Let C be the corresponding normalized cut. We now define rules to assign customers to the independents in I.

Rule 1: If $f_{i,j} = d_i$, then customer i is assigned to an independent j.

To formulate the second rule, we form a bipartite graph (V, I, A), where A contains arcs (i, j) such that $\{i, j\} \in Partial$ and $V \subseteq U$ consists of starting points of arcs in A.

Lemma 4.3. *The bipartite graph (V, I, A) contains a matching M of size $|V|$.*

Rule 2: If $(i, j) \in M$, then customer i is assigned to an independent j, where M is some matching obtained from Lemma 4.3.

If $f_{r,i} > 0$, then customer i is assigned to an independent from I either by Rule 1 or by Rule 2. Thus it remains to handle cases when $f_{r,i} = 0$.

Rule 3: If $f_{r,i} = 0$, then customer i is assigned to an arbitrary independent j such that (i, j) is an edge in the network (equivalently, such that $i \in DoubleR(j)$).

If the customers assigned to a single antenna have their sum of demands greater than 1, then we introduce more antennas with exactly the same range. At this point we have instances of the bin packing problem. In the analysis it is important that we pack according to Rule 1 first, then according to Rule 2, then according to Rule 3, and we try to fit them in already open bins (so-called Any Fit rule).

4.4 Assignment of Potentials and the Analysis of the Algorithm

We assign potentials to customers and show that if the sum of potentials is P, any solution must use at least P antennas and we use at most $3P$. For this we will utilize the minimum cut C and its normal property.

Classification of customers: For i in Layer 1, if $i \in C$ we put i into the class of *Free customers* (Fc), and if $i \notin C$ we put i into the class of *Busy customers* (Bc). Note that if i is in Bc, then arc (r, i) contributes to the capacity of C and thus $f_{r,i} = d_i$. A customer in Fc may have a smaller flow from r.

Classification of independents in I: For j in Layer 2, if $j \in C$ we put j into the class of *Busy independents* (Bi), and if $j \notin C$ we put j into the class of *Free independents* (Fi). Again, if j is in Bi, then $f_{j,s} = 1$, and if j is in Fi, then j may have a smaller flow to s.

Because C is normalized, we have no arcs from customers in Fc to independents in Fi. Also, because our flow had the value equal to the capacity of C, each path of the flow was crossing exactly one edge contributing to the capacity of C, as a result we have no assignments (marked by positive flow) between customers in Bc and independents in Bi.

Initial distribution of potential: If a customer i is in Fc it has a potential of $2d_i/3$, and if it is in Bc then it has a potential of $d_i/3$. Similarly, every independent in Fi has a potential 2/3, and every independent in Bi has a potential of 1/3. The potential of a set (antenna) is the sum of potentials of customers assigned to it.

Let S' be a set in a valid solution to MinAntVar. Let S be the union of S' and the duplicate element of S' in I (if any). The first observation is that the sum of potentials of the elements of S does not exceed 1. If S contains no independent from I, then potential of S comes only from the customers whose demands add to at most 1 and the respective potentials add to at most 2/3. If S contains an independent from Bi, we increase this estimate by 1/3, but this is still at most 1. If S contains an independent from Fi, then all of other customers in S are in Bc, hence the customers in Bc contribute at most 1/3 to the potential. Therefore it is still the case that the potential of S is at most 1.

This proves that the size of the optimum partition is at least as large as the sum of all the potentials. Now it suffices to show that to every antenna we introduce, we can assign a potential of at least 1/3. In the remaining part of the analysis we show that this is indeed the case.

Redistribution of potential from Busy customers: A customer in Bc has assignment to an independent in Fi only. We give the potential of the customer to the independent to which it was assigned by Rule 1 or Rule 2. Note that assignment by Rule 3 is not possible here (Rule 3 only assigns Free customers).

Balancing potential of Free independent j: If j receives a load of less than 1 from the assigned customers, then it has to open at most 2 antennas (one ranging over $R(j)$, other over $R^{rev}(j)$) and for that j's own potential of 2/3 is adequate. Otherwise, if it receives a load exceeding 1 then it has an extra 1/3 of potential, enough for a single new antenna. Note that all assignments according to Rule 1 definitely fit in two antennas, and that the single assignment according to Rule 2 has to fit in one antenna as well. Thus j has to justify at most 3 antennas and potential 1 to do so.

Redistribution of potential from Free customers: A customer in Fc has assignment to an independent in Bi only. If it was assigned according to Rule 1 or Rule 2, we collect the customer's potential to a "central location". Now the total sum of all collected potentials is at least as large as the flow from the independents in Bi to s, and the flow is saturated. Therefore the total sum of all collected potentials is at least equal to the size of Bi and we can assign potential 2/3 from the central location to each independent in Bi. If a customer in Fc was assigned to an independent j according to Rule 3, then the customer brings its potential to j.

Balancing potential of Busy independent j: Because of the redistribution, every independent in Bi has potential 1 before we start applying Rule 3 (it had a potential of 1/3 from the beginning and another 2/3 was assigned during redistribution).

Case 1: Rule 2 assigned to j some customer i with $d_i > 1/2$. Now j uses at most two antennas (one ranging over $R(j)$, other over $R^{rev}(j)$) to accommodate all customers assigned to it by Rule 1, and at most one more antenna to accommodate i, and potential 1 to justify these three antennas. Because flow f is not bad, every k that can be assigned to j by Rule 3 has $d_k > 1/2$, and thus it brings potential of $2d_k/3 > 1/3$ that suffices to justify a new antenna. Therefore all antennas get a potential of at least 1/3.

Case 2: Rule 2 assigned to j some i with $d_i \leq 1/2$ (if it assigned nothing, pretend that it assigned some i with $d_i = 0$). Without loss of generality, we assume that $i \in R^{rev}(j)$. Let Rule 1 assign a load of b to an antenna ranging over $R^{rev}(j)$ and c to an antenna ranging over $R(j)$. Note that $b + c \leq 1$.

Case 2.1: $\mathbf{c \leq 1/2}$. In this case j already has a potential of 1 to justify for at most three antennas opened together from Rule 1 and Rule 2. We first consider the set of antennas ranging over $R^{rev}(j)$. Now when Rule 3 was applied, Any Fit (possibly) created new antennas ranging over $R^{rev}(j)$, but at most one of these can be loaded to less than 1/2, say with load x. Each of the antennas assigned with a load greater than 1/2 bring in a potential of at least 1/3 that suffices to justify them. Moreover in this case, Any Fit had to increase the load of the antenna that accommodated i to $1-x$, thus by at least $1/2-x$. This unaccounted load of $1/2-x$ can be used to assign a potential of $2/3(1/2 - x + x) = 1/3$ to the under-loaded antenna. Remember that antenna containing i already had obtained a potential of 1/3.

