Constructing Multiple Wireless Meshes in the Same Region with Beam-Crossing Grids

Pai-Hsiang Hsiao and H. T. Kung
Division of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
{*shawn, htk*}*@eecs.harvard.edu*

***Abstract*— Given a set of wireless access points (APs) with directional antennas, we consider the problem of steering their antennas to form multiple non-interferencing wireless meshes in the same region. The resulting multiple wireless meshes provide not only a simultaneously usable, but also a fail-safe, wireless network infrastructure. We describe a mesh construction algorithm, called "beam-crossing grids", that allows the constructed multiple wireless meshes to cover a large common area, while not interfere with each other. This means that, from locations in a large area, wireless terminals can have access to more than one of these meshes for improved network bandwidth and redundancy. By using analysis and simulation, we show that when the AP density is relatively high, the algorithm converges rapidly to a high-performance construction of wireless meshes.**

I. Introduction

A wireless mesh, as we mean in this paper, is a set of wireless access points (APs), where some pairs of them are directly connected with radio links. There has been an increasing interest in such wireless mesh networks due to their relatively ease of deployment. For example, Beyer [1] discusses the advantages of using such wireless mesh networks to provide residential broadband Internet access. Nortel Networks recently started trials of their wireless mesh networks with British Telecom and other organizations. They use APs equipped with smart antennas and integrated routers. [2]

Consider a set of APs placed in a 2-dimensional space. We assume that these APs use directional antennas for inter-AP wireless connections. Typically a wireless IP networking infrastructure can be formed using these APs in two steps. First, select pairs of APs to connect directly with radio links and then steer/configure their antennas to make the radio connections. Second, by using a routing protocol, compute a routing path (if it exists) between any two APs. These computed paths will then be used for transporting IP packets on the network.

A main concern of these wireless IP infrastructures is their reliability. Unlike wireline links, the performance of a wireless link can be affected by many environment-related factors, including path loss, shadowing, fading, interference from other users, and the non-linearity of underlying devices. A wireless infrastructure must tolerate not only permanent link failures, but also transient failures. Conventional routing protocols can recover from link failures, but their routing message overheads and end-to-end control delays often make them not suited to wireless infrastructures where bandwidths are more limited and transient link failures are frequent. To combat these problems, alternative routing methods have been proposed, such as those where multiple routes are maintained [3]–[5]. But these methods tend to be more complicated and difficult to manage.

To support a fail-safe wireless infrastructure capable of fast recovery from failure, in this paper we take the position that additional redundancy at the wireless link level ought to be provided. That is, the infrastructure should include multiple wireless meshes, so that any node or terminal can immediately switch over to another wireless mesh when detecting failure in the mesh that it currently uses.

We assume that we can configure and steer the directional antennas of the APs under software control (e.g., see [6] for such antennas). By steering the beams of these antennas, we will form multiple wireless meshes, which will cover as large a common area as possible, while not interfering with each other. We will use these multiple wireless meshes to provide redundancy at the link level. When there are link failures, the multiple wireless meshes can back up each other. When there is no failure, these multiple wireless meshes can operate in parallel to increase the total bandwidth of the wireless infrastructure.

We describe an algorithm, called *beam-crossing grids*, to construct multiple wireless meshes. The main insight of the algorithm, as its name suggests, is that it uses a grid of square regions, where two or triple crossing beams have been configured, as building blocks to form multiple wireless meshes. Given square regions with these crossing beams, it becomes relatively easy to connect them to form multiple meshes. In contrast, if square regions with parallel beams were used, then it tends to be the case that only one mesh can be constructed.

The paper is organized as follows. We first state the simplifying assumptions we make about the radios and antennas in Section II. We then describe the beam-crossing grids algorithm for constructing two wireless meshes in Section III. An upper bound on the running time of the algorithm is given in Section IV. We use simulations to evaluate the performance of this algorithm. The simulation results are reported in Section V. Section VI discusses the applications of our method, and its generalization in constructing more than two collocated

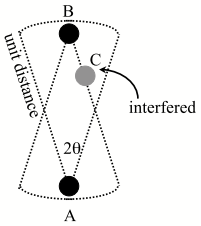


Fig. 1. Two ideally sectorized transmit beam patterns for inter-AP communication, where A and B are two communicating APs and C is an AP whose reception is interfered with by both A and B's transmission

networks in the region. We summarize this paper and provide conclusions in Section VII.

