Energy Efficient Data Gathering in Multi-Hop Hierarchical Wireless Ad Hoc Networks

Liron Levin   Michael Segal
Dept. of Communication Systems Engineering
Ben-Gurion University of the Negev
Beer-Sheva, 84105, Israel
levinlir@gmail.com, segal@cse.bgu.ac.il

Hanan Shpungin
Dept. of Computer Science
University of Calgary
Calgary, AB, T2N 1N4, Canada
hanan.shpungin@ucalgary.ca

ABSTRACT
This paper studies the problem of data gathering in hierarchical wireless ad hoc networks. In this scenario, a set of wireless devices generates messages which are addressed to the base station. As not all nodes can reach the base station through a direct transmission, messages are relayed by other devices in a multi-hop fashion. We consider data gathering without aggregation, i.e. all the generated messages are required to reach the base station – this is in contrast to the well studied problem of data gathering with aggregation, which appears to be significantly simpler.

The above scheme may have poor performance in wireless networks with hierarchical architecture. The devices in the layer closest to the base station form a bottleneck in terms of energy consumption as they experience a very high volume of forward requests. Eventually, these nodes will be the first to run out of their battery charges which will cause connectivity losses. In this paper we focus on prolonging the network lifetime of hierarchical networks through efficient balancing of forward requests, which is NP-hard. We develop an approximation scheme which produces a data gathering tree with network lifetime which is at most \( k \) times less than the optimal one, where \( k \) is the number of hierarchical layers. Our results are analytically proved and validated through simulations.

Categories and Subject Descriptors
C.2.2 [Computer-Communication Networks]: Network Architecture and Design—Network topology, Wireless communication; G.2.2 [Discrete Mathematics]: Graph Theory—Graph algorithms, Network problems

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

\( \text{FOMC'11, June 9, 2011, San Jose, California, USA} \)
Copyright 2011 ACM 978-1-4503-0779-6/11/06 ...$10.00.

1. INTRODUCTION
Multi-hop ad hoc wireless networks have become an integral part of many of today’s applications, such as mobile communication, radio broadcasting, and sensor monitoring. An obvious advantage of these networks is that wireless nodes can be easily deployed even in hostile environments or rough terrain. Once deployed, the devices cooperate to establish a communication backbone based on their relative proximity, transmission ranges, obstacles, and energy reserves. As wireless networks are much harder (or even impossible) to maintain, the choice of the communication backbone has a critical impact on the survivability and QoS of the network.

The complexity of multi-hop wireless ad hoc networks presents numerous scalability issues which appear in the
form of (1) wireless interference due to spatial concurrency, (2) limited and unreplenishable battery charges, and (3) heavy routing overhead due to node failure and constant changes in channel states. To overcome some of these issues, it is common to form a hierarchical network topology [3, 5, 7, 14, 27], where the wireless nodes are logically divided into $k$ processing (active) layers, $1 \ldots k$ such that data can be exchanged only between nodes which reside in two adjacent layers (see Figure 1); moreover, a node in layer $i$ communicates only with a single node in layer $i - 1$, $2 \leq i \leq k$.

In this paper we explore the well-known data gathering without aggregation network scheme [8], where each node generates messages to be sent to a base station (also referred to as the root node). As most devices typically cannot reach the base station with a direct transmission (single hop), they are propagated in a multi-hop fashion over a data gathering tree using other nodes as relays [6, 25, 26]. Note that all messages initiated by the wireless devices must eventually reach the base station; this is contrary to a variation of the scheme, when data aggregation at intermediate nodes in the data gathering tree is allowed [9, 21].

Data gathering without aggregation can cause a very high energy consumption in some of the nodes, namely nodes which are the direct descendants of the base station in the data gathering tree, since these nodes are responsible for relaying all the messages which are generated in the network and thus consume most of the energy. This phenomenon is especially severe in hierarchical networks where the nodes in the second layer have the highest number of transmissions, which results in a rapid depletion of the battery charges and network lifetime decrease.\(^1\) As wireless devices are usually deployed in hard-to-reach areas, battery replenishment is impractical or even impossible, which makes the issue of energy efficiency critical for successful network operation.

Energy efficient data gathering has been extensively studied in the past [1, 15–17, 26, 28]. Surprisingly, data gathering with aggregation is considerably easier to solve under the energy efficiency objective [9, 19, 20] than its unaggregated counterpart [1, 15, 17, 23]. One of the possible reasons for the difference may lie in the fact that once data is allowed to be aggregated, the efficiency of the solution is dictated by the chosen communication links since every node transmits only once. Thus, the problem can be usually reduced to a simpler problem in graph theory [19, 20] and solved by using some of the existing methods. Once data aggregation is not allowed, one has to take the amount of data into account as well, since nodes are required to make multiple transmissions over the same communication links.

