SMU Journal of Undergraduate Research

Volume 1 Article 5

2017

A Dynamic Hierarchical Network Topology to Reduce Interference in User-Rich LANs

Ian Johnson
Southern Methodist University, ianjjohnson@smu.edu

Erik Gabrielsen
Southern Methodist University, egabrielsen@smu.edu

Danh Nguyen
Southern Methodist University, danhn@smu.edu

Gavin Pham
Southern Methodist University, gpham@smu.edu

Alex Saladna
Southern Methodist University, asaladna@smu.edu

See next page for additional authors

Recommended Citation

Johnson, Ian; Gabrielsen, Erik; Nguyen, Danh; Pham, Gavin; Saladna, Alex; and Siems, Travis (2017) "A Dynamic Hierarchical Network Topology to Reduce Interference in User-Rich LANs," SMU Journal of Undergraduate Research: Vol. 3 , Article 5. DOI: https://doi.org/10.25172/jour.3.1.5

A Dynamic Hierarchical Network Topology to Reduce Interference in User-Rich LANs

Authors

Ian Johnson, Erik Gabrielsen, Danh Nguyen, Gavin Pham, Alex Saladna, and Travis Siems

Maximizing Throughput in Wireless LANs Using Hierarchical Network Topologies

Ian Johnson
ianj@smu.edu

Erik Gabrielsen
egabrielsen@smu.edu

Danh Nguyen
danhn@smu.edu

Gavin Pham
gpham@smu.edu

Alex Saladna
asaladna@smu.edu

Travis Siems
tsiems@smu.edu

ABSTRACT

In this paper we present a greedy approach to create hierarchical network topologies for throughput optimization in single access point to create hierarchical network topologies. By minimizing electromagnetic interference, we optimize throughput by creating topologies where the probability of a collision occurring is low. We evaluate a series of greedy topology algorithms based on the average through- put of the resulting network. We conclude that hierarchical network topologies generated with greedy algorithms significantly outperform networks with simple star topologies by up to 75%.

1. INTRODUCTION

Local Area Networks (LANs), operating over a wireless medium using variants of the 802.11 protocol, provide users the ability to connect and disconnect from a network at any time. This functionality is typically achieved by LANs connecting each new user to a statically assigned physical access point (AP), which routes the user's traffic to the rest of the network. 802.11 LANs often consist of hundreds of users who share the wireless link. Sharing is accomplished with Frequency-division Multiplexing (FDM) plus Time-division Multiplexing (TDM) using slotted Aloha with CSMA/CA. The 802.11 MAC sublayer is designed with two modes of communication: Distributed Coordination Function (DCF), where each station acts independently, and Point Coordination Function (PCF), where an AP controls all activity in its cell. DCF and PCF both use slotted Aloha with CSMA/CA to avoid collisions by waiting until the channel is free before sending a frame. After a frame is sent, a quick acknowledgement is sent back to the sender and communication continues. If no acknowledgement is sent, the frame is re- sent after an exponential back off. This approach is efficient in distributing network resources among nodes in a low or medium traffic density network, but deteriorates rapidly in a high host/traffic density LAN, partially due to interference. In all existing MAC algorithms and static multiplexing strategies such as FDMA and TDM, successful frame throughput declines significantly as host and traffic density increase on the network.

An approach to maximize throughput for user-rich LANs is topology optimization. Using a hierarchical topology, traffic is routed through subAPs to a root AP, thereby isolating small subnetworks, each of which function more efficiently due to their smaller size. The resulting network topology forms a hierarchy and improves the throughput of a net- work. This is often implemented in practice using multiple physical APs within the geographic span of a LAN. How- ever, this approach statically allocates network resources, which causes it to suffer the same fate as FDMA when un- even network activity distributions occur, and as a result, good throughput suffers. A dynamic approach with multiple physical APs is not realistic, as it would require that APs physically move around the network in response to varying network activity.

In order to overcome low throughput in high traffic wire- less LANs, we present a method of routing host traffic through subAPs to and from a root AP. By using a simulation to test various topology algorithms, we compare the interference and throughput for various network topologies.