II. Simplifying Radio and Antenna Model

We make several simplifying assumptions on the radio and antenna models in this paper. We assume that inter-AP radio communication uses a dedicated frequency band which is different from those used by APs in communicating with wireless terminals. For inter-AP communication, the transmitting AP uses a directional antenna while the receiving AP uses an omnidirectional antenna. All APs produce the same transmit beam pattern, which is an ideally sectorized beam pattern, as illustrated in Fig. II. The length of the beam sector is of unit distance, and the angular width of the beam is 2θ. The gain inside the beam pattern is constant and is negligible both outside the pattern and on the boundary. To avoid interference, unintended AP receivers, e.g., C as shown in Fig II, cannot lie in the area of a transmit beam pattern of an AP. These assumptions are similar to those used by some typical models in the literature (see, e.g., [7]–[9]).

Given the assumptions about radio and antenna and a set of APs, we can specify a *connectivity graph*. In the connectivity graph, each vertex represents an AP, and two vertices are connected with an edge if the two corresponding APs can transmit to each other directly. In addition, we can also define an *edge conflict graph*. In the edge conflict graph, a vertex represents an edge of the connectivity graph. An edge connecting two vertices in the edge conflict graph indicates the two underlying radio links interfere with each other. For the case when two edges of the connectivity graph share a common vertex, we declare that there is no conflict between the two edges. This is because, in practice, the two transmitting APs in this case will content with the shared AP using a medium access control protocol to avoid collision (see, e.g., [7]). In the example illustrated in Fig. II, AP C lies in the union area of the two radiated patterns will be interfered with by the communication, and the edge AB conflicts with all edges CD where $D \neq A$ and $D \neq B$.

Although we make several simplifying assumptions on the radio and antenna models, the algorithm to be described below only considers the connectivity graph and the edge conflict graph. It is possible to derive these two graphs from more sophisticated radio and antenna models, or from site-measurement database.

For all the simulations performed in Section III and Section V, θ is set to be $\frac{\pi}{12}$.

III. Constructing Two Wireless Meshes in the Same Region

In this section we describe an algorithm, called beam-crossing grids, for constructing two wireless meshes, from a given a set of APs randomly placed in a 2-dimensional space. The algorithm can be extended to constructing three or more collocated wireless networks as discussed in Section VI.

The goal of the algorithm is to produce two wireless meshes, as illustrated by Fig. 7, which cover as large a common area as possible, using links which are sufficiently apart to prevent possible radio interference.

We divide the entire area into unit squares, and in each square region we will find link pairs that can be used simultaneously without causing interference. We prefer beam-crossing link pairs, as illustrated in Fig. 2. Beam-crossing links will reduce the chance one wireless mesh may block the connectivity of another wireless mesh.

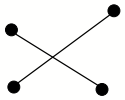


Fig. 2. Example of a beam-crossing link pair

The algorithm takes two inputs, a *connectivity graph* and an *edge conflict graph* (see Section II for the definitions of the two graphs). The output of the algorithm are two *subgraphs* of the connectivity graph, where any two edges in any subgraphs are interference-free according to the conflict graph. These two subgraphs represent the two wireless meshes that we are want to construct.

The algorithm assumes the location of each AP is known. The location information is used for two purposes: first, it is used to compute which square region an AP belongs to, and second, it is used to find proper link pairs in each square region, preferably beam-crossing ones. (The location information will not be necessary if the above two purposes can be achieved by other means.)