The same argument applies to the set of antennas ranging over $R(j)$ created by Any Fit, as we start with an antenna loaded with $c \leq 1/2$. Thus all the antennas opened by Any Fit have potential of at least 1/3.

Case 2.2: $\mathbf{c > 1/2}$. This implies that $b \leq 1/2$ and therefore $d_i + b \leq 1$. In this case j packs load of $d_i + b$ together, and thus before Rule 3 is applied there exists an antenna ranging over $R^{rev}(j)$ with a spare capacity of $p = 1 - d_i - b$. The other antenna ranging over $R(j)$ and to which a load of c was assigned, has a spare capacity of $q = 1 - c$. Since $d_i + b + c \leq 3/2$, we get $p + q \geq 1/2$. Since now only 2 antennas were opened, j has a spare potential of 1/3.

We again start off by concentrating on the set of antennas ranging over $R^{rev}(j)$. As in the previous case, when Rule 3 was applied, Any Fit (possibly) created new antennas ranging over $R^{rev}(j)$, but at most one of these can be loaded to less than 1/2, say with load δ. Let ε be the load added to the antenna that initially had spare capacity p. In this case, we use this unaccounted load of $\varepsilon + \delta$ to justify the under-loaded antenna (remaining antennas already have been assigned a load greater than 1/2, and thus a potential of at least 1/3). Because $1 - p + \varepsilon + \delta > 1$, clearly $\varepsilon + \delta > p$. If $\varepsilon + \delta \geq 1/2$, then there is already a potential of at least $2/3(\varepsilon + \delta) \geq 1/3$ to justify the under-loaded antenna. Otherwise, we may have a temporary deficit of potential of value no more than $2/3(1/2 - p)$.

Similar reasoning applied to the activities of Any Fit with $R(j)$ shows that a temporary potential deficit may be created, but of value no more than $2/3(1/2 - q)$. The sum of the two deficits is thus at most $2/3(1 - p - q) \leq 2/3 \times 1/2 = 1/3$, and we can use 1/3 of spare potential. Thus all the antennas opened by Any Fit have potential of at least 1/3.

Conclusion: Our potential analysis has shown that if the sum of potentials is P, any solution must use at least P antennas. Since our solution uses at most $3P$ antennas, we get an approximation ratio of 3. Our algorithm runs in polynomial time, its most time consuming phase is finding the maximum flow. Subsequent normalization amounts to no more than a standard augmentation process that is a part of a maximum flow algorithm. Similarly, finding a maximum matching can be done by finding a maximum flow in a very similar network. We may conclude this section with the following theorem and its subsequent corollary:

Theorem 4.4. *There exists a polynomial time approximation algorithm for the problem of* MinAntVar *with ratio* 3.

Corollary 4.5. *There exists a polynomial time approximation algorithm for the problem of* BIN-SCHEDULE *with ratio* 3.

5 Covering with Fixed Range Antennas

In this section we describe 1.5-approximation algorithms for both MINANT and MINANTLOAD. A 2-approximation for MINANT is given by a simple greedy approach of packing consecutive customers into an antenna till either the antenna capacity overflows, or the maximum angular range (ρ) is reached (details omitted in this extended abstract).

For MINANTLOAD, we consider the following reformulation: the input consists of the set of customers, the angle ρ, and $\mathcal{D}$, the (guessed) maximum load of a set in a valid solution with K sets. Given a correct guess, we promise to find a valid solution with K sets in which the maximum load of a set is at most $1.5\mathcal{D}$. Since we can test if the guess is too small or too large for the given K, we can efficiently obtain a value for $\mathcal{D}$ which is at most factor 1.5 away from the optimal $\mathcal{D}$. To handle both MINANT and MINANTLOAD together we rescale the customer demands to set $\mathcal{D} = 1$.

The algorithms are identical for both the problems except for a few minor differences. The common idea is to show the existence of a sufficiently good solution in which the sets are constrained to a very special form. Loosely speaking, in every set of this special form the small demand customers are contiguous, and the large demand customer (if any) also satisfies certain rules. In turn, the best solution in this special form can be found using dynamic programming.

Our reasoning could be a bit simpler if we could have defined a linear order of all customers such that allowed antenna ranges are intervals in that order. However, even though we cannot introduce a consistent global order we can define local ordering which is good enough for our purposes. In this preliminary version, we omit the case where $\rho \geq \pi$. This case is very close to bin packing and it does not pose any new challenge, but it requires a tedious case analysis. In the remainder of the section, we also assume w.l.o.g. that θ_i is different for each $i \in U$. All the proofs omitted in this section can be found in Appendix C. Recall as mentioned in Section 2.2, in the definitions of $R(i)$ and $R^{rev}(i)$, we replace function $\rho(r_i)$ with constant ρ and drop the condition $r_j \leq r_i$.

5.1 Relaxed Solutions, Normalization

To capture spatial closeness of customers, we define a directed graph

$$(U, Succ) \text{ where } (i,j) \in Succ \equiv R(i) \cap R^{rev}(j) = \{i,j\}.$$

Intuitively the existence of edge (i,j) in this graph implies that j follows i in the counter-clockwise direction. Note that if i and j are in distinct weakly connected components of $(U, Succ)$ then i and j cannot belong to the same $R(k), k \in U$; as a result we can join components into a single path and it is still the case that every $R(k)$ forms a path in $(U, Succ)$. Therefore in the rest of discussion, we can assume that $(U, Succ)$ forms a simple cycle. To visualize this graph one could assume w.l.o.g. that each customer is at unit distance from the origin on a cycle. This cycle oriented counter-clockwise defines the graph.

We can remove $U \setminus L$ or L from the cycle of $(U, Succ)$ to obtain induced graphs $(L, Succ_l)$ and $(U \setminus L, Succ_s)$. If $(i,j) \in Succ$ (respectively, $Succ_l$, $Succ_s$), we will use notation $j = Succ(i)$ (respectively, $j = Succ_l(i)$, $j = Succ_s(i)$).

