Optimizing the Placement of Integration Points in Multi-hop Wireless Networks

Lili Qiu, Ranveer Chandra, Kamal Jain, and Mohammad Mahdian

ABSTRACT

Efficient integration of a multi-hop wireless network with the Internet is an important research problem, and benefits several applications, such as wireless neighborhood networks and sensor networks. In a wireless neighborhood network, a few Internet Transit Access Points (ITAPs), serving as gateways to the Internet, are deployed across the neighborhood; houses are equipped with low-cost antennas, and form a multi-hop wireless network among themselves to cooperatively route traffic to the Internet through the ITAPs. In a sensor network, sensors collect measurement data and send it through a multi-hop wireless network to the servers on the Internet via ITAPs. For both applications, placement of integration points between the wireless and wired network is a critical determinant of system performance and resource usage. However there has been little work on this subject.

In this paper, we explore the placement problem under three wireless link models. For each link model, we develop algorithms to make informed placement decisions based on neighborhood layouts, user demands, and wireless link characteristics. We also extend our algorithms to provide fault tolerance and handle significant workload variation. We evaluate our placement algorithms using both analysis and simulation, and show that our algorithms yield close to optimal solutions over a wide range of scenarios we have considered.

1. INTRODUCTION

Unprecedented growth in wireless technology has made a tremendous impact on how we communicate. Ubiquitous wireless Internet access has become a reality in many public places, such as airports, malls, coffee shops, and hotels. More recently, people have applied wireless technology to obtain broadband access at home and a number of neighborhood networks have already been launched across the world [1, 8]. Using wireless as the first mile towards the Internet has a big advantage – fast and easy deployment [19]. Therefore it is especially appealing to homes that are out of reach of cable & DSL coverage, such as rural and suburban areas. Even for areas with cable or DSL coverage, providing an alternative for Internet access is extremely useful, as it helps to increase network bandwidth, and suits the diverse needs of different applications. A similar problem of efficiently bridging a multi-hop wireless network with the Internet also arises in sensor networks, where sensors collect data and send it through a multi-hop wireless network to servers on the Internet via Internet Transit Access Points (ITAPs).

Both these applications require efficient bandwidth utilization at end nodes, which can be achieved through a careful placement of ITAPs. Motivated by these applications, this paper explores efficient integration of multi-hop wireless networks with the Internet by placing ITAPs at strategic locations. We note that an alternative to multi-hop wireless networks is the cellular approach, which sets up an ITAP to service users within its communication range. However this approach requires significantly more ITAPs than the multi-hop scheme [3], and is also difficult to implement [8]. Therefore, we focus on the multi-hop approach in this paper.

Neighborhood networks are characterized by two important design constraints. They should be easy and cheap to deploy. Moreover, they should provide Quality of Service (QoS) guarantees to end users. To achieve both these constraints it is imperative to have an intelligent placement of ITAPs in the network. Any ITAP placement algorithm will have to (i) efficiently use wireless capacity, (ii) take into account the impact of wireless interference on network throughput, and (iii) be robust in face of failures and changes in user demands. There has been little previous work on this subject.

In this paper, we investigate schemes to efficiently place ITAPs in a multihop wireless network. Our key contributions are:

- We formulate the ITAP placement problem under three wireless models. For each model, we develop algorithms to efficiently place ITAPs in the network. Our algorithms aim to minimize the number of required ITAPs while guaranteeing users' bandwidth requirements. We demonstrate the efficiency of the algorithms through simulation and analysis.
- To enhance robustness, we present a fault tolerance version of the placement algorithm that provides bandwidth guarantees in the presence of failures.
- We extend the algorithms to take into account variable traffic demands by developing an approximation algorithm to simultaneously optimize ITAP placement based on demands over multiple periods. This algorithm is very useful in practice since user demands often exhibit periodic changes (e.g., diurnal patterns).

In the rest of this paper we first overview related work in Section 2. In Section 3 we describe the ITAP placement problem and our network models. In Section 4, we propose novel placement algorithms for three wireless link models. We further validate these link models using packet-level simulations in Section 5. In Section 6, we evaluate the performance of placement algorithms using various topologies and traffic patterns. In Section 7, we extend our placement algorithms to take into account two important practical factors: fault tolerance and changing workload. We evaluate their performance through analysis and simulation. We conclude in Section 8.

2. RELATED WORK

There has been a recent surge of interest in building wireless neighborhood networks. Some commercial networks that provide Internet access to home users using this technology are described in [1] and [8]. [1] presents a scheme to build neighborhood networks using standard 802.11b Wi-Fi technology [23] by carefully positioning access points in the community. Such a scheme requires a large number of access points, and direct communication between machines and the access points. This constraint is difficult to meet in real terrains. The other approach to building neighborhood networks is Nokia's Rooftop technology, presented in [8]. This scheme provides broadband access to households using a multi-hop solution that overcomes the shortcomings of [1]. The idea is to use a mesh network model with each house deploying a radio, as considered in this paper. This radio serves the dual purpose of connecting to the Internet and routing packets for neighboring houses [4]. The deployment and management cost of Internet TAPs in such networks is significant, and therefore it is crucial to minimize the required number of ITAPs to provide QoS and fault tolerance guarantees. However, these problems are not addressed in [1, 8].

There have been a number of interesting studies on placing servers at strategic locations for better performance and efficient resource utilization in the Internet. For example, the authors in [22, 18, 25] examine placement of Web proxies or server replicas to optimize clients' performance; and Jamin *et al.* [17] examines the placement problem for Internet instrumentation. The previous work on server placement cannot be applied to our context because they optimize locality in absence of link capacity constraints. This may be fine for the Internet, but is not sufficient for wireless networks since wireless links are often the bottlenecks. Moreover, the impact of wireless interference, and considerations of fault tolerance and workload variation make the ITAP placement problem very different from those studied earlier.

It is worth mentioning that the ITAP placement problem can be considered as a facility location type of problem. Facility location problems have been considered extensively in the fields of operations research and approximation algorithms (e.g., [21, 29]). Approximation algorithms with good worst case behavior have been proposed for different variants of this problem. However, to the best of our knowledge, these results do *not* concern the case where *links* have capacities, as considered in this paper. The presence of link capacity constraints makes our problem more challenging from a theoretical perspective. In addition, the effects of wireless interference and variable traffic demands have not been considered in the previous facility location work.

The work closest to ours is the pioneering work in [3]. It aims to minimize the number of ITAPs for multi-hop neighborhood networks based on the assumption that ITAPs use a Time Division Multiple Access (TDMA) scheme to provide Internet access to users. However, TDMA is difficult to implement in multi-hop networks due to synchronization and channel constraints [2]. Furthermore, the proposed slotted approach might not utilize all the available bandwidth due to unused slots. In comparison, in this paper we look at more general and efficient MAC schemes, such as IEEE 802.11. Removing the TDMA MAC assumption yields completely different designs, and increases applicability of the resulting algorithms.

In summary, placing ITAPs under the impacts of *link capacity constraints, wireless interference, fault tolerance, and variable traffic demands* is a unique challenge that we aim to address in this paper.

3. PROBLEM DESCRIPTION AND NETWORK MODEL

We describe the ITAP-placement problem and our approach in the context of wireless neighborhood networks, but the same problem formulation and approaches apply for sensor networks. The ITAP-placement problem, in its simplest form, is to place a minimum number of ITAPs that can serve a given set of nodes on a plane, which we call houses. A house h is said to be successfully served, if its demand, w_h, is satisfied by the ITAP placement. A house h is served by an ITAP i through a path between h and i. This path is allowed to pass through other houses, but any two consecutive points on this path must have wireless connectivity between them. We are usually interested in the fractional version of this problem. That is, we consider the flexibility that a house is allowed to route its traffic over multiple paths to reach an ITAP.

This problem can be modeled using the following graph-theoretic approach. Let $\mathcal{H}$ denote the set of houses and $\mathcal{I}$ denote the set of possible ITAP positions. We construct a graph G on the set of vertices $\mathcal{H} \cup \mathcal{I}$ by connecting two nodes if and only if there is wireless connectivity between them. The goal is to open the smallest number of ITAPs (denoted by the set $\mathcal{I}'$), such that in the graph $G[\mathcal{H} \cup \mathcal{I}']$, one can route w_h units of traffic from house h to points in $\mathcal{I}'$ simultaneously, without violating capacity constraints on vertices and edges of the graph, where w_h is the demand from house h.

The edge capacity, Cap_e, in the graph denotes the capacity of a wireless link. In addition, each node also has an upper bound on how fast traffic can go through it. Therefore, we also assign each node with a capacity, Cap_h. Usually $\mathrm{Cap}_h = \mathrm{Cap}_e$, as both represent the capacity of a wireless link. (Our schemes work even when $\mathrm{Cap}_h \neq \mathrm{Cap}_e$, e.g., when a node's processing speed becomes the bottleneck.) Moreover, each ITAP also has a capacity limit, based on its connection to the Internet and its processing speed. We call this capacity, the ITAP capacity, Cap_i.