The algorithm consists of the following six steps:

1) Divide the area into unit squares, where the unit distance is the radio range of the APs use for inter-AP communication.
2) For each square we find a pair of links, if possible, that has no conflict. Both APs of a link must lie in the same square and the link pair can not share an AP. When there are multiple link pairs satisfying these conditions, the algorithm chooses the one with the largest weight, where weight is the sum of the length of the two links minus the sum of the minimal distances from APs to corners as shown in Fig. 3. Note that this selection criterion favors link pairs whose beams cross.
3) For each square for which a link pair has been found, recompute or eliminate link pairs if necessary so that

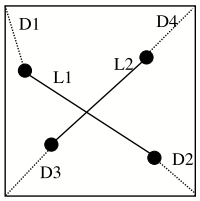


Fig. 3. An illustration of the weight computation in step 2. The weight of a link pair is $L1 + L2 - (D1 + D2 + D3 + D4)$. The use of each corner is exclusive, meaning that a corner can not be used twice. Note that this weight metric favors link pairs whose beams cross.

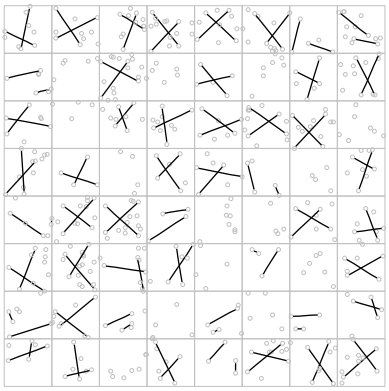


Fig. 4. The resulting link pairs after step 3. The link pairs are shown in black lines. Note that most of them are the beam-crossing type. It is possible that in some squares, no link pair can be found.

there will be no conflicts between links in adjacent squares. The algorithm achieves this by eliminating the link pair that has the most conflicts, and replaces it with a new link pair. Specifically, for each link pair, the algorithm computes its *penalty number* which is the number of other link pairs that it has conflicts with. The algorithm then eliminates the link pair with the largest penalty number and tries to find a replacing link pair in the same square. This time, the link pairs of adjacent squares are used as constraints. That is, the new link pair cannot have conflicts with the link pairs of the adjacent squares. No link pair will be chosen for the square if the no link pair satisfies the constraints. This process repeats until all link pairs' penalty numbers are zero. Fig. 4 is an example after this step.

4) For each square for which a link pair has been found, new links are added to connect them to link pairs in adjacent squares. (For computational simplicity, we include diagonally adjacent squares in our simulation). The two APs of an added link must be APs of previously chosen link pairs, and must belong to different squares. The algorithm looks at two adjacent squares at a time. Among the several ways of connecting two link pairs, we use the one that gives the largest number of links that are conflict-free. (This maximizes the number of in- and out-degrees of a square, thus helping find multiple networks.) The added links must also satisfy the constraint that there be no conflicts with nearby link pairs or other added links. This means that all the links, including the initial link pairs and the added links, are all conflict-free and can be used simultaneously. An example of the output of this step is shown in Fig. 5.

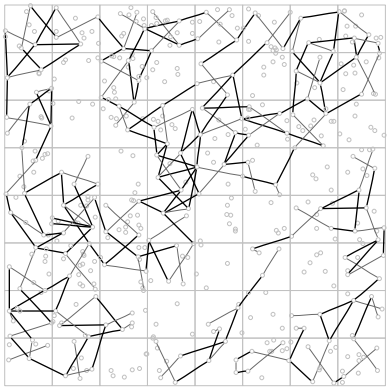


Fig. 5. The resulting link pairs after step 4. Link pairs from step 3 are shown in grey lines, and the added links of step 4 are shown in black lines.