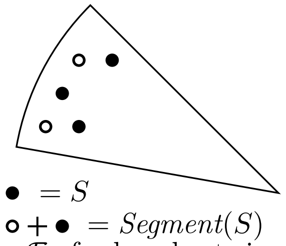
A *relaxed set* S is defined by variables $x_{S,i}$ for each $i \in U$ that describe to what extent i belongs to S, *i.e.*, $x_{S,i} \in [0,1]$. A *normal set* T can be viewed as a special case of a relaxed set such that $x_{T,i} \in \{0,1\}$. We will use this previous notation for normal sets L and $U \setminus L$. We also extend the notation of relaxed sets to $S \setminus L$ and $S \cap L$ with the following formulae:

$$x_{S \setminus L,i} = x_{U \setminus L,i} \cdot x_{S,i} \qquad \text{and} \qquad x_{S \cap L,i} = x_{L,i} \cdot x_{S,i}.$$

We now define for a set S

$$\begin{aligned} \widehat{S} &= \{i \in U : x_{S,i} > 0\}, \\ \textit{Segment}(S) &= R(i) \cap R^{rev}(j) \text{ if } i,j \in \widehat{S} \text{ and } \widehat{S} \subseteq R(i) \cap R^{rev}(j), \\ \textit{Seg-small}(S) &= \textit{Segment}(S \setminus L). \end{aligned}$$

Intuitively for a set S, its *Segment* contains all the customers lying geographically between first and last customers of S in a counter-clockwise ordering. *Segment*(S) is defined for a set S iff any two customers in S are within ρ of each other. The set *Seg-small*(S) is a subset of *Segment*(S) restricted to small demand customers. We also define *first-small*(S) as $j \in S$ such that, for some $k \in S$ we have *Seg-small*$(S) = R(j) \cap R^{rev}(k)$. A collection $\mathcal{F}$ of relaxed sets is a relaxed solution to MinAnt or MinAntLoad if



$$\begin{aligned} \textit{Segment}(S) \quad &\text{is} \quad \text{defined for each } S \in \mathcal{F}, && (1) \\ \sum_{S \in \mathcal{F}} x_{S,i} &= 1 \text{ for each } i \in U, && (2) \\ d(S) = \sum_{i \in U} d_i x_{S,i} &\leq 1 \text{ for each } S \in \mathcal{F}. && (3) \end{aligned}$$

A relaxed set S is *compact* if: (a) $0 < x_{S,i} < 1$ for some i implies that *Seg-small*$(S) = R(j) \cap R^{rev}(k)$ and $i \in \{j,k\}$, and (b) $x_{S,i} > 0$ may be true for at most one $i \in L$ and if it is true, $x_{S,i} = 1$. If a set S is compact relaxed and $x_{S,i} = 1$ for $i \in L$, then we use the notation $i = L(S)$. Our first normalization criterion is that there exists a relaxed solution where all sets are compact.

Lemma 5.1. *Any solution F to an instance of* MinAnt *or* MinAntLoad *can be replaced with a relaxed solution $\mathcal{F}$ in which each $S' \in F$ is replaced with a compact relaxed set $S \in \mathcal{F}$ such that (a) $S \cap L = S' \cap L$, (b) $d(S \setminus L) = d(S' \setminus L)$, and (c) Seg-small$(S) \subseteq$ Seg-small(S').*

Below we will use the term *compact solution* to describe a relaxed solution that contains only compact relaxed sets.

Defining Succession: We remove from consideration every set S for which *Segment*$(S) = \{L(S)\}$. To convert a compact solution $\mathcal{F}$ into an approximate solution we first define succession of sets that have at least one small demand customer. For $i \in U \setminus L$, define $\bar{\mathcal{F}}_i = \{S \in \mathcal{F} : \textit{Seg-small}(S) = \{i\}\}$. If $|\bar{\mathcal{F}}_i| = 1$ then the question of ordering $\bar{\mathcal{F}}_i$ is moot. Otherwise $0 < x_{A,i} < 1$ for each $A \in \bar{\mathcal{F}}_i$, and hence $L(A)$ is defined. There exists a path (that is not a cycle) in $(U, \textit{Succ})$ that contains every such $L(A)$ and we determine the succession according to the precedence of $L(A)$'s on this path. Now lets assume that (p,q,r) is a path in $(U \setminus L, \textit{Succ}_s)$, and let $\mathcal{F}_q = \{S \in \mathcal{F} : x_{S,q} > 0\}$. Note that $x_{B,p} > 0$ is true for at most one B in $\mathcal{F}_q$, if such a set exists, it starts the ordering of $\mathcal{F}_q$; similarly if $x_{C,r} > 0$ for $C \in \mathcal{F}_q$ then C ends the ordering of $\mathcal{F}_q$, the rest of this ordering is the ordering of $\bar{\mathcal{F}}_q$. Finally, if $\mathcal{F}_p$ and $\mathcal{F}_q$ are disjoint, we have a succession from the last set of $\mathcal{F}_p$ to the first set of $\mathcal{F}_q$.

Rounding to an Approximate Solution: To *round* $\mathcal{F}$ we split it into paths of the succession order where a pair of successive sets (S,T) is connected if *Seg-small*$(S) \cap$ *Seg-small*$(T) = \{i\}$ for some i, with i as *junction*. A path spanning $k+1$ sets has k junctions. To get a solution for MinAnt, we create sets that contain consecutive pairs of junctions. If we have $2\ell + 1$ compact relaxed sets, we add ℓ new sets, and if we have 2ℓ compact relaxed sets, we add $\ell - 1$ sets with two junctions each and 1 set with just one junction. If our path of sets forms a full cycle, we first add each junction to the preceding set, and at least one set will still have $d(S) \leq 1$; we remove this set from consideration and handle the remaining sets. It is simpler to obtain a solution for MinAntLoad in which every $d(S)$ is at most 1.5: we allocate each junction to its preceding set.

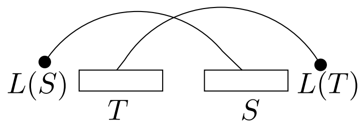


Figure 2: A possible scenario of strong crossing between sets S and T.

To find a rounding of a compact solution we will use dynamic programming. However, to assure that we do not have to consider more than a polynomial number of possibilities we restrict the structure of the allowed solution even further.