Our main contribution in this paper is the study of energy efficient data gathering problem in hierarchical networks, where each node $v$ generates an arbitrary number of messages $q(v)$. We seek to minimize the maximum number of messages relayed by a single node and thus increase the network lifetime, which is NP-hard [13]. We develop an efficient and elegant solution with a provable approximation ratio of $k$, where $k$ is the number of layers in the network. To the best of our knowledge this is the first optimization result for data gathering without aggregation in multi-hop hierarchical wireless ad hoc networks.

The rest of this paper is organized as follows. In Section 2 we describe our network model and present a formal problem definition. Existing works are surveyed in Section 3. Then, in Section 4 we develop and analyze our approximation algorithm, followed by Section 5 which holds the simulation results. Finally, we conclude the work and discuss future research directions in Section 6.

2. PROBLEM FORMULATION

In this section, we first define the hierarchical network model used in this paper and then present a formal problem definition.

2.1 Hierarchical network model

The network consists of $n$ wireless devices (nodes), $V$ and a base station (root node) $r$. The nodes are logically divided into $k$ disjoint processing layers $V_1, \ldots, V_k$ such that $\bigcup_{i=1}^{k} V_i = V$. For convenience of notation let $V_0 = \{r\}$. Let $G = (V \cup \{r\}, E)$ be the transmission possibilities graph, where $(u, v) \in E$ iff there exists $i$, $0 \leq i < k$ such that $u \in V_i$ and $v \in V_i$. Note that the edge set $E$ is determined based on multiple factors, such as the relative disposition of the wireless nodes, their transmission ranges, layer association, presence of obstacles in the deployment area, etc. We refer to $G$ as the $k$-layered input graph or simply input graph if $k$ is clear from the context.

The data gathering process is executed in discrete rounds. In every round each node $u$, has $q(u)$ messages to send to the root node $r$. The messages are propagated towards the root node in a multi-hop fashion by using a data gathering tree $T = (V \cup \{r\}, E_T)$ (subtree of $G$), where all the edges point towards the root $r$, i.e. $T$ is a reversed arborescence rooted at $r$. The same tree $T$ is used for all the rounds.

For any $u \in V$, let $T(u) = (V(u), E_{T(u)})$ be the subtree rooted at $u$ such that all the nodes in $T(u)$ pass their messages through $u$. Eventually, in every round, every node $u \in V_i$, $1 \leq i \leq k$, forwards $cn(T, u) = \sum_{v \in V(u)} q(v)$ messages to its ancestor in $V_{i-1}$. We refer to $cn(T, u)$ as the congestion of $u$. Note that according to this model, the non-leaf nodes of the tree forward all the messages which originate in their respective subtrees.

Let $\sigma(u)$ be the transmission power used by node $u \in V$. We assume that all the nodes $V_i$ in the $i$-th layer, $1 \leq i \leq k$, share the same transmission power $\sigma_1$ (also referred to as the cost) and that $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_k$ (this is a valid assumption as typical network architectures use weaker and smaller devices on the lower hierarchical levels). We define the cost of a node $u \in V$, $C(T, u)$, as the total energy consumed by $u$ in a single round as a result of transmitting $cn(T, u)$ messages, i.e. $C(T, u) = cn(T, u) \cdot \sigma(u)$.

Each node $u \in V$ has an initial battery charge of $b(u)$, which is reduced by $\sigma(u)$ after a transmission of a single message. Recall that the same tree $T$ is used in all rounds and the lifetime of a node, $l(u)$ is defined as the number of complete rounds in which it can participate, i.e. $l(u) = \lfloor b(u)/C(T, u) \rfloor$ [11, 18]. The network lifetime, $l(T)$ is defined as the last round in which every node can successfully complete all its transmissions, i.e. $l(T) = \min_{u \in V} l(u)$. We assume that all initial battery charges are equal, $b(u) \equiv b$, $b > 0$.

2.2 Problem definition

The data gathering problem we address in this paper is defined as follows.

We define the network lifetime as the time that the first node runs out of its battery charge (see Section 2).\(^2\)}
For aggregated data gathering, Kalpakakis et al. [9] show an approximation algorithm by Stanford and Tongtong [22], with a restriction that the initial battery charge per node can vary (since all messages must be relayed through a node from L1 in order to reach r). We define $OPT$, as the maximum congestion among all nodes in the solution to $IP_{2-layered}$. The first step in our solution is to create a linear program $LP_{2-layered}$ by changing the binary constraint to:

$$x_{u,v} \geq 0 \quad \forall e_{u,v} \in E$$

Another interesting model is by Liang et al. [15], who explore the data gathering problem without message aggregation for unit disc graphs. They show how to construct a data gathering tree with lifetime that is $\Omega(\frac{\log n}{\log \log n})$ of the optimal, which means there is an instance of the optimal data gathering tree with lifetime better by a factor of at least $\frac{\log n}{\log \log n}$ with respect to their tree solution.