5) Find connected components or subgraphs among links that are selected in step 3 and step 4. A beam-crossing link (any grey edge shown in Fig. 5) is randomly selected as the seed. The subgraph now contains the single seed edge. Starting from this seed edge, other connected edges are added to this subgraph through breadth-first search. A subgraph can only contain at most one beam-crossing link for each square. If the subgraph can not grow any more, the process starts with another randomly selected link and forms another subgraph. This process is repeated until every grey edge is included in a subgraph.
   This entire step is executed multiple times (we can bound the number of execution times for this step in the next section). Among these multiple executions, we will use the result of the execution that produces the largest combined size between the largest and second-largest subgraph.
   Fig. 6 shows an example of the resulting computed subgraphs. Only the largest four subgraphs are shown in this figure; the smaller subgraphs are not shown.
6) The algorithm now merges these subgraphs into two subgraphs. For each subgraph that is not the largest two subgraphs, we will perform a breadth-first search for a path that will connect the subgraph to one of the two subgraphs. The subgraph will select the connection path that will maximize the common area occupied by the resulting two subgraphs. An example of the two subgraphs resulting from step 6 is shown in Fig. 7.

IV. An Upper Bound on the Running Time of the Beam-Crossing Grids Algorithm

In this section we provide an upper bound on the running time of the beam-crossing grids algorithm presented in Section III. We will see that the upper bound can be expressed in terms of the size of the largest connected component, and

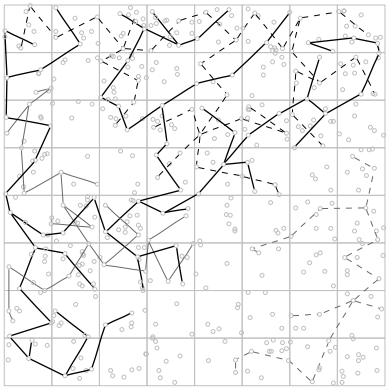


Fig. 6. The resulting subgraphs after step 5. There are four subgraphs shown in the figure. The largest one is depicted by the black lines, the second largest one by the dashed black lines, the third largest one by the grey lines, and the smallest one by the dashed grey lines.

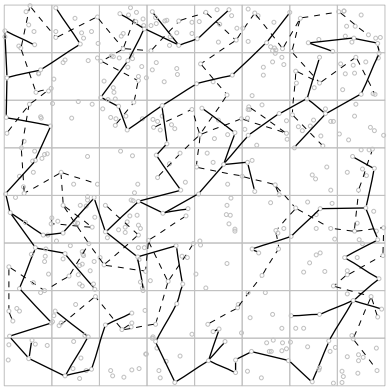


Fig. 7. The resulting two subgraphs after step 6. The first one is represented in black lines and has 99 nodes, while the second one is represented in dashed black lines and has 89 nodes.

then, in terms of the density of APs. We focus only on steps 5 and 6, since they are the dominating steps in running time.

Consider the graph resulting from steps 3 and 4. Let α be the ratio of the number of nodes in the largest connected component over that in the entire graph, and E the number of edges in the entire graph.

For any execution of step 5 of the algorithm, if the randomly selected seed happens to belong to the largest connected component, then through breadth-first search the execution will find this largest connected component. Thus the number of execution times for step 5 is bounded above by E minus the number of edges in the largest connected component.

Similarly, we can provide an upper bound on the number of times that step 6 of the algorithm will need to perform its breadth-first search operations. The number is bounded above by E minus the number of edges in the largest two connected components, which in turn, is number of edges in the largest connected component.

We can give tighter upper bounds for steps 5 and 6 in the following manner.

Theorem 1 1: Let T be the expected running time of step 5 or 6 of the algorithm of Section III. Then $T = O(\frac{\alpha^3-\alpha+1}{\alpha^2} \times E)$.

Proof: Let C be the expected running time of step 5 or 6 when it completes in 1 run and, for i = 1, 2, ..., let $A(i)$ be the expected running time of the algorithm when it completes in i runs and $B(i)$ be the expected running time of the algorithm when it completes in more than i runs. Then

$$\begin{aligned} C &= O(E) \\ A(i) &\le iC \\ T &= \frac{\alpha E}{E} A(1) + (1 - \frac{\alpha E}{E}) B(1) \\ &= \alpha A(1) + (1-\alpha) B(1) \end{aligned}$$