Dealing with Large Demand Customers: We define a symmetric relation *crosses* on compact relaxed sets. For any two compact relaxed sets S and T, this relation holds only if $\mathit{Segment}(S) \cap \mathit{Segment}(T) \neq \emptyset$. In that case, we use the notation $\mathit{path}(S,T) = (i_1, \ldots, i_\ell)$ to denote the path in (U, Succ) that connects the elements in $\mathit{Segment}(S) \cup \mathit{Segment}(T)$ (we also use the notation $\mathit{path}(S) \equiv \mathit{path}(S,S)$). Firstly S crosses T if $\mathit{Segment}(S) \subset \mathit{Segment}(T)$. Additionally, S crosses T if for some $b < a$ and $c < d$ we have $L(S) = i_a$, $L(T) = i_b$, $i_c \in \mathit{Seg\text{-}small}(S)$ and, $i_d \in \mathit{Seg\text{-}small}(T)$. Also, if S crosses T and $\{i_1, i_\ell\} \not\subseteq U \setminus L$ we say that S *strongly crosses* T. See also Fig. 2.

Given a compact solution $\mathcal{F}$, we consider the graph ($\mathcal{F}$,crosses). Our goal is to limit the connected components of this graph to a very special form. Over the next three lemmas we achieve this by showing there exists a compact solution with no strong crossing.

Lemma 5.2. *If a relaxed set $S \in \mathcal{F}$ strongly crosses $T \in \mathcal{F}$ then either Seg-small(S) $\subset$ Segment(T) or Seg-small(T) $\subset$ Segment(S).*

Remark. In the above proof we had the implication: if S crosses T and the first or the last element on $\mathit{path}(S,T)$ equals $L(T)$, then $\mathit{Seg\text{-}small}(S) \subset \mathit{Segment}(T)$.

Lemma 5.3. *Assume that $R, S, T \in \mathcal{F}$, S strongly crosses T, path(S,T) $= (i_1, \ldots, i_\ell)$, and that for some a, b, c we have $1 \leq a < b < c \leq \ell$, $i_a \in$ Seg-small(S), $i_b \in$ Seg-small(R), $i_c \in$ Seg-small(T). Then R strongly crosses S or T.*

Lemma 5.4. *A compact solution $\mathcal{F}$ to* MinAnt *can be replaced with another compact solution that has the same number of compact relaxed sets and in which no set strongly crosses another set.*

Now we can assume that no two sets of our compact solution ($\mathcal{F}$) strongly cross each other and we want to characterize the connected components of the graph ($\mathcal{F}$,crosses). We will ignore sets with empty $U \setminus L$, and the remaining ones we classify as follows: $L(S)$ does not exists — S is *small*, $L(S)$ starts $\mathit{path}(S)$ — S is *left*, $L(S)$ ends $\mathit{path}(S)$ — S is *right*, $L(S) \in \mathit{Seg\text{-}small}(S)$ — S is *straight*. Under this notion the set S in Fig. 2 is a right set, whereas the set T in the figure is a left set. The following two lemmas helps us in characterization. In the following lemma we will use the term *interior* of a path to denote the sub-path formed by deleting the endpoints of the path.

Lemma 5.5. *Assume that we have a pair of successive sets (S,T) such that S is not right and T is not left. Let i be the end of path(S). Then there exists no $Q \in \mathcal{F}$ such that i is in the interior of path(Q).*

Lemma 5.6. *Assume that no pair of successive sets in $\mathcal{F}$ satisfies the assumptions of Lemma 5.5. Then either all sets in $\mathcal{F}$ are left or all of them are right.*

If the assumptions of Lemma 5.5 are not satisfied, then the assumptions of Lemma 5.6 are satisfied and we use a greedy packing (see Appendix D) to find an approximate solution. Otherwise, for every pair of successive sets S, T such that S is not right and T is not left we break the path of $(U, Succ)$ at i selected according to Lemma 5.5. If $i \in$ *Seg-small*(T), then we split i into two elements, i' and i'', with $d_{i'} = d_i x_{S,i}$ and $d_{i''} = d_i x_{T,i}$. The element i' is assigned to S, whereas i'' is assigned to T. A point to note is that during this process for no crossing set pair (P, Q) is the *path*(P, Q) altered. Finally after all the breaking, in every (remaining) pair of successive sets (A, B) either A is right or B is left. A path of successor pairs may have some number of right sets, followed by a straight or small set (if any), followed by a number of left sets. Every crossing occurs within a single path.

Another observation is that a crossing between two right (left) sets is a strong crossing. Thus a right set may cross left sets only and possibly the straight set in the middle of the path. Note that if it crosses any of the left sets, it crosses all left sets that are before it. Therefore right sets that participate in crossings form the initial portion of the sub-path, while the left sets that participate in crossings form the final part. As a result, the sets participating in crossings form a contiguous sub-path, with the only possible exception for the separating straight set, if the latter exists. In turn, the large demand customers ($L()$'s) of the sets that participate in crossings form a contiguous path P in $(L, Succ_l)$. See also Fig. 5 in Appendix D.

This finishes the normalization's that the dynamic programming algorithm exploits.

5.2 Algorithms for MinAnt and MinAntLoad

In Appendix D, we briefly sketch an algorithm for finding a solution to MinAntLoad that is a rounding of a compact solution with no strong crossing and is within a factor 1.5 from the optimum. The algorithm uses dynamic programming if the assumptions of Lemma 5.5 are satisfied, otherwise it uses greedy packing. The algorithm for MinAnt is very similar and is omitted in this extended abstract. We may conclude this section with the following theorem and its subsequent corollary:

Theorem 5.7. *There exists polynomial time approximation algorithms for the problems of* MinAnt *and* MinAntLoad *with ratio* 3/2.

Corollary 5.8. *If all customers have fixed patience, then there exists a polynomial time approximation algorithm for the problem of* BinSchedule *with ratio* 3/2.

6 Concluding Remarks

An obvious open problem resulting from this work is to improve the approximation ratio for the problem MinAntVar; although different techniques than those seem to be required. Another interesting question is to investigate the extension of our work to a setting where there are multiple base stations.

7 Acknowledgments

We thank Sudarshan Vasudevan for suggesting the problem and initial discussions. We also thank Webb Miller for supporting Jieun Jeong through NIH grant HG02238 and Martin Fürer for supporting Shiva Kasiviswanathan through NSF Grant CCR-0209099.

References

[1] L. Bao and J.J. Garcia-Luna-Aceves, *Transmission scheduling in ad hoc networks with directional antennas*, Proceedings of ACM MOBICOM '02, 2002, pp. 48–58.