In addition to edge and vertex capacities and house demands, another input to the placement algorithms is a wireless connectivity graph (among houses). We can determine whether two houses have wireless connectivity using real measurements, and give the connectivity graph to our placement algorithms for deciding ITAP locations. In our performance evaluation, since we do not have wireless connectivity graphs based on real measurements, we instead derive connectivity graphs based on the *protocol model*, introduced in [13]. In this model, two nodes i and j can communicate directly with each other if and only if their Euclidean distance is within a communication radius, CR. Given the position of all the nodes, we can easily construct a connectivity graph by connecting two nodes with an edge if their distance is within CR. However our placement algorithms can also work with other wireless connectivity models (e.g., physical model [13] or based on real measurements). In the following sections, we study several variants of this placement problem.

3.1 Incorporating Wireless Interference

There are several ways to model wireless interference. One approach is to use a fine-grained interference model based on the notion of a conflict graph, introduced in [16]. The conflict graph indicates which groups of links mutually interfere and hence cannot be active simultaneously. The conflict graph model is flexible to capture a wide variety of wireless technologies, such as directional antennas, multiple radios per node, multiple wireless channels, and different MAC protocols. The main challenge of using the fine-grained interference model is high complexity (sometimes prohibitive), since for even a moderate-sized network the number of interference constraints arising from the conflict graph can become hundreds of thousands.

An alternative approach is to use a coarse-grained interference model that captures the trend of throughput degradation due to wireless interference. Since there are usually a limited number of wireless channels available, not all links can be active at the same time to avoid interference. As a result, wireless throughput generally degrades with the number of hops in the path as we show in the following scenario. Consider a linear-chain network, where each link has a unit capacity. Since the interference range of a node is typically larger than the communication range [28], it is possible that all the nodes in the chain interfere with each other. In this case, only one link can be active at a time, which suggests that the maximum throughput from node 0 to node n is $\frac{k}{n}$ for $k < n$ and 1 for $k \geq n$, where k is the number of available channels, and n is the number of hops. As we can see, if we have enough channels, the throughput can approach the channel capacity. On the other hand, if we only have one channel, then throughput degrades as a function of $\frac{1}{n}$. This has also been confirmed by several simulation studies based on 802.11 and other MAC protocols similar to 802.11 [15, 12].

In practice, the network topology can be more complicated, and the relationship between throughput degradation and an increasing hop-count depends on many factors, such as communication range versus interference range, the types of antenna (directional versus omni-directional), MAC protocols, and the number of contending radios. There is no single available function that can capture the impact of interference on wireless throughput. Therefore, we study the placement problem under several link models. We describe the link models using two related functions. In our discussion, $Throughput_l$ denotes the amount of throughput on a link along a path of length l, assuming each wireless link capacity is 1. The other function, $g(l)$, denotes the amount of link capacity consumed if it is on a path of length l and the end-to-end throughput of the path is 1. It is clear that $g(l) = \frac{1}{throughput_l}$, since in order to get one unit throughput along a path of length l, we need to have $\frac{1}{throughput_l}$ capacity at each edge along the path, assuming the end-to-end throughput increases proportionally with the edge capacity, which is true in practice.

In this paper, we study the following models separately:

1. Ideal link model: If $throughput_l = 1$ for all l, or equivalently, $g(l) = 1$, we get the basic version of the problem. This model is appropriate for the environment with very efficient use of spectrum. A number of technologies, such as directional antennas (e.g., [7, 11]), power control, multiple radios, and multiple channels, all strive to achieve close to this model by minimizing throughput degradation due to wireless interference.

2. General link model: A more general model is when $throughput_l$ or $g(l)$ is a linear function of l. As we will show in Section 4.2, we can formulate the ITAP placement problem for the general link model as an integer linear program, and develop polynomial placement algorithms. In addition, we also develop more efficient heuristics for two forms of $g(l)$.

   (a) Bounded hop-count model: If $throughput_l = 1$ for $l \leq k$ and $throughput_l = 0$ for $l > k$ (or equivalently, $g(l) = 1$ for $l \leq k$ and $g(l) = \infty$ for $l > k$), we get a variant in which flow cannot be routed through paths of length more than k. This approximates the case where we try to ensure each flow gets at least a threshold amount of throughput by avoiding paths that exceeds a hop-count threshold.

   (b) Smooth throughput degradation model: This corresponds to the case when $throughput_l = \frac{1}{l}$, where l is the number of hops in the path. This is equivalent to $g(l) = l$ for all l's (i.e., the capacity consumed is equal to the flow times the number of hops). This represents a conservative estimate on throughput in a linear-chain network as we show above, and therefore this model is appropriate when tight bandwidth guarantees are desired.

Note that the above models capture wireless interference and contention among nodes whose paths to ITAPs share common links or nodes. A more accurate model will have to handle interference among nodes on independent paths (e.g., using the conflict graph [16]). However, in Section 5, we use packet-level wireless network simulations to show that an ITAP placement based on the above models gives satisfactory performance.

3.2 Incorporating Fault Tolerance Consideration

A multi-hop scheme for building neighborhood networks requires different houses along a path to the ITAP to forward traffic to and from a house. The bandwidth requirements of a house may not be satisfied if even one house decides to shut itself down. Furthermore, ITAPs may be temporarily down. Our placement scheme handles such scenarios by routing traffic through multiple independent paths, and over-provisioning the delivery paths.

3.3 Incorporating Workload Variation

Several studies show that user traffic demands exhibit diurnal patterns (e.g., [6, 20, 27]). Since it is not easy to change ITAP locations once they are deployed, these ITAPs should handle demands over all periods. In Section 7.2, we present algorithms to simultaneously optimize ITAP locations based on workload during different periods.

3.4 Generic Approach

In the following sections, we will investigate different variants of the placement problem. Our generic approach is as follows. Given a set of potential ITAP locations, which may include all or a subset of points in the neighborhood, we first prune the search space by grouping points into equivalence class, where each equivalence class is represented by the set of houses that are reachable via a wireless link. For example, if points A and B have wireless connectivity to the same set of houses, then they are equivalent as far as ITAP placement is concerned. Therefore we only need to search through all the equivalence classes, instead of all points on the plane. (Refer to [24] for details). Then based on our choice of wireless link model, fault-tolerance requirements, and variability in user demands, we apply one of the placement algorithms described in Section 4, Section 7.1, and Section 7.2 to determine ITAP locations.

4. PLACEMENT ALGORITHMS

In this section, we study how to place ITAPs under the impacts of link capacity constraints and wireless interference.

4.1 Ideal Link Model

First, we consider the placement problem for the ideal link model. We formulate the problem as a linear program, and present an approximation algorithm.

4.1.1 Problem Formulation

We formulate the placement problem for the ideal link model as an integer linear program shown in Figure 1. For each edge e and house h, we have a variable $x_{e,h}$ to indicate the amount of flow from h to ITAPs that is routed through e. For each ITAP i we have a variable y_i that indicates the number of ITAPs opened at the location i (More precisely, y_i is the number of ITAPs opened at locations in the equivalence class i, where the equivalence class is introduced in Section 3.) Cap_e, Cap_h, and Cap_i denote the capacity of the edge e, house h, and ITAP i, respectively; w_h denotes the traffic demand generated from house h.

Now we present a brief explanation of the above integer linear program. The first constraint ($\sum_{e=(v,h')} x_{e,h} = \sum_{e=(h',v)} x_{e,h}$) formulates the flow conservation constraint, i.e., for every house except the house originating the flow, the total amount of flow entering the house is equal to the total amount of flow exiting it. The inequality $\sum_{e=(h,v)} x_{e,h} \geq w_h$ formulates the constraint that each house has w_h amount of flow to send, and the third constraint indicates that a house does not receive flow sent by itself. The next three inequalities of the above program capture the capacity constraints on the edges, houses, and ITAPs. The inequality $\sum_{e=(v,i)} x_{e,h} \leq w_h y_i$ says that no house is allowed to send any traffic to an ITAP unless the ITAP is open. Notice that this inequality is redundant and follows from the ITAP capacity constraint and the assumption that y_i is an integer. However, if we want to relax the integrality assumption on y_i's in order to derive a lower bound using an LP solver (see Section 4.3.4 for example), then it is important to include this inequality in the linear program so that we can get a tighter lower bound.

The following theorem shows that it is computationally hard to optimally solve the ITAP placement problem for the ideal link model. Refer to [24] for the proof.

$$
\begin{array}{lll}
\text{minimize} & \sum_{i\in\mathcal{I}} y_i & \\
\text{subject to} & \sum_{e=(v,h')} x_{e,h} = \sum_{e=(h',v)} x_{e,h} & \forall h,h' \in \mathcal{H}, h' \neq h \\
& \sum_{e=(h,v)} x_{e,h} \geq w_h & \forall h \in \mathcal{H} \\
& \sum_{e=(v,h)} x_{e,h} = 0 & \forall h \in \mathcal{H} \\
& \sum_{h} x_{e,h} \leq \text{Cap}_e & \forall e \in E(G) \\
& \sum_{h',e=(v,h)} x_{e,h'} \leq \text{Cap}_h & \forall h \in \mathcal{H} \\
& \sum_{h',e=(v,i)} x_{e,h'} \leq \text{Cap}_i y_i & \forall i \in \mathcal{I} \\
& \sum_{e=(v,i)} x_{e,h} \leq w_h y_i & \forall i \in \mathcal{I}, h \in \mathcal{H} \\
& x_{e,h} \geq 0 & \forall e \in E(G), h \in \mathcal{H} \\
& y_i \in \{0,1,2,...\} & \forall i \in I
\end{array}
$$

Figure 1: LP formulation for the ideal link model

THEOREM 1. *It is NP-hard to find a minimum number of ITAPs required to cover a neighborhood in an ideal link model. Moreover, the problem has no polynomial approximation algorithm with an approximation ratio better than* $\ln n$ *unless* $P = NP$.