Note that

$$\begin{aligned} B(1) &= \frac{\alpha E}{E-1} A(2) + (1 - \frac{\alpha E}{E-1}) B(2) \\ &\le A(2) + (1 - \frac{\alpha E}{E}) B(2) \\ &\le 2C + (1-\alpha) B(2) \end{aligned}$$

Similarly, for i = 2, 3, ..., we have

$$\begin{aligned} B(i) &= \frac{\alpha E}{E-i} A(i+1) + (1 - \frac{\alpha E}{E-i}) B(i+1) \\ &\le A(i+1) + (1 - \frac{\alpha E}{E}) B(i+1) \\ &\le (i+1)C + (1-\alpha) B(i+1) \end{aligned}$$

Thus

$$\begin{aligned} T &\le \alpha C + (1-\alpha) \frac{\alpha E}{E-1} A(2) + (1 - \frac{\alpha E}{E-1}) B(2) \\ &\le \alpha C + (1-\alpha) 2C + (1-\alpha)^2 3C + (1-\alpha)^3 4C + \cdots \end{aligned}$$

By computing a closed-form expression for the right-hand side of the above inequality, we have

$$T \le \frac{\alpha^3 - \alpha + 1}{\alpha^2} C$$

■

Fig. 8 and Fig. 9 are simulation results which relate the size of the largest connected component to the average density of APs in a square. The former shows the largest component size as a function of probability that adjacent squares can be connected, and the latter shows the probability that adjacent squares can be connected as a function of the average density of APs in a square. Combing these results with the earlier result in this section, we see that the expected running time of the beam-crossing grids algorithm of Section III is a function of the average density of APs in a square. In particular, we see that the former will decrease as the latter increases.

V. Evaluation and Simulation Results

We evaluate the performance of the proposed algorithm using simulations, and report the simulation results in this section. We first introduce a coverage ratio that measures how well the area is covered by the meshes. Finally, we describe the simulation setup and report the results in terms of coverage ratio and number of APs used.

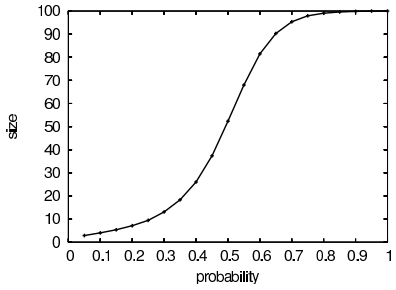


Fig. 8. Largest component size as a function of probability that adjacent squares can be connected

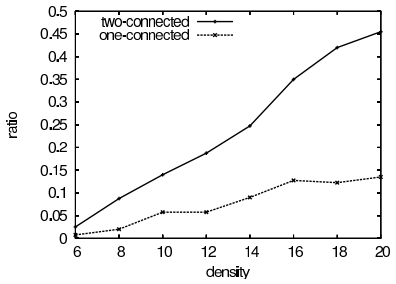


Fig. 9. Probability that two adjacent squares are connected as a function of density. Two-connected probability represents the probability that the two links can be found to connect the pairs of beam-crossing links; one-connected probability represents the probability that only one link can be found to connect the pairs of beam-cross links.

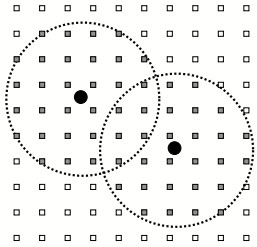


Fig. 10. An example of coverage ratio computation. There are a total of 100 uniformly placed sampling locations in the network and there are two APs, each covering a part of the network. The AP on the left covers 32 sampling locations, while the AP on the right covers 29 sampling locations. They jointly cover 55 sampling locations and their coverage intersects at 6 sampling locations. The union coverage ratio of the two APs combined is then 0.55 and the intersection coverage ratio is 0.06.

density	no. of APs	size of network 1	size of network 2
4	256	31.39	39.78
6	384	69.96	70.84
8	512	95.36	91.96
10	640	108.64	104.76
12	768	117.51	111.83
14	896	122.61	116.71
16	1024	124.72	117.83