In organizing a set of subnetworks, a number of considerations must be taken into account. The typical wireless LAN uses a star topology, in which all hosts connect directly to a physical AP. We hereby refer to one of such algorithm as the "null algorithm." A "superior" algorithm must outperform the star topology in most, if not all, network viability metrics for a variety of possible network distributions.

A typical LAN may be geographically organized in a nor- mal or pseudo-random distribution, or a clustered distribution. In a pseudo-random distribution, hosts have no tendency to gravitate toward one another, thus interference differs depending on the configuration of the LAN. In a clustered distribution, hosts tend to group together, and clusters can either be geographically fixed or mobile. In a geographically fixed clustered distribution, users gravitate around a physical landmark within the geographic span of a LAN. An example of such a landmark would be the seating arrangement at an airport gate, where users tend to aggregate. In a dynamically clustered distribution, the geographic location of a cluster is subject to

change over time. An example of such a distribution would be a large conference room, wherein people move around in groups while each of their phones continuously talk to an AP. These clustered distributions greatly increases the amount of interference and lowers the throughput of the network as a whole. To solve the problem of interference, we will treat the two scenarios as one agglomerate 'clustered' distribution. The rest of our paper is organized as follows. First, we explore existing approaches to throughput optimization via multiplexing, MAC algorithms, and topology optimization. We then expand on the idea of topology optimization by de- scribing a series of greedy topology algorithms and testing them to build topologies for both clustered and random net- work layouts. Finally, we identify the strengths and weak- nesses of each algorithm based on their performance metrics.

2. RELATED WORK

There exists substantial research into various network topologies for Wireless Sensor Networks. There is no fixed infrastructure in such networks; devices are mobile and routes can break [15]. Furthermore, a network must be flexible since connectivity amongst nodes may vary with time due to nodes departing [8]. As such, AP selection algorithms can place nodes in connections of sub-optimal bandwidth allocion [14]. Interference is an issue faced in many networks and is reduced using various methods like the ALOHA algorithm and CSMA, CD/CA and BEB[10, 6]. However, traffic can greatly affect their performance especially when dealing with OFDM (Orthogonal Frequency Division Multiplexing) and its ability to allow more users access to the subdivided band- width [11]. As Kaynia and Jindal have mentioned - "for lower densities, CSMA with transmitter-sensing actually performs worse than ALOHA, having about 10(percent) more out- age probability" [10]. In Blaszczyszyn's research it is shown that, as network load increases, these protocols will reach a maximum threshold of performance and begin to lose efficiency[2]. For this reason we cannot rely on these protocols to manage ideal throughput.

There are various algorithms for topology calculation, but among the most notable is Prim's algorithm. This algorithm starts at a random node on the graph and examines all avail- able edges in order to choose the node with the smallest cost [12]. However, it is also important to look at the network topology that is formed. There is extensive research into the type of topology formed and the benefits that can be gained [1, 3]. One example is a cellular connection where the nodes are broken off into four groups, and each group is connected to its opposite and adjacent groups by only one connection. Airoldi's research states that different connection orientations of equally spaced nodes can greatly alter the overall performance [1]. However in Prim's, there is no hierarchical structure as the 4 groups are equal in relation to one another. Our network will contain one centralized AP, with a multitude of hosts connected to this single AP in a hierarchical manner. We find that this greatly increases throughput performance.

Another commonly known algorithm is Kruskal's algorithm for finding the minimum spanning tree. Kruskal's algorithm works by first finding the shortest path between each node, then linking up subtrees by the shortest path, avoiding cyclic paths. [5] Like Prim's, Kruskal's algorithm works well with grouped, non-hierarchical networks, but does not provide a solution to hierarchical networks with one AP [7]. Kruskal's and Prim's also are greedy in which paths they take based on the weight or distance from one node to the next. Forms of topology control exist to help and are used to modify different network routes in a more efficient manner [4]. These two parameters are often inseparable and are important to take into consideration [13]. This is problematic in that they both do not attempt to maximize the throughput of a network, only find shortest paths [9]. These work well for Ethernet LANs, but not for wireless protocols. Because each of these algorithms act upon net- works with no clear infrastructure, we have outlined contraints that will prevent ambiguous networks and allow for us to properly compare several different greedy algorithms in a concrete manner. It is important to understand both of these algorithms when implementing our network topology and will be useful in establishing network paths.