4.1.2 Our Approach – Greedy Placement

We design the following greedy placement. We iteratively pick an ITAP that maximizes the total demands satisfied when opened in conjunction with the ITAPs chosen in the previous iterations. The major challenge is to determine how to make a greedy move in each iteration. This involves efficiently computing the total user demands that can be served by a given set of ITAPs. We make an important observation: computing the total satisfied demands can be formulated as a network flow problem. This is easy to see since our formulation for the ideal link model shown in Figure 1 satisfies the three properties, namely, capacity constraint, skew symmetry, and flow conservation. This suggests that we can apply the network flow algorithms [9] to efficiently determine the satisfied demands. A few transformations are required to make the network flow algorithm applicable. Figure 2 shows a skeleton of the algorithm, which finds a multiset S of ITAPs to open, where a multiset is the same as a set, except that it allows duplicate elements. Allowing duplicate elements in S indicates that we can open multiple ITAPs in the locations that belong to the same equivalence class (i.e., reachable from the same set of houses), which is certainly feasible.

The following theorem shows a worst-case bound on the performance of the above algorithm. An empirical performance analysis of this algorithm is presented in Section 6.1.

THEOREM 2. *Consider the ITAP placement problem in the ideal link model with integral demands and integral house and link capacities, and let* D *denote the total demand of the houses. The approximation factor of the greedy algorithm for this problem is at most* $\ln(D)$. *In other words, if the optimal solution for the ITAP placement problem opens* K *ITAPs, the greedy algorithm opens at most* $K\ln(D)$ *ITAPs.*

We will need the following lemma to prove the above theorem. This lemma is non-trivial and uses the Ford-Fulkerson maximum flow-minimum cut theorem [9] in the proof.

LEMMA 3. *Assume a multiset* S *of ITAPs are opened. Consider an optimal way of routing the maximum total demands from houses to the ITAPs in* S, *and let* f_i *denote the amount of traffic routed to ITAP* i *in this solution, where* $i \in S$. *Assume that at a later time, a multiset* $S \cup T$ *of ITAPs are opened. Then, there is an optimal way of routing the maximum total demands from houses to these ITAPs in which* f_i *units of traffic is routed to ITAP* i *for every* $i \in S$.

Input: Set of houses $\mathcal{H}$, set of ITAPs $\mathcal{I}$, graph G on the set $\mathcal{H} \cup \mathcal{I}$ with capacities on its edges and vertices.
Output: A multiset S of ITAPs to be opened.
begin

$S := \emptyset$; $Flow := 0$;

while $Flow$ is less than the total demand **do**

$max := 0$;

for each $j \in \mathcal{I}$ **do**

- Let G' be the subgraph of G induced on $\mathcal{H} \cup S \cup \{j\}$, with the same capacities as G. (If there are duplicates in $S \cup \{j\}$, we create one point for each duplicated element.)
- For each house, transform its vertex capacity constraint to an edge capacity constraint by replacing the house h with two nodes, in_h and out_h; and connect in_h to out_h using a directed edge with capacity cap_h; all incoming edges toward the house go to in_h and all out-going edges from h come from out_h.
- Add two vertices s and t to G', edges of capacity w_h from s to each $h \in \mathcal{H}$, and edges of capacity cap_i from each $i \in S \cup \{j\}$ to t.
- Find the maximum flow from s to t in G'; Let f be the value of this flow.
- **if** $f > max$, **then**
  $max := f$; $bestITAP := j$;

endfor;

$S := S \cup \{bestITAP\}$; $Flow := max$;

endwhile;

end.

Figure 2: Greedy placement algorithm in the ideal link model

Refer to [24] for the proofs of the above theorem and lemma. Based on Theorem 2, we have the following corollary.

COROLLARY 4. *Let* N *be the number of houses. The approximation factor of the greedy algorithm in the ideal link model is* $\ln(N)$ *when the capacities of edges and vertices are integer-valued and every house has either zero or one unit of demand.*

Remark 1. Corollary 4 in combination with Theorem 1, shows that this algorithm achieves the best possible (worst-case) approximation ratio for the graph theoretic model when every house has either zero or one unit of demand. Furthermore, even though in our model we allow fractional routing of the flow, our greedy algorithm always finds an integral solution in this case, i.e., the demand from each house will be served through one path to an open ITAP. This is a consequence of the integrality theorem [9].
Remark 2. Notice that $\ln(D)$ is the *worst-case* bound for heterogeneous demands. To make the worst-case bound tighter, we can normalize house demands, edge capacities, and node capacities before we apply the greedy placement algorithm. This yields a lower approximation factor, since D is reduced after normalization. Moreover, as we will show later in this section, in practice the greedy algorithm performs quite close to the optimal, and much better than the worst-case bounds, $\ln(D)$ or $\ln(N)$.

4.2 General Link Model

The problem of efficient ITAP placement is more challenging when the throughput along a path varies with the path length. This corresponds to the general link model introduced in Section 3.1. In this section, we first formulate the problem for a link model with an arbitrary throughput degradation function, and then present efficient heuristics for two variants of this degradation function.

4.2.1 Problem Formulation

We formulate the placement problem for the general link model as an integer linear program shown in Figure 3. In this program $x_{e,h,l,j}$ denotes the total amount of flow routed from house h to the ITAPs using a path of length l when edge e is the j'th edge along the path. Variable y_i is an indicator of the number of ITAPs opened in the equivalence class i, and each house h has w_h units of traffic to send. The throughput degradation function for a path of length l is denoted by $g(l)$. L is an upper bound on the number of hops on a communication path, and if there is no such upper bound, we set $L = |\mathcal{H}|$. The other variables in the program are similar to the ones used by the program presented in Figure 1.

$$
\begin{array}{lll}
\text{minimize} & \sum_{i \in \mathcal{I}} y_i & \\
\text{subject to} & \sum_{e=(v,h')} x_{e,h,l,j} = \sum_{e=(h',v)} x_{e,h,l,j+1} & \forall h, h' \in H, h' \neq h, \\
 & & l, j \in \{1, \ldots, L\}, j < l \\
 & \sum_{e=(h,v),l} x_{e,h,l,1} \geq w_h & \forall h \in H \\
 & \sum_{h,l,j \leq l} g(l)\, x_{e,h,l,j} \leq \mathrm{Cap}_e & \forall e \in E(G) \\
 & \sum_{h', e=(v,h), l, j \leq l} g(l)\, x_{e,h',l,j} \leq \mathrm{Cap}_h & \forall h \in H \\
 & \sum_{h', e=(v,i), l, j \leq l} g(l)\, x_{e,h',l,j} \leq \mathrm{Cap}_i y_i & \forall i \in I \\
 & \sum_{e=(u,i), l, j \leq l} x_{e,h,l,j} \leq w_h y_i & \forall i \in I, h \in H \\
 & x_{e,h,l,j} \geq 0 & \forall e \in E(G), h \in H, \\
 & & l, j \in \{1, \ldots, L\}, j \leq l \\
 & y_i \in \{0, 1, 2, \ldots\} & \forall i \in I
\end{array}
$$

Figure 3: LP formulation for the general link model, where $g(l)$ models throughput degradation with increasing hop-count.

The following theorem is an immediate consequence of Theorem 1, as the ideal link model is a special case of the general link model, when $g(l) = 1$.

THEOREM 5. *It is NP-hard to find a minimum number of ITAPs to cover a neighborhood for a general link model.*

4.2.2 Our Approach: Greedy Placement

The high-level idea of the greedy algorithm is similar to the one presented for the ideal link model. We iteratively select ITAPs to maximize the total user demands satisfied. The new challenge is to determine a greedy move in this model. Unlike in the ideal link model, we cannot compute the total satisfied demands by modeling it as a network flow problem since the amount of flow now depends on the path length. As we will describe below, this computation can be done by solving a linear program, or by using a heuristic.

Expensive algorithm for the general link model: Without making assumptions about $g(l)$, we can compute the total satisfied user demands, for a given set I' of ITAPs, by solving a slightly modified LP problem than the one in Figure 3. In this linear program, we replace the variable y_i by the number of occurrences of i in I' (This amounts to removing all the variables corresponding to edges ending in ITAP positions outside I' and removing inequalities containing these variables). The objective will be to maximize $\sum_h \sum_{e=(h,v),l} x_{e,h,l,1}$, which corresponds to maximizing the satisfied demands. We also modify the second constraint to be $\sum_{e=(h,v),l} x_{e,h,l,1} \leq w_h$ in order to limit the maximum flow from each house h.

In theory, solving a linear program takes polynomial time. However, in practice an LP solver, such as cplex [10], can only handle small-sized networks under this model due to the fast increase in the number of variables and constraints with the network size.

Below we develop more efficient algorithms for two special forms of $g(l)$: (i) bounded hop-count: $g(l) = 1$ for all $l \leq k$, and $g(l) = \infty$ for $l > k$, and (ii) smooth degradation: $g_l = l$ for all l.