TABLE I
SIZE OF EACH NETWORK

We assume each AP covers a circular area centered at the AP and the radius is its *access radio range*. The area a network covers is the coverage area of all its APs. The coverage ratio measures approximately the fraction of area covered by a network, by sampling at uniformly placed sampling locations. More specifically, all sampling locations lying in the circular area of any AP belonging to a particular network are said to be covered by the network. The coverage ratio is the number of covered sampling locations to the total number of sampling locations. An example of the coverage ratio computation is shown in Fig. 10. The union coverage ratio and the intersection coverage ratio of two networks can also be computed. For example, if each AP in Fig. 10 is a network, then the union coverage ratio is 0.55 and the intersection coverage ratio is 0.06.

For all simulations, the APs are randomly placed in an area of size 8 by 8. A *density* is used to control the total number of APs to be deployed in the networking area. We consider the following densities: 4, 6, 8, 10, 12, 14, 16. This translates into a total of 256, 384, 512, 640, 768, 896 and 1024 APs, respectively. For computing coverage ratio, we place sampling locations 0.05 apart, so there are a total of 25600 sampling locations in the networking area. Because this access radio range affects coverage ratio, we consider the following access radio ranges: 1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4. For step 5 of the algorithm, 10 random trials are performed and the best results are chosen.

Fig. 11(a) to Fig.11(g) show the coverage ratios for various configurations. Each figure has a fixed access radio range, and shows the coverage ratios of all APs, the union and intersection coverage ratio of two networks, and the coverage ratios of individual networks. The coverage ratios reported are the averages of 100 simulations, each with randomly generated AP locations. Table I shows the average size, in number of nodes, of each network.

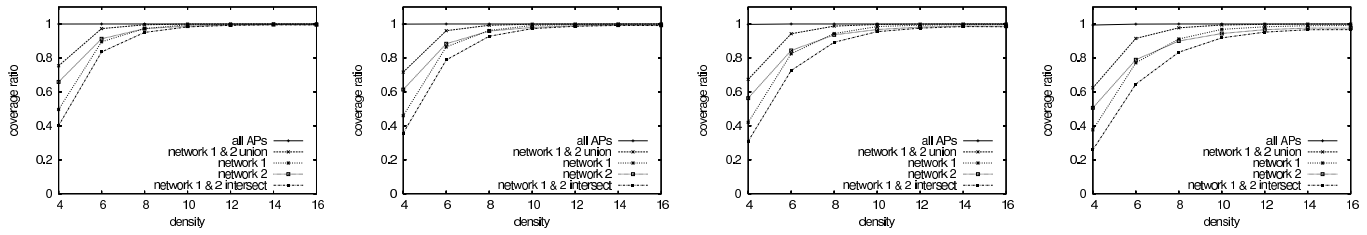
The results of coverage ratios show that, when the density is 8 or higher than and the access radio range is 0.6 or larger, each network on average covers more than 80% of the area. The union coverage of both networks in this case covers more than 95% of the area. This demonstrates that the proposed algorithm is effective in finding two networks without losing coverage.

Because the algorithm connects single links found in small regions, the resulting individual network uses no more than two APs per square. On the other hand, to achieve good performance the density must be greater than or equal to 8. This means that in order to achieve good performance, twice as many APs must be supplied to the algorithm than those would be used.

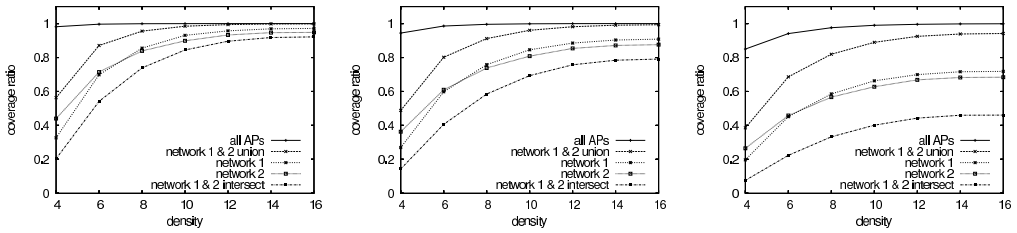
VI. DISCUSSION

The described method and algorithm can be used to organize APs in order to increase the throughput of broadband Internet services [1]. For example, in a rural area where wireline broadband has not yet be deployed, household APs can be connected using the method to increase the throughput of the network.