3. TOPOLOGY VIABILITY MODEL

We model a LAN topology as a set of possible geographic host distributions, ranging from randomized distributions to heavily clustered distributions. The model also considers a range of average host densities per geographic area, discretized into three categories: low, medium, and high- density. While our focus lies in the performance of LAN topology algorithms in high-density (user-rich) LANs, a viable algorithm must also provide competitive functionality in low and medium-density host distributions.

Interfering node count
The number of nodes whose broadcasts interfere with the communications of any given node.
Number of re-broadcasts
The number of nodes through which the communications of any given node are routed before reaching the AP.
Signal distance to AP
The total distance over which signals from a host travel before eventually reaching the AP.
Total network traffic
The total amount of communication occurring over the network at any given time. Calculated as the sum of the transmission rate of each node multiplied by the number of re-broadcasts that must occur to connect a host to the AP.
Throughput
Throughput, t is the function of the mean number of interfering nodes, n, and the number of re-broadcasts, b, such that $t = \frac{1}{n*(b+1)}$. This is derived from two facts: the ability of a node to transmit traffic is inversely proportional to the number of interfering nodes, and the rate at which data gets from a host to the AP is inversely proportional to the number of nodes through which data must be routed.

Table 1: Topology Viability Model Assumptions

Our model operates under a set of assumptions that streamline the process of assessing topology algorithms.

Table 2: Topology Performance Metrics used in our Model

Constant Transmission Rates
All hosts send data across the network at a constant rate. In practice, this rate would be the average data transmission rate of a host.
Single access point
All hosts connect, directly or indirectly, to a single root AP. Because each algorithm tested (excluding the null algorithm) generates a hierarchical topology, it is assumed that each algorithm could be reasonably extended to a multi-AP system.
Mutable NIC Broadcast Range
All NICs can change the power/range with which they transmit a broadcast signal. The lack of this assumption precludes any significant interference minimization.
Hosts within range of AP
All hosts on the LAN are within range of the AP such that each host can broadcast a signal with sufficient strength to reach the AP and the AP, in turn, can broadcast a signal with sufficient strength to reach each host.

Based on this set of assumptions, our model measures the viability of an algorithm for any given environment based on the topology performance metrics laid out in Table 2.

Each of these metrics is calculated for each proposed algorithm in 9 possible host layout scenarios. Interference, hops, and throughput are shown in a table for each algorithm. Throughput is then calculated for each algorithm/layout pair and compared in the results summary section.

A star topology algorithm is used as a null algorithm, providing a basis for comparison of topology algorithms. In a star topology, each host on the network connects directly to the AP. For each topology algorithm tested, the algorithm is used to generate a topology for 1000 networks from each possible combination of semi-clustered, very-clustered, and unclustered (random) host distributions with low, medium, and high host densities.

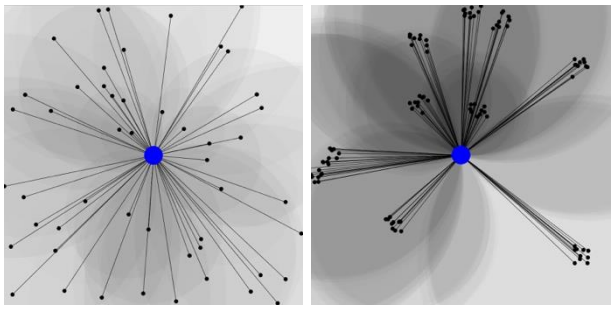


Figure 1: Two network topologies generated using the null algorithm for a clustered and geographically random net- work. The high density of grayness across the network indi- cates significant signal interference. The circles around each node indicate the range of interference for that node.