Efficient algorithm for the bounded hop-count model: We can use the following greedy algorithm to find the total demands satisfied by a given set of ITAPs. The hop-count constraint suggests we should favor short paths in the graph. Therefore, in each iteration, the algorithm finds the shortest path from demand points to opened ITAPs in the residual graph, routes one unit of flow along this path, and decreases the capacities of the edges on the path by one in the residual graph. This is continued until the shortest path found has length more than the hop-count bound. This algorithm is similar to the algorithm proposed in [14] for a similar problem. While this heuristic does not guarantee computing the maximum flow (so each greedy step is not local optimal), it works very well in practice as shown in Section 6.2.1.

Efficient algorithm for the smooth throughput degradation model: When $g(l) = l$ or $throughput_l = \frac{1}{l}$, the total demands satisfied by a set of ITAPs are given by the expression: $maximize \sum_{p_i \in P} \frac{1}{|p_i|}$ where P is a collection of edge-disjoint paths in the graph, and $|p_i|$ denotes the length of the path p_i. Therefore to maximize this objective function, our heuristic should prefer imbalance in path lengths, and this motivates the following algorithm.

As the heuristic for the bounded hop-count model, in the smooth throughput degradation model we compute the total satisfied demands by the selected ITAPs through iteratively removing shortest paths in the residual graph. However, we make the following modifications. First, since we no longer have bounds on hop-count, we continue picking paths until there is no path between any demand point and any open ITAP. Second, to ensure the throughput follows $throughput(l) = 1/l$, we compute the demand satisfied along each path p, denoted as SD_p, according to the throughput function after we obtain all the paths. The total satisfied demands are the sum of SD_p over all paths p. Although this algorithm does not always find the maximum flow (so each greedy step is not local optimal), it yields very good performance as shown in Section 6.2.2.

4.3 Alternative Algorithms

In the rest of this paper, we compare our greedy placement algorithm to four alternative approaches.

4.3.1 Augmenting Placement

The idea of the augmenting placement algorithm is similar to the greedy algorithm. The main difference is that in the augmenting algorithm we do not make a greedy move; instead we are satisfied with any ITAP that increases the total amount of demand satisfied. More specifically, we search over the set of possible ITAP locations, and open the first ITAP we see that results in an increase in the amount of satisfied demand when opened together with the already opened ITAPs.

The augmenting placement algorithm can be applied to all three wireless link models with the following difference. In the ideal link model, we compute the total amount of demand satisfied under a given set of ITAPs by finding the maximum flow in the graph; whereas in the general link models, we use the heuristics described in Section 4.2.2 to derive the total amount of demand satisfied.

4.3.2 Clustering-based Placement

We compare our placement algorithms to the clustering-based scheme, proposed in [3]. The basic idea of the algorithm is to partition the network nodes into a minimum number of disjoint clusters, and place an ITAP in each cluster. We use the Greedy Dominating Independent Set (DIS) [3] heuristic to determine a set of clusterheads, which are used as possible ITAP locations. The nodes are then clustered to ensure that each node is associated with the closest clusterhead, and a shortest path tree rooted at the clusterhead is used for sending packets from and delivering packets to the cluster. The cluster is further divided into sub-clusters if either the weight or relay-load constraints are violated. The weight constraint specifies that an ITAP can serve nodes as long as the sum of their demands does not exceed the capacity of the ITAP, and the relay-load constraint specifies an upper bound on the maximum flow that can go through a node in the neighborhood cluster. We refer the reader to [3] for more details of this algorithm.

To apply the clustering-based algorithm for the ideal link model, in our simulations we use the ITAP capacity instead of wireless capacity when checking the weight constraint of placing an ITAP at a particular house; this is necessary since the ITAP capacity can be greater than the wireless capacity in our simulations. This ensures a fair comparison of the clustering algorithm with our placement schemes.

To apply the algorithm to the bounded-hop count model, we make the following modification. We divide a cluster into sub-clusters not only when the weight or relay-load constraints are violated, but also when the distance between any node and its clusterhead exceeds the hop-count threshold. The algorithm, however, does *not* apply to the smooth throughput degradation model.

4.3.3 Random Placement

This algorithm randomly places an ITAP at a house iteratively until all the user demands are satisfied. To avoid wasting resource, it ensures that each house has at most one ITAP. This approximates un-coordinated deployment of ITAPs in a neighborhood, and gives a baseline to evaluate the benefits of the more sophisticated algorithms presented above.

As the augmenting algorithm, there are three variants of random placement algorithms for different wireless link models. They differ in how we compute the total demand satisfied under a given set of ITAPs. We run the maximum flow algorithm to compute the satisfied demand under the ideal link model, and apply the heuristics in Section 4.2.2 to compute the satisfied demand under the general link models.

4.3.4 Lower Bound

It is useful to compare our algorithms with the optimal solution. However, our problem is NP-hard, and it is too expensive to derive an optimal solution. Therefore we compare our algorithms with the lower bounds. We derive the lower bound by relaxing the integer constraints on y_i in the LP (in Figure 1) and solving the relaxed LP problem using cplex [10]. The lower bound is a useful data point to compare with, as it gives an upper bound on the difference between a practical algorithm and the optimal.

We use the same scheme to derive lower bounds for all three link models. Note that ideally we would like to derive the lower bound for the general link models by relaxing the integrality constraint in Figure 3, and solving the relaxed linear program. However, although in theory linear programs can be solved in polynomial time, we were unable to solve the program in Figure 3 for large networks due to the memory constraints. So in our performance evaluation section, we use the solution to the LP formulation of Figure 1 for the ideal link model, as the lower bound for the general case too. This lower bound is always correct, since the ideal link model is a relaxation of the general model. However, it might not be tight, since it ignores the throughput degradation with hop count, and therefore requires fewer ITAPs than necessary. However, in Section 6.2.1 and Section 6.2.2 we show that the results from our greedy and augmenting algorithms are still close to these loose lower bounds.

5. VALIDATION

To validate the wireless link models used in this paper, we run simulations in Qualnet [26], a commercial network simulator. More specifically, given a neighborhood layout, the placement algorithms determine the ITAP locations and the set of paths each house uses to reach the ITAPs. We use the same neighborhood layout and ITAP locations in the simulations. Every node in the simulation uses an omni-directional antenna and 802.11b MAC, with the communication range and interference range being 195 meters and 376 meters, respectively. Every house sends CBR traffic to the ITAPs at the rate specified by the placement algorithms' output. To support multi-path routing, we implemented probabilistic source-routing in Qualnet, where the paths used in source routing and the probability that each path is chosen are based on the placement algorithms' output.

As shown in Figure 4, the ITAPs, determined using the smooth degradation model, satisfy the user demands to a great extent: around 80% houses have their demands completely satisfied when houses are randomly placed in 1000*1000 m^2, and all houses receive their demands when houses are randomly placed in 1500 * 1500 m^2. The better performance in the latter scenario comes from the fact that the larger separation among houses lowers interference among cross traffic. Note that even for the former case, we can further improve the clients' throughput by over-provisioning. As shown in the same figure, with over-provisioning (assuming that each user's demand is 500 Kbps when the actual demand is 208 Kbps), most of the clients' demands are satisfied.

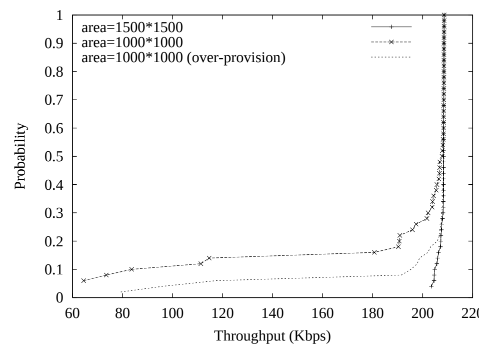

Figure 4: Validation of general link models: CDF of clients' throughput, where $N = 50$, $WC = 5Mbps$, and $w_h = 208Kbps\ \forall h \in H$.

Since ideal link and bounded hop-count models are more optimistic about the impact of interference, they are more suitable for the environments with efficient spectral use (e.g., when directional antennas and/or multiple radios are used). As part of our future work, we plan to evaluate how well these two models capture the impact of wireless interference under such environments.

6. PERFORMANCE EVALUATION

In this section, we evaluate the performance of different placement algorithms using various network topologies, house demands, and link models.

6.1 Performance Under the Ideal Link Model

First, we look at the performance under the ideal link model under various scenarios. We use the following notations in our discussion.

- N: the number of houses
- WC: a wireless link's capacity
- IC: an ITAP's capacity
- CR: communication radius

- HR: average inter-house distance
- w_h: house h's demand

We compare the performance of different algorithms by varying each of the above parameters. In our evaluation, we use both random topologies and a real neighborhood topology. The random topologies are generated by randomly placing houses in a region of size $N * N$, and varying the communication radius. The real neighborhood topology contains 105 houses, spanning over a region of 1106m*1130m. (We cannot reveal the source of the data for confidentiality.) The average inter-house distance in the real topology is 74 meters. (We determine the inter-house distance by averaging the distance between a house and its immediate neighbors.) Unless otherwise specified, for the same parameter setting, we run simulations three times, and report the average number of ITAPs required for each placement algorithm.