[2] Hervé Brönnimann and Michael T. Goodrich, *Almost optimal set covers in finite VC-dimension*, Discrete & Computational Geometry **14** (1995), no. 4, 463–479.

[3] R. Choudhury, X. Yang, R. Ramanathan, and N. Vaidya, *Using directional antennas for medium access control in ad hoc networks*, proceedings of the ACM MOBICOM '02, 2002, pp. 59–70.

[4] Julia Chuzhoy and Joseph (Seffi) Naor, *Covering problems with hard capacities*, SIAM Journal on Computing **36** (2006), no. 2, 498–515.

[5] Kenneth L. Clarkson and Kasturi R. Varadarajan, *Improved approximation algorithms for geometric set cover*, Proceedings of ACM SOCG '2005, 2005, pp. 135–141.

[6] Edward G. Coffman, Michael R. Garey, and David S. Johnson, *Approximation algorithms for bin packing: A survey*, Approximation algorithms (D. Hochbaum, ed.), PWS Publishing Company, 1997.

[7] Guy Even, Dror Rawitz, and Shimon Shahar, *Approximation algorithms for capacitated rectangle stabbing*, Proceedings of CIAC '06, vol. 3998, Springer, 2006, pp. 18–29.

[8] Michael R. Garey and David S. Johnson, *Computers and intractability*, W.H. Freeman and Co., New York, 1979.

[9] Sudipto Guha, Refael Hassin, Samir Khuller, and Einat Or, *Capacitated vertex covering*, Journal of Algorithms **48** (2003), no. 1, 257–270.

[10] Dorit S. Hochbaum and Wolfgang Maass, *Approximation schemes for covering and packing problems in image processing and VLSI*, Journal of the ACM **32** (1985), no. 1, 130–136.

[11] Dorit S. Hochbaum and David B. Shmoys, *Using dual approximation algorithms for scheduling problems: Theoretical and practical results*, Journal of the ACM **34** (1987), no. 1, 144–162.

[12] John J. Bartholdi III, James B. Orlin, and H. Donald Ratliff, *Cyclic scheduling via integer programs with circular ones*, Operations Research **28** (1980), 1074–1085.

[13] C. Peraki and S. Servetto, *On the maximum stable throughput problem in random networks with directional antennas*, proceedings of the ACM MobiHoc '03, 2003, pp. 76–87.

[14] A. Schrijver, *Theory of linear and integer programming*, John Wiley & Sons, 1986.

[15] A. Spyropoulos and C. S. Raghavendra, *Capacity bounds for ad-hoc networks using directional antennas*, Proceedings of IEEE ICC '03, 2003, pp. 348–352.

[16] Vijay V. Vazirani, *Approximation algorithms*, Springer-Verlag, 2001.

[17] Laurence A. Wolsey, *An analysis of the greedy algorithm for the submodular set covering problem*, Combinatorica **2** (1982), no. 4, 385–393.

[18] S. Yi, Y. Pei, and S. Kalyanaraman, *On the capacity improvement of ad hoc wireless networks using directional antennas*, proceedings of the ACM MobiHoc '03, 2003, pp. 108–116.

Appendix A: Integrality Gap for MinAntVar with Set Cover ILP

Assume that $n = 36m$ and $\alpha = \pi/18m$. Customer i is located at a point with polar coordinates $(r_i, i\alpha)$, where $r_i = 2$ if i is divisible by 3 (distant customer) and otherwise $r_i = 1$ (nearby customer). We set $\rho(r) = 4\alpha/r^2$.

First observation is that no two distant customers can be covered together, so we need $12m$ antennas to cover them, and in the process we can cover also $12m$ nearby customers, so it remains to cover the remaining $12m$ remaining nearby customers. We cannot cover more than 3 of them, the only way to have 4 nearby customers in a group of 5 consecutive customers is to have a pattern **n n d n n**, but in this case the antenna for the central distant customer already covers one of them (**n** denotes nearby customer and **d** denotes distant customer). Hence we need $4m$ additional antennas for the total of $16m$.

However, if we are allowed to select sets with coefficient 0.5, then we can select two "half-antennas" for each distant customer: one that covers it together with the preceding nearby customer, and one that covers it together together with the following nearby customer. That way after selecting $24m$ half-antennas all nearby customers are half-covered. Then we can half-cover them once more, 4 at the time, so we add $24m/4 = 6m$ half-antennas for the total of $30m$ half-antennas, and this is equivalent to $15m$ antennas.

A more elaborate example would allow additional improvements when coefficients lower than 0.5 are introduced. A similar example can also be constructed for BinSchedule with variable customer patience.

Appendix B: Proofs from Section 4

Proof of Lemma 4.1. If $\{a, b\} \in Partial$ then in the residual network defined by f it corresponds to two arcs with positive capacity, *e.g.,* (a, b) has capacity $c_{a,b} - f_{a,b}$ and (b, a) has capacity $f_{a,b}$. If we have a cycle in $Partial$, then we also have a directed cycle in the residual network with positive capacities on arcs, and if the minimum of these capacities is t, we can augment f by adding t to $f_{a,b}$ if (a, b) is on the directed cycle, and by subtracting t from $f_{a,b}$ if (b, a) is on the directed cycle. Afterward any arc on the cycle that had residual capacity t does not correspond to an edge in $Partial$ anymore. We can repeat the process until $Partial$ is acyclic. Note that the augmentations change flow values only in arcs of $Partial$ and flow values in other arcs are unaffected. Therefore by repeating this process we get a flow f' satisfying the stated conditions. ❑

Proof of Lemma 4.2. We will now describe possible steps of the normalization process. To estimate their number we will introduce two quantities, $\gamma(f)$ and $\beta(f)$, that we will use as counters.

$$\begin{aligned}\gamma(f) &= |\{i \in U : d_i \leq 1/2 \text{ and } f_{r,i} = d_i\}|,\\ \beta(f) &= |\{(i, j) : d_i > 1/2 \text{ and } 0 < f_{i,j} < d_i\}|.\end{aligned}$$

If flow f is bad we can modify it in such a way that $\gamma(f) - \beta(f)$ increases. Suppose that f is bad because of g, h and j.

Case 1: $f_{g,j} = t \leq d_h$: we increase both $f_{r,h}$ and $f_{h,j}$ from 0 to t, decrease $f_{r,g}$ by t, and decrease $f_{g,j}$ to 0. This does not decrease $\gamma(f)$, but decreases $\beta(f)$.