Table 3: Viability metrics for the null algorithm

Layout	Interference	Hops	Throughput
R:L	9.033 ±3.919	0.0±0.0	0.110
R:M	30.16 ±9.434	0.0±0.0	0.033
R:H	72.65 ±26.95	0.0±0.0	0.013
SC:L	13.96 ± 4.362	0.0±0.0	0.071
SC:M	32.6 ± 15.09	0.0±0.0	0.030
SC:H	75.63 ± 31.96	0.0±0.0	0.013
C:L	19.2 ± 7.263	0.0±0.0	0.052
C:M	40.99 ±18.58	0.0±0.0	0.024
C:H	85.30 ±29.38	0.0±0.0	0.011

Table 3 shows the viability metrics for the null algorithm. The 'Layout' column refers to the geographic distribution of hosts in the form *Layout*: *Density*, where *layout* is *R* for random, *SC* for semi-clustered and *C* for clustered, and *density* describes the geographic host density as *L* for low, *M* for medium, and *H* for high-density. The throughput metric will be used to compare the relative performance of the algorithms.

Figure 1 shows an example of a star topology in our network model. The grayed-out area represents

geographic regions where interference is occurring. Darker shades of gray indicate greater interference.

4. TOPOLOGY ALGORITHMS

In this section, we present a number of greedy algorithms for hierarchical topology generation. We use each algorithm to generate topologies and then evaluate the throughput of the resulting topologies to evaluate the relative efficiency of the algorithms.

4.1 Proximity Greedy Algorithm

Our first approach to generating an optimized network topology promotes nodes geographically proximate to the AP into sub-APs. To do so, we implemented a greedy algorithm that assigns the nearest unassigned host to become a sub-AP, and assigns the *n* nodes closest to the new sub-AP to be its children. Here, *n* is a load factor calculated as the total number of hosts on the network divided by the desired number of sub-APs. This approach attempts an even distribution of re-transmission loads among the hosts assigned to be sub-APs.

4.1.1 Simple Proximity Greedy Algorithm

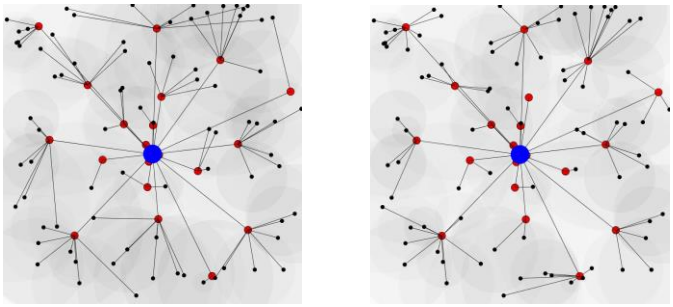
An un-optimized, single-iteration of the proximity greedy algorithm is the most basic approach to topology optimization. This algorithm simply assigns sub-APs as those hosts nearest to the AP.

Table 4: Viability metrics for the basic proximity-greedy algorithm

Layout	Interference	Hops	Throughput
R:L	4.4 ± 3.693	0.666 ± 0.471	0.136
R:M	9.25 ± 8.536	0.84 ± 0.366	0.058
R:H	16.03 ± 22.80	0.856 ± 0.351	0.033
SC:L	7.1 ± 4.7	0.8 ± 0.4	0.078
SC:M	10.65 ±11.68	0.82 ± 0.384	0.051
SC:H	18.51 ±25.67	0.856 ± 0.351	0.029
C:L	10.83 ±6.812	0.833 ± 0.372	0.050
C:M	15.24 ±18.16	0.85 ± 0.357	0.035
C:H	21.44 ±28.25	0.868 ± 0.338	0.024

4.1.2 Post-Configuration Optimization

In post-network configuration, the greedy algorithm can be optimized by having each non-subAP host switch to a new subAP, if the host is closer to a subAP different from its own. This slightly deteriorates the even distribution of traffic routed through each subAP, which exists in the basic proximity greedy algorithm. The topologies generated by this algorithm had the highest throughput of the algorithms tested for most layouts.