The method and algorithm can also be used for planning the deployment of APs. Instead of choosing from the set



(a) Access Radio Range = 1.0 (b) Access Radio Range = 0.9 (c) Access Radio Range = 0.8 (d) Access Radio Range = 0.7



(e) Access Radio Range = 0.6 (f) Access Radio Range = 0.5 (g) Access Radio Range = 0.4

Fig. 11. Density vs. coverage ratio performance with different access radio ranges. The coverage ratios reported are average coverage ratios of 100 randomly generated networks.

of already-deployed APs, we consider all potential AP deployment locations. The output of the algorithm is a list of locations where APs should be placed and how they form networks for higher throughput. In other words, the method and algorithm can be used to select AP deployment locations. If APs can be placed at arbitrary locations, then there are other strategies that can be used to deploy these APs. For example, in our prior work [10], we consider the problem of collocating identical square mesh wireless networks.

The algorithm generates two wireless networks that each has the topology of a tree. Tree topology has the property that it uses minimal number of edges to connect all vertices. Because each additional link used in the network imposes additional interference constraint to itself and the other network, it is nature to design the algorithm to construct the network with tree topology. We could add shortcut links that connect branches of the network to reduce path length between any pair of APs, but we don't consider this optimization technique in this paper.

The major disadvantage of tree topology is its lack of redundancy— any link breaks can result in the partitioning of the network. It may not be possible to reconnect the network by using other APs around the faulty area, so new networks should be recomputed using non-faulty links. When links break due to faulty APs, the algorithm excludes the faulty APs as well as all links associated with them, and reconstructs the networks. Fig. 12 demonstrates, by simulations, that the algorithm can still find large-coverage collocated networks when some APs used by the networks are removed. For those simulations, the networking area is a 8 by 8 area and initially there are a total of 640 APs randomly placed in the network

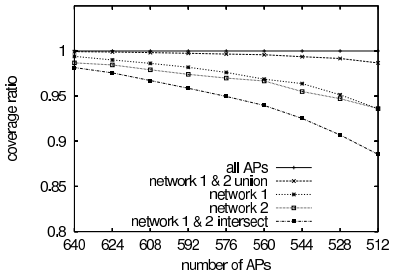


Fig. 12. Changes in coverage ratios when more APs used by the computed networks are removed. Initially, there are a total of 640 APs in the area (density of 10). At each iteration, 8 randomly selected APs are removed from each network and two new networks are computed from the remaining APs. The coverage ratios reported are the average of 100 randomly generated networks.

(density of 10). At each iteration, 8 randomly selected APs are removed from each computed network and the algorithm computes two new networks using the remaining APs. The process repeats until a total of 128 APs (64 from each network) are removed from the network. The coverage ratios reported are average coverage ratios from 100 randomly generated networks and are computed with access radio range set to be 1.0. The density becomes 8 after all of the 128 APs are removed, and the coverage ratios are in general lower than the coverage ratios reported in Fig. 11(a) with the same density and access radio range. This is because only the APs that were used in the networks are removed and these APs are better positioned to form networks than the others. When the density of remaining APs are high, the algorithm can still perform well even though a large number of better positioned APs are removed.

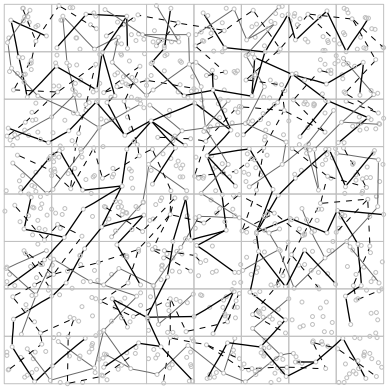


Fig. 13. An example of three collocated networks. The largest one is shown in black solid lines, the second largest one is shown in black dashed lines and the smallest one is shown in grey solid lines.