Case 2: $f_{g,j} > d_h$: we increase $f_{r,h}$ and $f_{h,j}$ from 0 to d_h (this increases $\gamma(f)$), and decrease both $f_{r,g}$ and $f_{g,j}$ by d_h. $\beta(f)$ is unaffected.

Therefore in both cases, we increase $\gamma(f) - \beta(f)$. Note that subsequent acyclic normalization does not decrease $\gamma(f) - \beta(f)$. Since this increase can be done only a finite number of times, we eventually get a flow f' that is not bad. ❑

Proof of Lemma 4.3. We will prove that (V, I, A) satisfies the assumption of Hall's theorem. For a set $X \subseteq V$ we define the set of neighbors of X, $N(X) = \{w : \exists x \in X \text{ s.t. } (x, w) \in A\}$. The assumption of Hall theorem states that $|N(X)| \geq |X|$.

We can break $X \cup N(X)$ into weakly connected components $X_1 \cup N(X_1)$, ..., $X_k \cup N(X_k)$, and then it suffices to prove the inequality for every $X_l, l \in \{1, \ldots, k\}$. Thus we may assume that the bipartite graph $(X, N(X), A)$ is weakly connected.

Now, it suffices to show that X is incident to at least $2|X| - 1$ arcs of A. We show that if X has $|X| - 1$ neighbors or less (i.e., $\sum_{x \in X} |N(\{x\})| < 2|X| - 1)$), we would have a cycle in $Partial$, a contradiction. Note that each term in this summation is at least 1, this implies that at least 2 of them are 1. Suppose then that for $x \neq y$, $x, y \in X$ we have i and j such that $N(\{x\}) = \{i\}$, and

```
$j \leftarrow 0$
$i \leftarrow \textit{first-small}(S'_1)$
$r \leftarrow d_i$
$D \leftarrow \textit{debt}(S'_1)$
do forever
    if $D > r$
    $x_{S_j,i} \leftarrow r/d_i$
        $D \leftarrow D - r$
        $i \leftarrow Succ_s(i)$
        $r \leftarrow d_i$
    else
        $x_{S_j,i} \leftarrow D/d_i$
        if $j = m$
            terminate
        $r \leftarrow r - D$
        $j \leftarrow j + 1$
        $D \leftarrow d(S'_j \setminus L)$
```

Figure 3: Algorithm Norm. The algorithm converts sets $S'_1, \dots, S'_m$ in a solution to compact relaxed sets. The variables $x_{S_0,i}$ do not correspond to any of the constructed relaxed sets.

$N(\{y\}) = \{j\}$. Then $f_{r,x} = f_{x,i} < d_x$ and $f_{r,y} = f_{y,j} < d_y$. Thus arcs of A show one path in $Partial$ from x to y, while we have another, (x, r, y), and so we have a cycle in $Partial$, a contradiction. ❑

Appendix C: Proofs from Section 5

Sketch of Proof of Lemma 5.1. Consider $S' \in F$, if $S' \subseteq L$ then we set $S = S'$. Let F_0 be the family of the remaining sets of F. We order F_0 as $S'_1, S'_2, \dots, S'_m$ so that in $(U, Succ)$ has a simple path with subsequence *first-small*(S'_1), *first-small*(S'_2), …, *first-small*(S'_m). The process of replacing each S'_i with S_i is described as an algorithm in Fig. 3.

Properties (a) and (b) follow straightforwardly from the algorithm, and so are the properties (2) and (3) of a valid relaxed solution. It remains to show that for some initial value of D we assure property (c) and thus property (1) of a valid relaxed solution.

For $S'_a, S'_b \in F$ define

$$\begin{aligned} \textit{Debt}(S'_a, S'_b) &= \begin{cases} \varnothing \text{ if } \textit{first-small}(S'_a) \notin \textit{Seg-small}(S'_b) \text{ or } S'_a = S'_b \\ \textit{Seg-small}(S'_a) \cap S'_b \cap (U \setminus L) \text{ otherwise,} \end{cases} \\ \textit{Debt}(S'_a) &= \bigcup_{S'_c \in F} \textit{Debt}(S'_a, S'_c), \\ \textit{debt}(S'_a) &= d(\textit{Debt}(S'_a)). \end{aligned}$$

With respect to a run of our algorithm, we also define *offset*(S'_k): let $(i_1, \dots, i_\ell)$ be the path in $(U \setminus L, Succ_s)$ that connects elements of *Seg-small*$(S'_k) \setminus L$, and assume that when the algorithm sets j to k we have $i = i_a$ and $r = r_0$; then *offset*$(S'_k) = r_0 + \sum_{h=1}^{a-1} d_{i_h}$.

It is easy to show that *offset*$(S'_j) =$ *debt*(S'_j) implies *offset*$(S'_{j+1}) =$ *debt*(S'_{j+1}). Thus if we set the initial value of D to *debt*(S'_1) then we have *debt*$(S'_j) =$ *offset*(S'_j) for $j = 1, \dots, m$.

One can also see that if *offset*$(S'_j) + d(S'_j \setminus L) \le d($*Seg-small*$(S'_j) \setminus L)$ then *Seg-small*$(S_j) \subseteq$ *Seg-small*(S'_j). Therefore it is enough to show that *debt*$(S'_j) + d(S'_j \setminus L) \le d($*Seg-small*$(S'_j) \setminus L)$.

Finally, $debt(S'_j) + d(S'_j \setminus L) \leq d(Seg\text{-}small(S'_j) \setminus L)$ because $Debt(S'_j)$ and $S'_j \setminus L$ are two disjoint subsets of $Seg\text{-}small(S'_j)$. ❑

Proof of Lemma 5.2. Let $path(S,T) = \{i_1, \ldots, i_\ell\}$, and let $i_\ell = L(T)$. If $Segment(S) \subset Segment(T)$ then clearly $Seg\text{-}small(S) \subset Segment(T)$ and the claim is true. Otherwise, $L(S) = i_a$ for some $a < \ell$, and therefore for some c, d we have $c < d$, $i_c \in Seg\text{-}small(T)$, and $i_d \in Seg\text{-}small(S)$. Because S is a compact relaxed set, $Seg\text{-}small(S) \subset \{i_c, i_{c+1}, \ldots, i_\ell\} \subset Segment(T)$. Therefore again $Seg\text{-}small(S) \subset Segment(T)$. ❑