(a) Un-Optimized (b) Optimized

Figure 3: (a) shows an un-optimized topology generated using a proximity greedy algorithm. (b) shows the same topology after post-configuration optimizations. Network topologies generated with the optimized algorithm typically have 20% higher throughput than those generated with the basic algorithm.

Algorithm 1 Optimized Proximity Greedy Algorithm

Require: *hosts* is a list of all hosts on the network, *subAPs* is an empty list of subAPs, and *rootAP* is the physical AP of the network

procedure buildNetwork(*hosts, subAPs, rootAP*)
  sort hosts by distance to rootAP
  for *i* ← 1 to *k* **do**
    subAP s[*i*] ← *host*[*i*]
  for *i* *k* to *numHosts* **do**
    sort *subAP s* by distance to *host*[*i*]
    host[*i*].*myAP* ← *subAP s*[0]

Figure 2: Pseudocode for the optimized proximity greedy algorithm

Table 5: Viability metrics for the optimized proximity- greedy algorithm

Layout	Interference	Hops	Throughput
R:L	3.933 ± 3.678	0.666 ± 0.471	0.152
R:M	7.82 ± 8.464	0.84 ± 0.366	0.069
R:H	12.84 ± 23.25	0.856 ± 0.351	0.041
SC:L	5.433 ± 4.923	0.8 ± 0.4	0.102
SC:M	8.67 ± 11.86	0.82 ± 0.384	0.063
SC:H	14.13 ± 25.62	0.856 ± 0.351	0.038
C:L	7.166 ± 6.039	0.833 ± 0.372	0.076
C:M	9.75 ± 14.12	0.85 ± 0.357	0.055
C:H	15.93 ± 28.15	0.868 ± 0.338	0.033

Figure 4 shows a comparison of the two algorithms for a geographically randomly distributed network. The grayed- out area represents geographic regions where interference is occurring. Darker shades of gray indicate greater interference.

4.2 Recursive Proximity Greedy Algorithm

As an expansion of the basic proximity greedy algorithm, two recursive versions exist. After assigning sub-APs, each sub-AP uses the same

proximity greedy algorithm used by the AP to assign their own sub-subAPs. A semi-recursive version restricts the network to generating a limited number of layers to the AP hierarchy. A fully recursive version generates a tree of sub-APs until it reaches edge nodes on the LAN. Both versions are implemented.

Figure 5: Pseudocode for the recursive proximity greedy algorithm

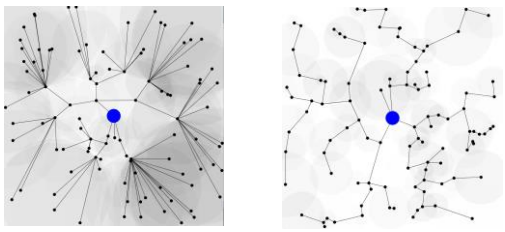
Algorithm 2 Recursive Proximity Greedy Algorithm

Require: *hosts* is a list of all hosts on the network, *subAPs* is an empty list of subAPs, and *rootAP* is the physical AP of the network

```
procedure buildNetwork(hosts, subAP s, rootAP )
    if length(hosts) <= 1 then
        return
    sort hosts by distance to rootAP
    for i ← 1 to k do
        subAP s[i] ← hosts[i]
    for i ← k to numHosts do
        sort subAP s by distance to hosts[i]
        hosts[i].myAP ← subAP s[0]
        subAPs[0].myChildren.append(hosts[i])
    for subAP in subAP s do
        buildNetwork(subAP.myChildren, [], subAP )
```

4.2.1 *Semi-Recursive Proximity Greedy Algorithm*

In a semi-recursive approach, the proximity greedy algorithm recurses up to k times to define subnetworks, sub- subnetworks, and so on. k is calculated as a function of the total number of nodes on the network, such that the algorithm recurses with depth relative to the density of the network. A viable, but less robust solution would be to sim- ply recurse n times, where n is some constant hard-coded in the algorithm. However, this approach fails to properly scale to the total number of hosts on the network.