Effects of the communication radius: We start by examining the effect of communication radius (CR) on the placement algorithms. It is easy to see from the problem formulation that only the ratio, $\frac{CR}{HR}$, is important. Therefore in our evaluation, we vary the communication radius from 1 to 50, while fixing the inter-house distance by randomly placing 100 houses in an area of 100*100, which yields an average inter-house distance of 4.5 - 6.

Figure 5 illustrates the number of ITAPs required on varying CR. We make the following observations. First, we see that an increase in CR results in a greater overlap of wireless coverage of the houses, and therefore fewer ITAPs are sufficient to satisfy the house demands. Second, comparing the performance across different algorithms, we observe that the greedy algorithm performs very close to the lower bound over all cases. Interestingly, the augmenting algorithm performs quite well, too. The good performance of the augmenting algorithm comes from the requirement that new ITAPs should lead to throughput improvement, which avoids wasting resource on the already covered region. This is especially useful after several ITAPs have been placed, since at this point only a few locations remain that can further increase the satisfied demands.

In contrast, the clustering and random-house placement schemes perform much worse. Compared to the greedy strategy, both schemes often require 2 to 10 times as many ITAPs. Note that when the communication radius is very large, the clustering algorithm yields worse performance than the random-house placement. This is because in the clustering algorithm data dissemination follows a shortest path tree, instead of maximizing the total amount of flow that can be pushed to the ITAPs. In comparison, the other algorithms, including the random-house placement, run the network algorithm to maximize the total satisfied demands.

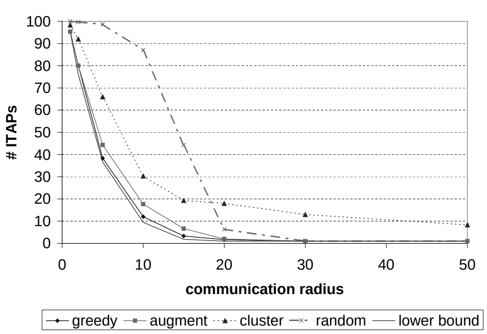

Figure 5: Ideal link model: varying communication radius, where $N = 100$, $WC = 6$, $IC = 100$, and $w_h = 1\ \forall h \in H$.

Effects of network size: Next we study the impact of network size on the placement algorithms. We randomly place N houses in an $N * N$ area while fixing the communication radius to 10. Figure 6 shows the number of required ITAPs using the different placement algorithms for various network sizes. As we would expect, an increase in the number of houses leads to a larger number of ITAPs required to cover the neighborhood. Moreover, the greedy algorithm continues to perform very well, with its curve mostly overlapping with the lower bound. The augmenting algorithm performs slightly worse, whereas the clustering and random algorithms perform much worse – requiring up to 5 and 8 times as many ITAPs, respectively. In addition, the benefit of greedy algorithm increases as the network gets larger.

Figure 6: Ideal link model: varying the number of nodes, where CR= 10, $WC = 6$, $IC = 100$, and $w_h = 1\ \forall h \in H$.

Effects of wireless link capacity: We also study the effects of wireless bandwidth on the placement algorithms. As shown in Figure 7, the relative ranking of the algorithms stays the same. The effect of bandwidth is only pronounced when it is very limited. For example, when the wireless bandwidth is equal to a single house's demand, the number of ITAPs required is considerably large. As the bandwidth increases and the wireless link is no longer the bottleneck, the number of required ITAPs remains the same with a further increase in the wireless link capacity.

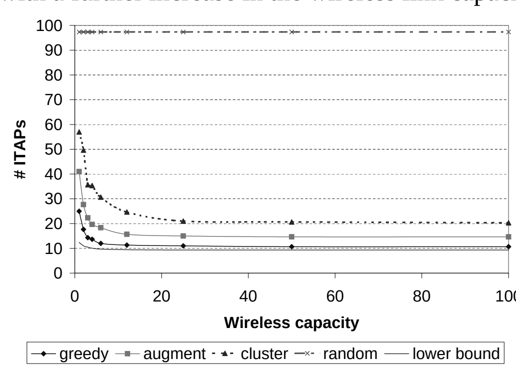

Figure 7: Ideal link model: varying wireless link capacity, where $N = 100$, $CR = 10$, $IC = 100$, and $w_h = 1\ \forall h \in H$.

Effects of the ITAP capacity: We compare the placement algorithms by varying the ITAP capacity. As Figure 8 shows, when ITAP capacity is small and hence is a bottleneck, the number of required ITAPs decreases proportionally with an increase in ITAP capacity. As the ITAP capacity is large enough and no longer the bottleneck, the number of required ITAPs is unaffected by a further increase in ITAP capacity. Moreover, the relative performance of different placement algorithms is consistent with the previous scenarios.

Effects of heterogeneous house demands: So far we have considered homogeneous house demands (i.e., each house generates one unit demand). A number of studies show that realistic user

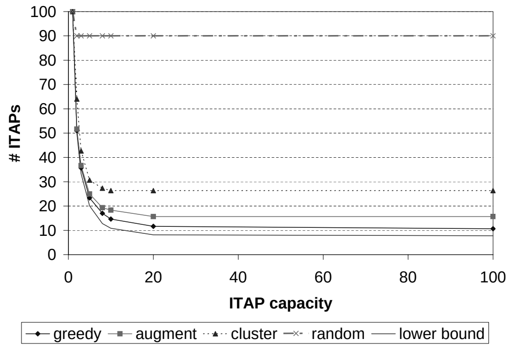

Figure 8: Ideal link model: varying ITAP capacity, where $N = 100$, $CR = 10$, $WC = 6$, and $w_h = 1\ \forall h \in H$.

demands are very heterogeneous, and often exhibit Zipf-like distributions [5, 6]. Motivated by these findings, below we evaluate the placement algorithms when house demands follow a Zipf distribution. Figure 9 summarizes our results. As it shows, the results are qualitatively the same as those of using the homogeneous house demands. The greedy algorithm continues to out-perform the others significantly and yield nearly optimal solutions.

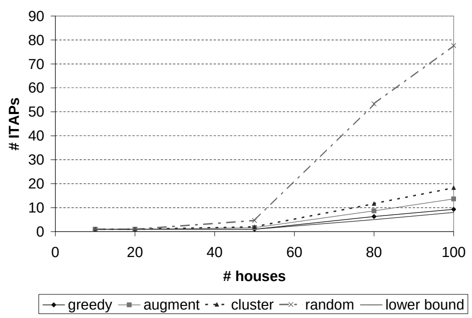

Figure 9: Ideal link model: varying the number of nodes, where $CR = 10$, $WC = 6$, $IC = 100$, and the house demands follow a Zipf-distribution.

Real neighborhood topology: Finally we evaluate the placement algorithms using a real neighborhood topology of 105 houses. We again use Zipf-distributed house demands. As shown in Figure 10, initially when the communication range is too small, most houses are unreachable from other houses, and therefore all the algorithms require close to 105 ITAPs. As the communication range increases, fewer ITAPs are needed to cover the neighborhood. At the extreme, when the communication range reaches 250 meters, the neighborhood forms a single connected component, and therefore most algorithms require only one ITAP. (Note that this is only true for the ideal model. As shown in the next section, when considering wireless interference, we often needs more ITAPs even for a single connected component.) Moreover, the greedy algorithm performs close to optimal over all communication radii considered.

6.2 Performance Under the General Link Models

In this section, we evaluate the performance of placement algorithms under two general link models, namely bounded hop-count and smooth throughput degradation models.

6.2.1 *Bounded Hop-count Model*

We compare the placement algorithms for bounded-hop count model by varying the hop-count threshold, communication radius, and neighborhood topology.

Effects of hop-count threshold: First we compare the placement

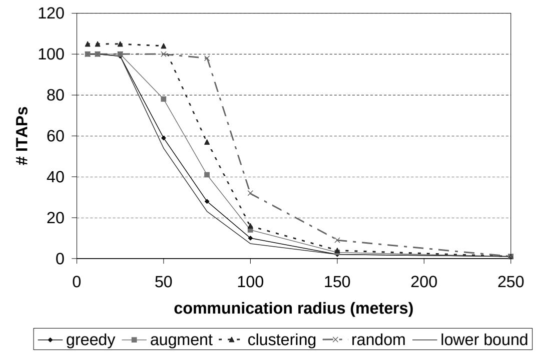

Figure 10: Ideal link model: results of a real neighborhood topology with various communication radii, where $N = 105$, $WC = 6$, $IC = 100$, and the house demands follow a Zipf distribution.

algorithms by varying the hop-count threshold. As shown in Figure 11, when the hop-count threshold increases, the effect of hop-count reduces, since all or most paths are within hop-count limit. Comparing the different placement algorithms, we see that the greedy placement performs very close to the lower bound, especially for large hop-count thresholds. When the hop-count threshold is small, the gap between the lower bound and greedy algorithm is slightly larger, since the lower bound ignores throughput degradation with the hop-count, and is not as tight. Compared to the greedy algorithm, the augmenting algorithm requires 50% more ITAPs; the clustering algorithm in [3], requires 2 - 3 times as many ITAPs; and the random algorithm requires 4 to 8 times as many ITAPs.