Proof of Lemma 5.3. Because of symmetry, and by Lemma 5.2, we can assume that $Seg\text{-}small(S) \subset Segment(T)$. This implies that $L(T) = i_1$ and because R is a compact relaxed set, $Seg\text{-}small(R) \subset Segment(T)$. If $Segment(R) \subset Segment(T)$ then R strongly crosses T, so we can assume that $L(R) \notin Segment(T)$. We can now consider a path in $(U, Succ)$ that connects all elements of $Segment(S) \cup Segment(T) \cup \{L(R)\}$. If $L(R)$ follows i_1 on that path then again R strongly crosses T. If $L(R)$ precedes i_1, then $Seg\text{-}small(S) \subset Segment(R)$; if $Segment(S) \subset Segment(R)$, then S strongly crosses R, otherwise $L(S)$ follows $L(R)$ while $i_a \in Seg\text{-}small(S)$ precedes $i_b \in Seg\text{-}small(R)$, so once more, R strongly crosses S. ❑

Sketch of Proof of Lemma 5.4. For a relaxed set S, we define $length(S)$ as follows: if $Segment(S) = \{i\}$ then $length(S) = x_{S,i}$; if $|Segment(S)| \geq 2$, we can assume that $Segment(S)$ is connected in $(U, Succ)$ with a path $(p, \ldots, q)$, and we define $length(S) = |Segment(S) \setminus \{p, q\}| + x_{S,p} + x_{S,q}$. We will use the following characteristic of a relaxed solution: we place $length(S)$ for $S \in \mathcal{F}$ in non-increasing order that we call *ordered length vector*. We will show that if $\mathcal{F}$ contains some S, T that strongly cross each other then we can modify $\mathcal{F}$ in such a way that the old ordered length vector will lexicographically precede the newer one.

Suppose that $S, T \in \mathcal{F}$ and S strongly crosses T and $path(S,T) = (i_1, \ldots, i_\ell)$. If there exist $R \in \mathcal{F} - \{S, T\}$ and $f < g < h$ such that $\{i_f, i_h\} \subseteq Seg\text{-}small(S) \cup Seg\text{-}small(T)$ while $i_g \in Seg\text{-}small(R)$: we replace S, T with a strongly crossing pair R, S or R, T in the analysis. We repeat it as long the condition of the previous statement is satisfied. We now assume that sets S and T don't satisfy the above condition. Define a relaxed set $Q = (S \setminus L) \cup (T \setminus L)$ by $x_{Q,i} = x_{S \setminus L,i} + x_{T \setminus L,i}$. In $(U \setminus L, Succ_s)$ set $\widehat{Q}$ (remember that $\widehat{Q}$ is the set of all customers $a \in U$ with $x_{Q,a} > 0$) is connected by a path, say $(j_1, \ldots, j_m)$ ($m > 1$, otherwise S would not cross T), and because of our choice of S and T, we have $x_{Q,j_k} = 1$ for $k = 2, \ldots, m-1$.

By symmetry, we can assume that $\{i_1\} = T \cap L$ and thus by Lemma 5.2, $Seg\text{-}small(S) \subset Segment(T)$. This means that for some a we have $Seg\text{-}small(S) = \{j_1, \ldots, j_a\}$ and $Seg\text{-}small(T) \subseteq \{j_a, \ldots, j_m\}$. We will alter S and T using the Algorithm Norm from Fig. 3. The parameters to the algorithm are set as: $S'_1 = T$, $S'_2 = S$, $i = j_1$, and $D = (1 - x_{Q,j_1}) d_{j_1}$. The sets S_1 and S_2 returned by the algorithm are called T' and S', respectively.

One can see that $Segment(T')$ is a subset of the $Segment(T)$, and that $length(T')$ gets decreased by either $d(S \setminus L)$ or by x_{T,j_m}. Now we need to compare new set S' with the former set S.

If we had $Segment(S) \subset Segment(T)$, then $Segment(S')$ is also a subset of the $Segment(T)$, so S' is still a compact relaxed set, within the range of $R(i_1)$. Moreover, the $length(S')$ is at most $length(T) - 1$, as i_1 is not in $Segment(S)$. Therefore the decrease of the ordered length vector in the lexicographic order is determined by the decrease in $length(T')$.

Now assume $Segment(S) \not\subset Segment(T)$. Because $Seg\text{-}small(S) \subset Segment(T)$, we must have $i_\ell = L(S)$, and thus $Seg\text{-}small(T) \subset Segment(S)$. Now the same arguments that we used to compare T' with the T apply to the comparison of S' to S: $Segment(S')$ is contained in the $Segment(S)$, implying that $Segment(S')$ is contained in $R^{rev}(i_\ell)$, and $length(S')$ is decreased by at least $d(T \setminus L)$.

Since only a finite number of such decreases is possible, eventually we will obtain a compact

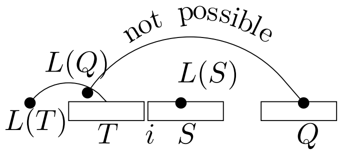


Figure 4: Illustration for the proof of Lemma 5.5. The set S is a straight set, whereas the set T is a right set.

solution without any strong crossings. ❑

Proof of Lemma 5.5. Since S is not a right set, i cannot be in the interior of *path*(S). Similarly since T is not a left set, i cannot be in the interior of *path*(T). Now suppose there exists a $Q \in \mathcal{F}$ such that i is in the interior of *path*(Q). Assume w.l.o.g. that *path*(Q) ends with $L(Q)$; then the entire *Seg-small*(S) is on *path*(Q) on which it preceded by *Seg-small*(Q), and this implies a crossing of Q with S. Now because *path*(Q, S) ends with $L(Q)$, this is a strong crossing, a contradiction. See also Fig. 4. ❑

Sketch of Proof of Lemma 5.6. Consider a pair of successive sets (A, B). From the assumptions of the lemma, either A is a right set or B is a left set. Therefore there are six types of situations arising: A is a right set and B is right, left or straight set, or B is left set and A is left, right or straight set. The proof goes by showing that unless A and B are either both left or right sets there is a contradiction with the assumptions. In this extended abstract, we will only consider the case where A is a right set and B is a left set, the other cases are similar. Since B is a left set its successor has to be also a left set (to prevent contradiction). By continuing this argument we get that the set C of which A is a successor is also a left set. But then the pair of successive sets (C, A) contradict the assumptions. ❑

Appendix D: Algorithm for MinAntLoad

We now briefly sketch the ideas behind a (quite involved) algorithm for MinAntLoad that achieves an approximation ratio of 1.5.