(a) Semi-Recursive Proximity Greedy Algorithm
(b) Fully-Recursive Proximity Greedy Algorithm

Figure 4: (a) shows a topology generated by a semi-recursive proximity greedy algorithm. This limits the number of hops an edge node would have to take to reach the AP. (b) shows a topology generated by a fully-recursive distance greedy algorithm.

4.2.2 *Fully-Recursive Proximity Greedy Algorithm*

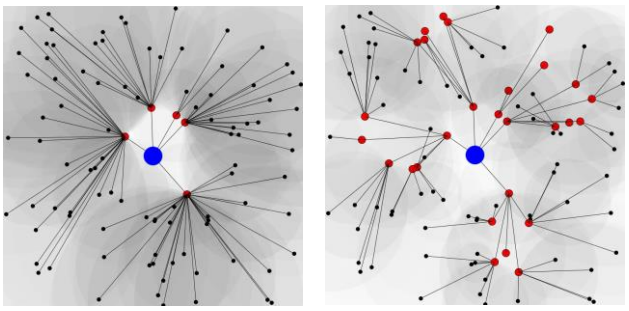
In a fully-recursive greedy approach, the proximity algorithm recurses indefinitely until it reaches edge nodes on each branch of recursion. This approach attempts to minimize total interference, not taking into account additional overhead introduced by having numerous rebroadcasts. This algorithm produced topologies with very low interference, but low throughput due to the high hop-count introduced by the algorithm.

Table 6: Viability metrics for the fully-recursive proximity- greedy algorithm

Layout	Interference	Hops	Throughput
R:L	1.766 ± 1.308	2.933 ± 1.711	0.143
R:M	2.07 ± 1.484	5.38 ± 2.481	0.075
R:H	2.032 ± 1.367	10.88 ± 5.088	0.041
SC:L	3.366 ± 3.506	4.866 ± 3.603	0.050
SC:M	2.6 ± 2.894	9.45 ± 5.237	0.036
SC:H	2.364 ± 2.766	13.19 ± 6.161	0.029
C:L	3.866 ± 4.462	3.5 ± 2.334	0.057
C:M	3.08 ± 4.318	8.18 ± 4.387	0.035
C:H	3.556 ± 6.443	14.15 ± 7.905	0.018

Table 7: Viability metrics for the semi-recursive proximity greedy algorithm

Layout	Interference	Hops	Throughput
R:L	2.333 ± 1.349	2.0 ± 1.064	0.142
R:M	9.54 ± 6.632	2.71 ± 0.725	0.028
R:H	40.41 ± 23.62	2.892 ± 0.473	0.006
SC:L	4.633 ±3.525	2.2 ± 1.077	0.067
SC:M	19.08 ±12.23	2.77 ± 0.690	0.013
SC:H	49.14 ±28.57	2.9 ± 0.458	0.005
C:L	7.133 ±4.356	2.266 ± 1.093	0.042
C:M	16.78 ±10.68	2.78 ± 0.686	0.015
C:H	42.68 ±22.83	2.892 ± 0.473	0.006



(a) Single-tier centralization centralization (b) Multi-tier

Figure 6: Topologies generated by single and multi-tier centralization greedy algorithms

4.3 Centralization Greedy Algorithm

In order to optimize for areas of high host density, which are isolated from the location of the AP, another greedy algorithm selects sub-APs based on

hosts' centralization relative to all other hosts on the network. Centralization is defined as the average distance to each other host on the network from any given node. This approach is implemented as a single-tier hierarchy as well as a recursive multi-tier hierarchy.

4.3.1 Single-Tier Centralization

In a single-tier centralization approach, a single iteration of the centralization greedy algorithm assigns k sub-APs based on highest overall proximity. Here, k is a function of the total number of hosts on the network.