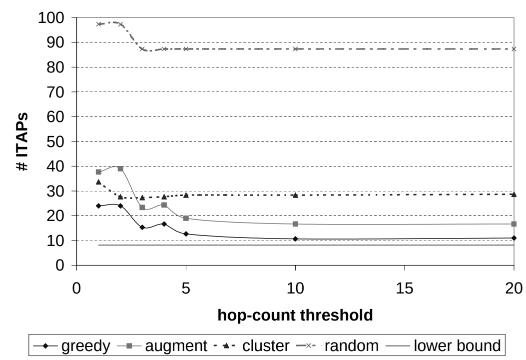

Figure 11: Bounded hop-count model: varying the hop-count threshold, where $N = 100$, $CR = 10$, $WC = 6$, $IC = 100$, and $w_h = 1$ $\forall h \in H$.

Effects of communication radius: Next we fix the hop-count threshold to 3, and vary the communication radius. As Figure 12 shows, an increase in communication radius reduces the number of ITAPs required to cover the neighborhood. Moreover the greedy continues to perform significantly better than the alternatives. We observe similar results for other hop-count thresholds.

Real neighborhood topology: We also evaluate the placement algorithms using the real neighborhood topology. As shown in Figure 13, the results are qualitatively the same as the random topologies. The greedy algorithm performs very close to the lower bound for all the communication radii considered.

6.2.2 *Smooth Throughput Degradation Model*

Next we empirically study the placement algorithms for the smooth throughput degradation model.

Effects of communication radius: Figure 14 compares the performance of different placement algorithms for the smooth throughput degradation link model on varying the communication radius. As we can see, the number of required ITAPs decreases as the communication radius increases. The gap between different algorithms' performance is the largest when the communication radius

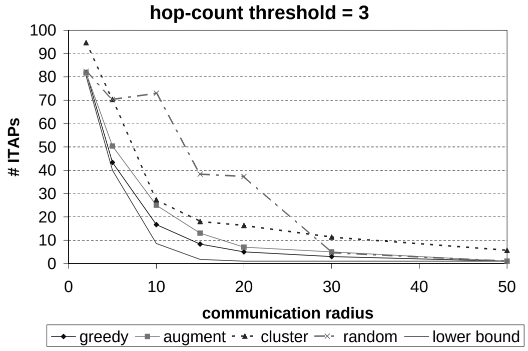

Figure 12: Bounded hop-count model: varying the communication radius, where $N = 100$, $WC = 6$, $IC = 100$, hop-count threshold = 3, and w

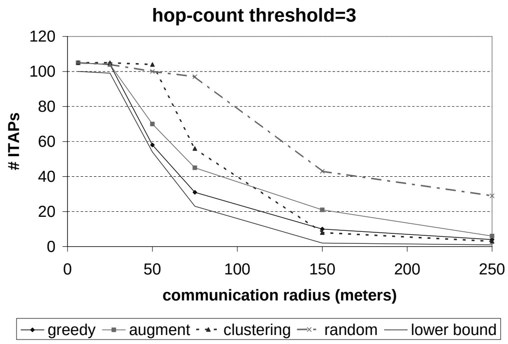

Figure 13: Bounded hop-count model: results of a real neighborhood topology for various communication radii, where $N = 105$, $WC = 6$, $IC = 100$, hop-count threshold = 3, and the house demands follow a Zipf distribution.

is between 5 and 20 (The average inter-house is around 5.). This occurs because when the radius is very small, most houses are disconnected from one another, and therefore the number of ITAPs required is close to the number of houses regardless of placement algorithms; when the radius is very large, most houses are reachable from one another within one or few hops, and the number of ITAPs required becomes close to 1. In comparison, for medium communication radius, which is the most likely scenario in practice, the gap between the different algorithms is most significant. This is especially so when we compare the random placement with the other two. Note that the lower bound, which is derived by ignoring the impact of hop-count on throughput, is more loose for this scenario. Nevertheless the greedy is still competitive when compared with these loose lower bounds.

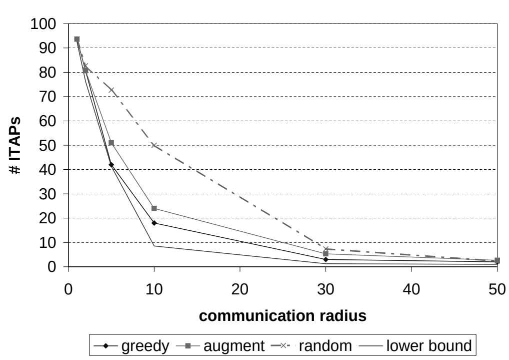

Figure 14: Smooth throughput degradation model: varying the communication radius, where $N = 100$, $WC = 6$, $IC = 100$, and $w_h = 1\ \forall h \in H$.

Real neighborhood topology: Figure 15 shows the results from the real neighborhood topology. As we can see, the greedy placement continues to perform well, yielding close to optimal performance.

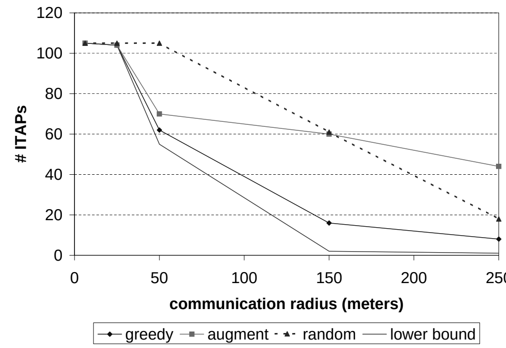

Figure 15: Smooth throughput degradation model: results of a real neighborhood topology for various communication radii, where $N = 105$, $WC = 6$, $IC = 100$, and the house demands follow a Zipf distribution.

7. PRACTICAL CONSIDERATIONS

We now enhance our algorithms to take into account two important requirements: providing fault tolerance and handling workload variation.

7.1 Providing Fault Tolerance

A practical solution to the ITAP placement problem should ensure Internet connectivity to all the houses in the neighborhood, even in the presence of a few ITAP and house failures. In this section we present an enhancement to our problem by incorporating this fault tolerance constraint. Fault tolerance is achieved by providing multiple independent paths from a house to ITAPs [1], and over-provisioning the delivery paths. Over-provisioning is a scheme that allocates more flow to a house than its demand, and therefore helps in providing QoS guarantees even when there are a few failures.

7.1.1 Problem Formulation

Here we formulate the placement problem with the fault tolerance constraint. Let each house have one unit of demand, and d independent paths to reach the ITAPs; the average failure probability of a path be p; and the over-provisioning factor be f (i.e., each independent path allocates $\frac{f}{d}$ capacity to a house, and the total capacity allocated to a house by d independent paths is f).

Since for every house, there are d independent paths from this house to ITAPs and the probability of failure of each path is p, the probability that exactly i of these paths fail is $\binom{d}{i} p^i (1-p)^{d-i}$. In this case, the amount of traffic that can be delivered is $\min(\frac{(d-i)f}{d}, 1)$. Therefore, the expected fraction of the traffic from a house that can reach an ITAP, $S(f, p, d)$, is given by the following formula.

$$S(f,p,d) = \sum_{i=0}^{d} \binom{d}{i} p^i (1-p)^{d-i} \min(\frac{(d-i)f}{d}, 1).$$

Given the expected guarantee desired by the home users, $S(f, p, d)$, we can use the above expression to derive the overprovision factor, f, based on path failure probability and the number of independent paths. We now provide fault tolerant LP formulations for the ideal and general link models.

Ideal Link Model with the Fault Tolerance Constraint : Figure 16 provides an LP formulation of the fault tolerant problem for the

[1]These can be different ITAPs since the ultimate goal is to provide Internet connectivity irrespective of which ITAP is used.

ideal case, i.e. when throughput is independent of the path length. For each edge e and each house h, the variable $x_{e,h}$ indicates the amount of flow from h to ITAPs that is routed through e. Also, for each ITAP i, the variable y_i denotes the number of ITAPs opened in equivalence class i. The above integer LP is similar to the one in Section 4.1.1. The differences are as follows: (i) the constraint $\leq w_h$ added to the first inequality, (ii) a change in the second constraint from w_h to $w_h d$ in the amount of flow originating from each house, and (iii) a multiplicative factor of $\frac{f}{d}$ on the left-hand side of the capacity constraints (since the amount of capacity each path allocates to each house is $\frac{f}{d}$). The first modification ensures that the flow from each house is served by independent paths; (ii) and (iii) are for the over-provisioning purpose.

$$
\begin{array}{lll}
\text{minimize} & \sum_{i \in \mathcal{I}} y_i & \\
\text{subject to} & \sum_{e=(v,h')} x_{e,h} = \sum_{e=(h',v)} x_{e,h} \leq w_h & \forall h, h' \in \mathcal{H}, h' \neq h \\
& \sum_{e=(h,v)} x_{e,h} - \sum_{e=(v,h)} x_{e,h} \geq w_h d & \forall h \in \mathcal{H} \\
& \frac{f}{d} \sum_{h} x_{e,h} \leq \text{Cap}_e & \forall e \in E(G) \\
& \frac{f}{d} \sum_{h', e=(v,h)} x_{e,h'} \leq \text{Cap}_h & \forall h \in \mathcal{H} \\
& \frac{f}{d} \sum_{h', e=(v,i)} x_{e,h'} \leq \text{Cap}_i y_i & \forall i \in \mathcal{I} \\
& \sum_{e=(v,i)} x_{e,h} \leq w_h y_i & \forall i \in \mathcal{I}, h \in \mathcal{H} \\
& x_{e,h} \geq 0 & \forall e \in E(G), h \in \mathcal{H} \\
& y_i \in \{0, 1, 2, \ldots\} & \forall i \in I
\end{array}
$$

Figure 16: LP formulation for the ideal link model with fault tolerance constraints, where d is the number of independent paths, and f is the over-provision factor.