Although the algorithm described finds two collocated networks, it can be generalized to any number. For example, to find three networks, instead of finding a link pair for each square, the algorithm should seek link triplets. The largest three subgraphs are then used to merge with other smaller subgraphs. Fig. 13 shows an example of 3 networks constructed by this extended algorithm. The θ is reduced to $\frac{\pi}{18}$ when performing the simulation.

We do not attempt to minimize the number of APs used. Minimizing the number of APs used while covering at least a certain portion of the area requires the knowledge of AP coverage patterns. Our algorithm does not assume specific coverage patterns, and it only assumes (1) the union coverage does not decrease as more APs are used, and (2) the closer the two APs are the smaller their union coverage is. On the other hand, due to its design, each network never uses more than two APs in a square. This provides an upper bound on the total number of APs for each network.

VII. Conclusion

In this paper, we describe an algorithm, called *beam-crossing grids*, for constructing two collocated wireless mesh networks from a collection of APs in a region. The two wireless meshes can operate independently of each other and wireless terminals can use the networks for fault-tolerance or load-balancing access.

The algorithm is evaluated using simulations. The coverage area of the individual wireless networks increases as the density of APs increases. When the density is 8 or higher and access radio range is 0.6 or higher, our simulation results show that the algorithm effectively constructs two collocated networks that each covers more than 80% of the area. The coverage ratio increases as density or access radio range increases. Because the algorithm divides the area into small regions and for each network it uses at most two APs in a region, the total number of APs used by each network is bounded above by two times the number of regions.

We also discuss how to extend the algorithm to construct more than two networks, and show an example of constructing three networks.

We plan to extend this work in several directions. First, we will take the AP coverage patterns into consideration to allow each computed wireless network to use fewer APs without sacrificing their coverage areas. Second, we will extend the method to 3-dimensional space when the coverage patterns are well-defined. Finally, we plan to investigate the application of this method for cases where APs can adjust their transmitting power levels. This can minimize interference caused by a transmission and allow constructing an expanded number of collocated networks.

Acknowledgments

This research was supported in part by NSF grant #ACI-0330244, and in part by a research grant from Microsoft Research.

References

[1] D. Beyer, "Wireless mesh networks for residential broadband," in *National Wireless Engineering Conference*, San Diego, 2002.

[2] Nortel Networks, "Nortel Networks announces plans to trial breakthrough wireless LAN architecture with MIT and BT," 2003, press release.

[3] V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *Proc. of IEEE INFOCOM*, 1997, pp. 1405–1413.

[4] A. Nasipuri, R. Castaneda, and S. R. Das, "Performance of multipath routing for on-demand protocols in ad hoc networks," *ACM/Kluwer Mobile Networks and Applications (MONET) Journal*, vol. 6, no. 4, pp. 339–349, 2001.

[5] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wirelese sensor networks," *Mobile Computing and Communications Review*, vol. 5, no. 4, pp. 11–25, 2001.

[6] J. Kajiya, "Commodity software steerable antennas for mesh networks," in *Mesh Networking Summit*, 2004.

[7] Y.-B. Ko, V. Shankarkumar, and N. H. Vaidya, "Medium access control protocols using directional antennas in ad hoc networks," in *Proc. of IEEE INFOCOM*, March 2000.

[8] M. Takai, J. Martin, and R. Bagrodia, "Directional virtual carrier sensing for directional antennas in mobile ad hoc networks," in *Proc. ACM Intern. Sym. on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, June 2002.

[9] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "Energy limited wireless networking with directional antennas: the case of session-based multicasting," in *Proc. of IEEE INFOCOM*, 2002.

[10] P.-H. Hsiao and H. T. Kung, "Layout design for multiple collocated wireless mesh networks," in *Proc. of IEEE VTC 2004 Fall*, September 2004.