Assume first that the assumptions of Lemma 5.5 are not satisfied. From Lemma 5.6, we know all the normalized sets are entirely of the left or right type and therefore no crossing exists. In the latter case we can separately break the circular order of L to get $(j_1, \ldots, j_k)$ and the circular order of $U \setminus L$ to get $(i_1, \ldots, i_\ell)$ and then we can find a solution with greedy packing. Set S_1 will have *first-small*$(S_1) = i_1$ and set S_a will have $L(S_a) = j_a$, and given *first-small*(S_a), we find the longest possible *Seg-small*(S_a) such that $d(S_a) \leq 1.5$ and $S_a \subseteq R(j_a)$ or $S_a \subseteq R($*first-small*$(S_a))$. If $a \neq k$, we define *first-small*(S_{a+1}) as the first element of $U \setminus L$ that follows *Seg-small*(S_a). We can run this greedy algorithm for every possible pair i_1, j_1.

Dynamic Programming Formulation: We now assume that assumptions of Lemma 5.5 are satisfied. The idea behind dynamic programming is to find all possible paths in $(U, Succ)$ that can be obtained by successive applications of Lemma 5.5 (as was explained in Section 5.1). To check if a path can be obtained in this manner we have two cases. The first case is that this path is covered with a single set; obviously we can quickly verify if this is possible. The second case is that the path can be covered with a number of right sets, followed by a straight or a small set, followed by a number of left sets (any of the three groups may be missing but there have to be at least two groups). Let $(j_1, \ldots, j_k)$ be the subsequence of the path that consists of elements of L, and let $(i_1, \ldots, i_\ell)$ represent the subsequence that consists of elements of $U \setminus L$. We have two subcases for this case.

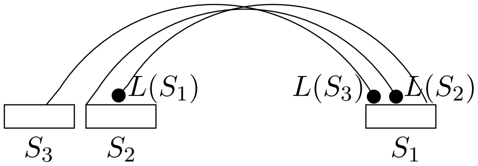


Figure 5: A possible arrangement of sets. Set S_1 is a right set, where as sets S_2 and S_3 are left sets. There are no strong crossings. The right sets form the initial part of the path, while the left sets form the final part.

The first subcase is that the set S_1 that is used in such a cover and such that $L(S_1) = j_1$ does not cross any other set. We form set $S_1 = \{j_1, i_1, i_2, \ldots, i_b\}$ such that $S_1 \subseteq R(j_1)$ or $S_1 \subseteq R(i_1)$ and $d(S_1) \leq 1.5$ and b is maximal, and then we ask the question about covering $(j_2, \ldots, j_k)$ and $(i_{b+1}, \ldots, i_\ell)$. The case when the set that contains j_k does not cross any other set is symmetric.

Now we consider the subcase in which the desired cover exists only if all right sets and left sets included in the cover participate in the set crossings. There are three more possible scenarios in this subcase: (a) a straight set exists, (b) it exists and it is a small set, and (c) it does not exists.

In this extended abstract, we will only describe the dynamic programming for the Scenario (a), the other two are similar and simpler. First in every possible way we guess the straight set S_c: we assume a certain $L(S_c) = j_c$ and *first-small*$(S_c) = i_d$. We find the maximal *Seg-small*(S_c) that is consistent with our assumptions, and we abandon this assumption if $j_c \notin$ *Seg-small*(S_c). Otherwise from the assumption about j_c, i_d we compute the last element of *Seg-small*(S_c). This computed last element is denoted by $f_1(j_c, i_d)$.

Now we consider a prefix of $(j_1, \ldots, j_k)$, say $(j_1, \ldots, j_a)$, and possible unions of *Seg-small*'s of sets $S_1, \ldots, S_a$ and S_c. One can see that in $(U \setminus L, Succ_s)$ this union forms two contiguous path, one that starts at i_1 and ends at some i_b (Part 1) the other starts at i_d and ends at $f_2(j_c, i_d, j_1, j_a, i_1, i_b)$ (Part 2). We need a recursive formula for f_2 and this will be the essence of our dynamic programming solution. We assume that *impossible* is one of the values of f_2.

The induction base is when $a = 1$. Here if $i_1 \neq Succ_s(i_b)$, then $f_2 =$ *impossible* (this means that Part 1 has to be empty). Otherwise, we find the longest sequence in $U \setminus L$ that we can associate with j_1 and that starts at $Succ_s(f_1(j_c, i_d))$. If S_1 is a right set, with *Segment*(S_1) starting at i_1, then we actually have the first subcase.

Now we compute $f_2(j_c, i_d, j_1, j_a, i_1, i_b)$ for $a > 1$. Now either (i) *Seg-small*(S_a) ends Part 1 or (ii) *Seg-small*(S_a) ends Part 2.

For (i), we find the minimal possible *first-small*(S_a) in ordered set $\{i_1, \ldots, i_b\}$ that satisfies the following conditions: *Seg-small*(S_a) ends at i_b, $L(S_a) = j_a$; $d(S_a) \leq 1.5$, $j_a \in R($*first-small*$(S_a))$. This computed *first-small*(S_a) is denoted by $f_3(j_a, i_b)$. Therefore we get the following recursive formula:

$$f_2(j_c, i_d, j_1, j_a, i_1, i_b) = f_2(j_c, i_d, j_1, j_{a-1}, i_1, f_3(j_a, i_b)).$$

For (ii), we start *Seg-small*(S_a) at $f_2(j_c, i_d, j_1, j_{a-1}, i_1, i_b)$ and this implies:

$$f_2(j_c, i_d, j_1, j_a, i_1, i_b) = f_1(j_a, f_2(j_c, i_d, j_1, j_{a-1}, i_1, i_b)).$$

Among (i) and (ii), we pick as solution the one covering more customers. Clearly, we can tabulate f_1 and f_3, so the recursive formula for f_2 can be computed in constant time, and a dynamic programming that computes f_2 fills a table with $O(n^6)$ entries.

We can use f_2 as follows: we can cover $(j_1, \ldots, j_k)$ and $(i_1, \ldots, i_\ell)$ with k sets if for some j_c, i_d we have $f_2(j_c, i_d, j_1, j_k, i_1, i_{d-1}) = i_\ell$.

One can see that in time $O(n^6)$ we can find all tuples $(j_1, j_k, i_1, i_\ell, m)$ such that $(j_1, \dots, j_k)$ together with $(i_1, \dots, i_\ell)$ can be covered with m sets from a family that forms a rounding of a compact solution.