General Link Models with the Fault Tolerance Constraint : Now we look at the problem where the throughput of a connection is a function of the number of hops it traverses. The integer LP in Figure 17 is similar to the one introduced in Section 4.2.1, with a few modifications similar to the ones described above.

The problem without fault tolerance constraints is just a special case of the one with these constraints when $d = 1$ and $f = 1$. Since the former was proved NP-hard in Sections 4.1 and 4.2, the latter problem is also NP-hard.

THEOREM 6. *It is NP-hard to find a minimum number of ITAPs required to cover a neighborhood while providing fault tolerance.*

7.1.2 Placement Algorithms

We present the greedy, augmenting and random placement algorithms for this problem. The high-level ideas behind these algorithms are the same as before: ITAPs are opened iteratively until all the user demands are satisfied. The difference between the algorithms is in the criterion used to pick an ITAP in each iteration. For the greedy algorithm, it is the ITAP that leads to the maximum increase in the supported demands; in the augmenting algorithm, it is the first ITAP that leads to an increase in the supported demands; and it is a random house in the random placement.

The above algorithms differ from the ones presented in previous sections in how they compute the total demands supported by a given set of ITAPs. For the ideal case, we compute the satisfied demands by slightly modifying the LP in Figure 16, and solving the resulting LP. The objective function is changed to be maximizing $\sum_h (\sum_{e=(h,v)} x_{e,h} - \sum_{e=(v,h)} x_{e,h})$, which corresponds to maximizing the supported demands. The variables y_i are replaced by the number of occurrences of i in I'. Furthermore, the second

$$
\begin{array}{lll}
\text{minimize} & \sum_{i \in \mathcal{I}} y_i & \\
\text{subject to} & \sum_{e=(v,h')} x_{e,h,l,j} = \sum_{e=(h',v)} x_{e,h,l,j+1} & \forall h, h' \in H, h' \neq h, \\
& & \quad l, j \in \{1, \ldots, L\}, j < l \\
& \sum_{e=(v,h'),l,j \leq l} x_{e,h,l,j} \leq w_h & \forall h, h' \in H, h \neq h' \\
& \sum_{e=(h,v),l} x_{e,h,l,1} \geq w_h d & \forall h \in H \\
& \frac{f}{d} \sum_{h,l,j \leq l} g(l)\, x_{e,h,l,j} \leq \text{Cap}_e & \forall e \in E(G) \\
& \frac{f}{d} \sum_{h', e=(v,h), l, j \leq l} g(l)\, x_{e,h',l,j} \leq \text{Cap}_h & \forall h \in H \\
& \frac{f}{d} \sum_{h', e=(v,i), l, j \leq l} g(l)\, x_{e,h',l,j} \leq \text{Cap}_i y_i & \forall i \in I \\
& \sum_{e=(u,i),l,j \leq l} x_{e,h,l,j} \leq w_h y_i & \forall i \in I, h \in H \\
& x_{e,h,l,j} \geq 0 & \forall e \in E(G), h \in H, \\
& & \quad l, j \in \{1, \ldots, L\}, j \leq l \\
& y_i \in \{0, 1, 2, \ldots\} & \forall i \in I
\end{array}
$$

Figure 17: LP formulation for the general link model with the fault tolerance constraints.

constraint is changed to $\sum_{e=(h,v)} x_{e,h} - \sum_{e=(v,h)} x_{e,h} \leq w_h d$ in order to limit the maximum flow from a node. For the general link model, we compute the satisfied demands by applying similar modifications to the LP in Figure 17.

We also compare the above algorithms to the lower bound, derived by relaxing the integrality constraint and solving the relaxed linear program.

7.1.3 Performance Evaluation

We now evaluate the performance of our placement algorithms under fault tolerance constraints. Both experiments are for the ideal link model.

Effects of the number of independent paths: Figure 18 compares the placement algorithms on varying the number of independent paths, d. We see that for all algorithms the number of ITAPs required increases linearly with the number of independent paths. Moreover, the results of the greedy algorithm are very close to the lower bound, and significantly better than the other two.

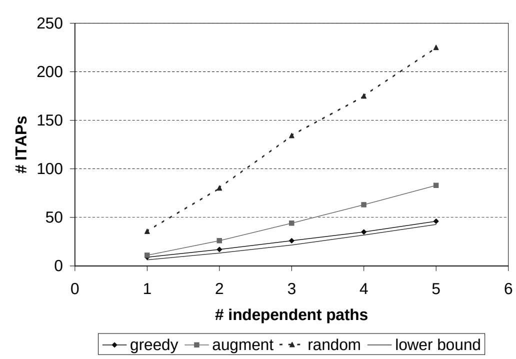

Figure 18: Ideal link model: varying the number of independent paths d, where $N = 50$, $CR = 7$, $WC = 6$, $IC = 100$, $f = 2$, and $w_h = 1\ \forall h \in H$.

Effects of the over-provisioning factor: Figure 19 shows the performance of placement algorithms for various over-provisioning factors, f. As we can see, the relative performance of the algorithms is the same as before, and the required number of ITAPs using the greedy algorithm is very close to the lower bound. Moreover, the number of ITAPs is mostly unaffected by the over-provision factor. This occurs because increasing the over-provision factor is equivalent to fixing the house demand while reducing the wireless capacity. As long as the reduced wireless capacity is still not a bottleneck, then the number of required ITAPs is unaffected, as also shown in Section 6.1.

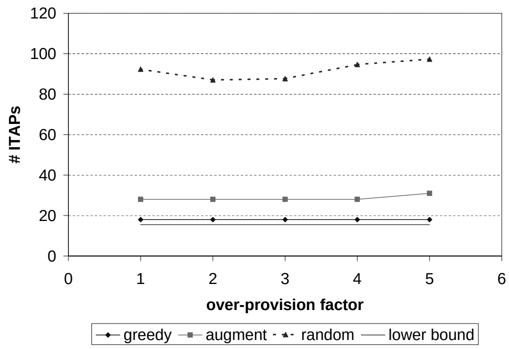

Figure 19: Ideal link model: varying the over-provision factor f, where $N = 50$, $CR = 7$, $WC = 6$, $IC = 100$, $d = 2$, and $w_h = 1$ $\forall h \in H$.

7.2 Handle Workload Variation

So far we have considered placing ITAPs based on static user demands. In practice, user demands change over time, and often exhibit diurnal patterns [6, 20, 27]. Since it is not easy to change ITAP locations once they are deployed, we would like to place ITAPs such that they can handle demands over all periods. In this section, we describe and evaluate two approaches to handle variable workloads. While our discussion focuses on the non fault-tolerant version of the placement problems, the ideas carry over easily to the fault-tolerant version as well.

One approach to take into account workload change is to provision ITAPs based on the peak workload. That is, if $w[h][t]$ denotes the demand of house h at time t, we use $\max_t w[h][t]$ as the demand for house h, and feed this as an input to the placement algorithms described in the previous sections. We call this approach *peak load based placement*. This algorithm is simple, but may sometimes be wasteful, e.g., when different houses' demands peak at different times.

To improve efficiency without sacrificing user performance, below we explore how to optimize ITAP locations for demands over multiple time intervals.

More formally, the problem can be stated as follows. Each house h has demand $w[h][t]$ at time t. Our goal is to place a set of ITAPs such that at any time t, they can serve all the demands generated at t, i.e., $w[h][t]$ for all h's.

Below we describe a greedy heuristic with a logarithmic worst-case bound for the ideal link model. The same idea applies to other link models. The high-level idea is to iteratively place the ITAP such that together with the already opened ITAPs it maximizes the total demands served. Unlike in the previous section, here the total demands include demands over multiple time intervals. More specifically, we place an ITAP such that it maximizes $\sum_{t \in T} SD_t$, where SD_t is the total satisfied demands at time t. This can be computed by changing the greedy algorithm of Section 4.1.2 as follows. In every iteration, for every $j \in \mathcal{I}$ and $t \in T$, we construct the graph G' as in the algorithm of Section 4.1.2 based on the demands at time period t. Then we compute the maximum flow $f_{j,t}$ in this graph. After these computations, we pick the ITAP j that maximizes $\sum_t f_{j,t}$, and open it. We call this algorithm *multiple-demand-based greedy placement* (M-greedy, for short). In the following theorem, we prove a worst-case bound on the M-greedy's performance in the ideal link model.

THEOREM 7. *Consider the ITAP placement problem in the ideal link model with integral demands and capacities, and let D_t be the total demand in period t. The approximation factor of the M-greedy algorithm for this problem is at most $\ln(\sum_t D_t)$. In other words, if the optimal algorithm requires K ITAPs to serve demands over L time periods, then the M-greedy requires at most $K \ln(\sum_t D_t)$ ITAPs.*

Refer to [24] for the proof. Based on the above theorem, we have the following corollary.