Table 8: Viability metrics for the single-tier centralization greedy algorithm

Layout	Interference	Hops	Throughput
R:L	4.933 ±2.112	0.833 ± 0.372	0.110
R:M	19.25 ±8.565	0.95 ± 0.217	0.026
R:H	63.30 ±25.70	0.98 ± 0.14	0.007
SC:L	9.9 ± 3.708	0.833 ± 0.372	0.055
SC:M	25.2 ± 11.04	0.95 ± 0.217	0.020
SC:H	65.52 ± 27.00	0.98 ± 0.14	0.007
C:L	11.46 ±7.387	0.833 ± 0.372	0.047
C:M	24.52 ±13.75	0.95 ± 0.217	0.020
C:H	60.32 ±28.50	0.98 ± 0.14	0.008

4.3.2 Multi-Tier Centralization

In a multi-tier centralization approach, the centralization greedy algorithm recurses up to $\log_2 k$, and on layer n of recursion. Here, k is a function of the total number of hosts on the network. The algorithm assigns $\frac{k}{2^n}$ subAPs, where $n = 0$ at the first layer of recursion.

Table 9: Viability metrics for the recursive centralization greedy algorithm

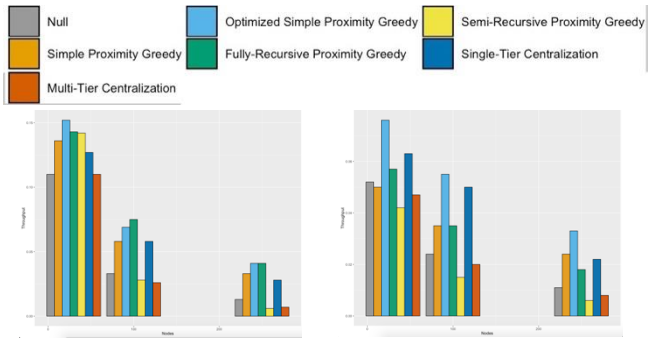
Layout	Interference	Hops	Throughput
R:L	3.866 ± 2.348	1.033 ± 0.604	0.127
R:M	6.44 ± 7.521	1.65 ± 0.572	0.058
R:H	13.49 ± 16.47	1.604 ± 0.572	0.028
SC:L	6.166 ± 4.913	1.033 ± 0.795	0.079
SC:M	8.08 ± 10.76	1.36±0.671	0.052
SC:H	13.38 ± 18.29	1.68±0.574	0.027
C:L	8.6 ± 8.920	0.833 ± 0.687	0.063
C:M	9.01 ± 11.90	1.2 ± 0.632	0.050
C:H	15.42 ± 14.91	1.872 ± 0.389	0.022

4.4 Algorithm Comparison

Table 10 shows the average throughput from the nine possible layouts for each algorithm. Figure 7 summarizes the performance of the algorithms in randomized and clustered network configurations, respectively. Each bar represents the throughput of an algorithm (higher is better).

Table 10: Average throughput for all algorithms

Algorithm	Average Throughput
Null	0.040
Simple Proximity Greedy	0.055
Opt. Simple Proximity Greedy	0.070
Fully-Recursive Proximity Greedy	0.054
Semi-Recursive Proximity Greedy	0.036
Single-Tier Centralization	0.056
Multi-Tier Centralization	0.033



(a) Random Layout Layout (b) Clustered

Figure 7: Throughput results for all algorithms

The highest-performing algorithm was the optimized proximity greedy algorithm, which improved upon the null topology by as much as 76%.

5. ANALYSIS

Analysis of throughput for all tested network layouts leads to two clear results. The first is that hierarchical topologies can improve upon the throughput of the basic star algorithm by up to 76%. This corroborates the existing theory that hierarchical topologies are more efficient than star topologies. The second is that the optimized proximity greedy algorithm is the best algorithm of those we tested. In each of the 9 scenarios, this algorithm outperformed every other algorithm tested. This algorithm performed competitively in random host distributions and produced exceptional results for semi-clustered and clustered distributions.