COROLLARY 8. *Let L denote the total number of periods, and N denote the number of houses. The approximation factor of the M-greedy in the ideal link model is $\ln(LN)$, when the capacities of edges and vertices are integer-valued and every house has either zero or one unit of demand at any time t.*

This is easy to see because $\sum_t D_t \le LN$.

For the peak load based placement algorithm, the approximation factor can be upper bounded as follows. The cost of the optimal solution based on the peak load is at most the sum of the costs of the optimal solutions in each time period (because by simply taking the union of the solutions for each time period, we obtain a solution for the peak loads). Therefore, the approximation factor of the greedy placement using the peak load is at most a factor of $\sum_t (\ln D_t)$. When $D_t = D_m$ for all t's, its cost is at most $L \ln(D_m)$. This is roughly L times the approximation factor of the M-greedy algorithm proved above.

7.2.1 Performance Evaluation

In addition to the worst-case analysis, we evaluate the effectiveness of the placement algorithms empirically as follows. We randomly select a fraction of houses to be active at daytime, and the remaining fraction to be active at nighttime. The demands distributed among the active houses follow a Zipf-distribution. Figure 20 shows the number of ITAPs required by the M-greedy and peak-greedy algorithms. For reference, we also include the curves of greedy-morning and greedy-evening, which represent the number of ITAPs required to satisfy only the morning workload ($numITAP_m$) or only the evening workload ($numITAP_e$) but not both. It is easy to see that the number of ITAPs required to satisfy both workloads should usually be no less than $max(numITAP_m, numITAP_e)$. (It may sometimes be less, since the greedy algorithm does not guarantee to find the smallest number of ITAPs to cover the neighborhood.)

As shown in Figure 20, the results from the peak-greedy and M-greedy are both close to $max(numITAP_m, numITAP_e)$. That is, both schemes do not incur significant increase in resource usage in order to satisfy multiple demands. Second, compared to the peak-greedy, the M-greedy reduces the number of ITAPs by up to 20%. We observe similar results when the demands from active houses are homogeneous. These results suggest that M-greedy handles variable demands quite well, satisfying demands over multiple periods with only a marginal increase in the number of ITAPs.

8. CONCLUSIONS

In this paper we look at the problem of efficient ITAP placement to provide Internet connectivity in multi-hop wireless networks. There are a range of emerging applications, such as wireless neighborhood networks and sensor networks, which can benefit from our work. Table 1 summarizes our key results. As it shows, we make three major contributions in this paper.

First, we formulate the ITAP placement problem under various wireless models, and design algorithms for each model.

Second, we address several practical issues when using these algorithms. In particular, we extend the placement algorithms to

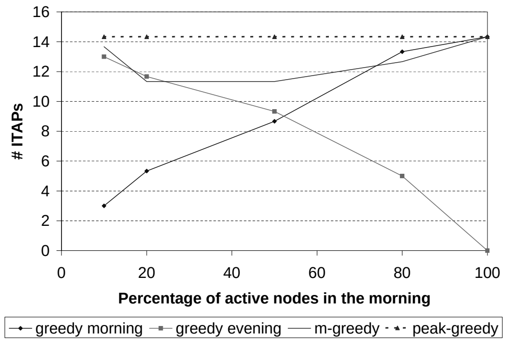

Figure 20: Ideal model: the performance under a diurnal demand pattern, where $N = 100$, $CR = 10$, $WC = 6$, $IC = 100$, demands from active houses follow a Zipf distribution.

	Ideal link	Bounded hop count	Smooth degradation
Throughput	$Thruput = 1$	$Thruput = 1$ for $hop \leq k$ and 0 otherwise	$thruput = \frac{1}{n}$
Approximated scenarios	Directional antennas with efficient spectral use		omni directional antenna
Greedy alg.	GreedyIdeal	GreedyBounded	GreedySmooth
Alternative alg.	Augment, clustering, random, lower bounds of all versions		
Provide fault tolerance	Using LP to determine the amount of demands satisfied with a given set of ITAPs		
Handle workload variation	Modify the greedy alg. to maximize $\sum_{t \in T} SD_t$ in each iteration, where SD_t is the total satisfied demand at time t		

Table 1: Summary of the algorithms introduced in this paper

provide fault tolerance and to handle variable user demands. These two enhancements improve robustness of our placement schemes in face of failures and demand changes.

Third, we demonstrate the efficiency of our placement algorithms using analysis and simulations, and show that the greedy algorithms give close to optimal solutions over a variety of scenarios we have considered.

To our knowledge this is the first paper that looks at the ITAP placement problem for general MAC schemes.

There are a number of avenues for future work. First, it is interesting to study placement algorithms under other wireless models (e.g., conflict-graph based interference model [16]). In addition, we are interested in exploring incremental versions of the placement algorithms that handle growing user subscriptions. Ideally we would like ITAPs to be placed not only to optimize for the current user subscriptions, but also to optimize for future user subscriptions. We think that the approach based on simultaneously optimizing ITAP placement for multiple scenarios, as used for dealing with variable workload, could be useful for developing incremental algorithms.

9. REFERENCES

[1] 802.11b community network list. http://www.toaster.net/wireless/community.html.

[2] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovi, editors. *Mobile Ad Hoc Networking*. Wiley-IEEE Press, Jul. 2004.

[3] Y. Bejerano. Efficient integration of multi-hop wireless and wired networks with QoS constraints. In *ACM MOBICOM*, Sep. 2002.

[4] D. A. Beyer, M. Vestrich, and J. J. Garcia-Luna-Aceyes. The rooftop community network: Free, High-Speed network access for communities. In *The First 100 Feet: Options for Internet and Broadband Access*. MIT Press: Cambridge, 1999.

[5] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and Zipf-link distributions: Evidence and implications. In *Proc. of IEEE INFOCOM 99*, Mar. 1999.

[6] M. Chesire, A. Wolman, G. M. Voelker, and H. M. Levy. Measurement and analysis of a streaming-media workload. In *Proc. of 3rd USENIX Symposium on Internet Technologies and Systems*, Mar. 2001.

[7] R. R. Choudhury, X. Yang, R. Ramanathan, and N. H. Vaidya. Using directional antennas for medium access control in ad hoc networks. In *Proc. of ACM MOBICOM*, Sept. 2002.

[8] Wireless networking reference - community wiress/rooftop systems. http://www.practically-networked.com/tools/wireless_articles_community.htm.

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms (Second Edition)*. MIT Press, Cambridge, 2001.

[10] CPLEX version 7.5.0. http://www.ilog.com/products/cplex/.

[11] T. ElBatt and B. Ryu. On the channel reservation schemes for ad-hoc networks: Utilizing directional antennas. In *5th IEEE Int'l Symposium on Wireless Personal Multimedia Communications*, 2002. http://www.wins.hrl.com/projects/adhoc/d/elbatt_wpmc02.pdf.

[12] M. Gerla, R. Bagrodia, L. Zhang, K. Tang, and L. Wang. TCP over wireless multi-hop protocols: Simulation and experiments. In *Proc. of 1999 IEE International Conference on Communications (ICC)*, Jun. 1999.

[13] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2), Mar. 2000.

[14] V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis. Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. In *Proc. of the 31st ACM Symposium on Theory of Computing (STOC)*, pages 19–28, 1999.

[15] G. Holland and N. Vaidya. Analysis of TCP performance over mobile ad hoc networks. In *Proc. of ACM Mobicom 99*, 1999.

[16] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proc. ACM MOBICOM*, Sept. 2003.

[17] S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. On the placement of Internet instrumentation. In *IEEE INFOCOM*, Mar. 2000.

[18] S. Jamin, C. Jin, A. Kurc, D. Raz, and Y. Shavitt. Constrained mirror placement on the Internet. In *Proc. of IEEE Infocom*, 2001.

[19] R. Karrer, A. Sabharwal, and E. Knightly. Enabling Large-scale Wireless Broadband: The Case for TAPs. In *2nd Workshop on Hot Topics in Networks (HotNets-II)*, 2003.

[20] D. Kotz and K. Essien. Analysis of a campus-wide wireless network. In *Proc. of ACM MOBICOM*, Sept. 2002.

[21] A. Kuehn and M. Hamburger. A heuristic program for locating warehouses. *Management Science*, 9:643–666, 1963.

[22] B. Li et al. On the optimal placement of Web proxies in the Internet. In *Proc. of IEEE INFOCOM*, 1999.

[23] L. M. S. C. of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Standard 802.11*, 1999.

[24] L. Qiu, R. Chandra, K. Jain, and M. Mahdian. On the placement of integration points in multi-hop wireless networks. In *MSR Technical Report MSR-TR-2004-43*, 2004. http://www.research.microsoft.com/~liliq/papers/pub/msrtr-placement2004.ps.

[25] L. Qiu, V. N. Padmanabhan, and G. Voelker. On the placement of Web server replicas. In *Proc. of IEEE Infocom*, 2001.

[26] Qualnet. http://www.scalable-networks.com/.

[27] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy. An analysis of Internet content delivery systems. In *Proc. of the 5th Symposium on Operating Systems Design and Implementation (OSDI'2002)*, Dec. 2002.

[28] J. L. Sobrinho and A. S. Krishnakumar. Quality of service in ad hoc carier sense multiple access wireless networks. In *JSAC*, 1999.

[29] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.