Recursive hierarchical topology algorithms that were tested produced topologies with very little interference compared to single-tier hierarchies. The semi and fully-recursive proximity greed algorithms had the lowest average interfering node counts of any algorithm. However, these algorithms have higher average re-transmission counts, and therefore did not outperform single-tier hierarchies for throughput.

In clustered layouts, the throughput performance of the algorithms varies much more than in random layouts. In random layouts, all algorithms tend to generate topologies whose throughputs are within 20% of each other. In clustered layouts,

however, the deviation was much larger. In a medium clustered layout, for example, the optimized simple proximity greedy algorithm performed more than three times better than the semi-recursive proximity greedy algorithm in terms of throughput of resulting topology.

6. CONCLUSIONS

Our research finds that hierarchical network topologies generated with greedy algorithms outperform star topologies for wireless LANs which follow our model's assumptions. Therefore, we conclude that, in theory, a hierarchical topology-based MAC algorithm could improve throughput in wireless LANs over the current 802.11 standard. Hierarchical topologies are shown to outperform the star topology used in 802.11 so, provided that the greedy algorithms could be efficiently implemented in a distributed setting, a standard which uses such algorithms could optimize 802.11.

We also conclude that, in general, recursively assigning sub-subAPs decreases throughput instead of optimizing it. While recursive greedy algorithms minimize interference, they introduce a large number of re-transmissions which decrease overall throughput. However, due to low localized interference in topologies generated by these algorithms, they may be useful for mesh networks or low-energy sensor networks.

7. REFERENCES

[1] Edoardo Airoldi and Kathleen Carley. Sampling algorithms for pure network topologies.

[2] P. Muhlethaler B. Blaszczyszyn and S. Banaouas. *Wireless Ad-Hoc Networks*, volume 1. InTech, 2012.

[3] Suman Banerjee, Archan Misra, Jihwang Yeo, and Ashok Agrawala. Energy-efficient broadcast and multicast trees for reliable wireless communication. In *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, volume 1, IEEE, 2003.

[4] Stefano Basagni, Alessio Carosi, and Chiara Petrioli. Sensor-dmac: Dynamic topology control for wireless sensor networks. Technical report, Northeastern University, The address of the publisher, 7 2004.

[5] Arindam K Das, Robert J Marks, Mohamed El-Sharkawi, Payman Arabshahi, and Andrew Gray. Optimization methods for minimum power bidirectional topology construction in wireless networks with sectored antennas. In *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, volume 6, IEEE, 2004.

[6] S. Kumar E. Munapo, B.C. Jones. A minimum incoming weight label method and its application in cpm networks. *ORiON*, 2007.

[7] Kevin Fuchs. A probabilistic approach to discover heterogeneous network topologies.

[8] Idress Skloul Ibrahim. *An Optimum Routing Algorithm for Ad Hoc Networks*. PhD thesis, Heriot Watt University, 3 2007. Mountbatten, Room G28.

[9] Ayad Ghany Ismaeel. Effective technique for allocating servers to support cloud using gps and gis. In *Science and Information Conference (SAI), 2013*, IEEE, 2013.

[10] M Kaynia and N Jindal. Performance of aloha and csma in spatially distributed wireless networks.

[11] Brijesh Kumar, Sumit Jindal, and Prakash Dua. *A REVIEW PAPER ON ORTHOGONAL FREQUENCY DIVISION MULTIPLEXING (OFDM)*. PhD thesis, 2 2015.

[12] Artur Mariano, Dongwook Lee, Andreas Gerstlauer, and Derek Chiou. Hardware and software implementations of prim's algorithm for efficient minimum spanning tree computation. 2013.

[13] Seah Tan. Dynamic topology control to reduce interference in magnets, 2005.

[14] Diot Kurose Towsley Vasudevan, Papagiannaki. Facilitating access point selection in ieee 802.11 wireless networks, 2005.

[15] Pedro Whiteman and Miguel Labrador. Topology control in wireless sensor networks, 2009. and Atarraya, a Simulation Tool to teach and Research Topology Control Algorithms for Wireless Sensor Networks.