For hierarchical network Levin et al. [13] showed that the data gathering problem is NP-hard for 3-layered graphs when message size per node varies, and for $k$-layered ($k \geq 3$) graphs even if message size is constant per node. They also present a $O(\log n)$ approximation algorithm for the problem on general graphs.

For more details we refer the reader to a recent survey by Ramanan et al. [10], which covers a diverse set of data gathering algorithms in ad-hoc networks.

### 4. THE APPROXIMATION SCHEME

In this section, we present a $k$-approximation algorithm for the data gathering problem on $k$-layered graphs. The proof is divided into two continuous sections. First, we show a factor 2-approximation algorithm for the data gathering problem on 2-layered graphs. Next, we expand the proof for general $k \geq 3$. The 2-approximation result for 2-layered graphs is an extended version of the proof obtained by the authors of this paper in [13]. We present it here since it serves as a basic building block in the $k$-approximation algorithm.

#### 4.1 2-layered graphs

In this version of the problem, the nodes are partitioned into 2 groups: $L_1$ which contains the nodes that are connected to the root, and $L_2$ which contains the nodes that are connected to nodes in $L_1$. Recall that the goal is to minimize the amount of messages transmitted by any node in the resulting tree (also referred to as the congestion). An integer program $IP_{2-layered}$ for the data gathering problem is the following:

$$\min \quad m$$

subject to:

$$\sum_{(v, e_{u,v} \in E)} x_{u,v} = 1 \quad \forall u \in L_2$$

$$\sum_{(u, e_{u,v} \in E)} x_{u,v} q(u) + q(v) \leq m \quad \forall v \in L_1$$

$$x_{u,v} \in \{0,1\} \quad \forall e_{u,v} \in E$$

In this program, $x_{u,v}$ is an indicator variable denoting whether we use node $v$ as a relay for the messages of node $u$ using edge $e_{u,v}$. The goal is to minimize $m$, the maximum congestion among all nodes. Note that the node with the maximum congestion must be a node from $L_1$ (since all messages must be relayed through $L_1$ in order to reach $r$). We define $OPT$, as the maximum congestion among all nodes in the solution to $IP_{2-layered}$. The first step in our solution is to create a linear program $LP_{2-layered}$ by changing the binary constraint to:

$$x_{u,v} \geq 0 \quad \forall e_{u,v} \in E$$

### 3. RELATED WORKS

The problem of efficient data gathering in wireless ad hoc networks has been well studied in the past several years. For aggregated data gathering, Kalpakakis et al. [9] show a polynomial optimal algorithm that runs in $O(n^{2+\epsilon} \log n)$. The long running time was improved using a $1 - \epsilon$ approximation algorithm by Stanford and Tongtong [22], with a $O(n^{1.5+\epsilon} \log n)$ running time. The fastest result for this model can be found at [4]. Some heuristics that are based on this model can be found in [24].

For the partially aggregated data gathering, where the node energy cost depends on the number of neighbors it has, Buragohain et al. [1] present an NP-hardness proof (with the restriction that the initial battery charge per node can vary) and provide a $1 + c_\epsilon$ approximation algorithm, where $c_\epsilon$ is a constant cost of receiving a message.

The maximum lifetime data gathering problem without aggregation was first studied at [8], where Kalpakakis et al. present an integer problem formulation for the problem and provide several heuristic algorithms for it. The authors also evaluate the performance of their algorithms through simulation.

For hierarchical network Levin et al. [13] showed that the data gathering problem is NP-hard for 3-layered graphs when message size per node varies, and for $k$-layered ($k \geq 3$) graphs even if message size is constant per node. They also present a $O(\log n)$ approximation algorithm for the problem on general graphs.
We run a binary search for a minimum value of $m$, 

$$m \in \left[ \min_{u \in V} q(u), \sum_{u \in V} q(u) \right],$$

for which in the solution to $LP_{2-layers}$ all messages are relayed to $r$ (this technique is referred to as parameter pruning). Clearly, the solution is less than or equal to $OPT$.

When the search terminates, some of the $x_{u,v}$ are equal to 1. For those variables, we can relay the messages of node $u$ using node $v$. The partially assigned variables, i.e. $0 < x_{u,v} < 1$, indicate that the corresponding nodes in $L_2$ relay their messages using multiple nodes in $L_1$ (according to the partially assigned variables).

We require the following property derived by Lenstra et al. [12] to provide a solution for the problem.

**Property 4.1.** In a linear program with $LP_{2-layers}$ formulation, after removing the edges which correspond to the fully assigned variables $x_{u,v}$ ($x_{u,v} = 1$), the resulting graph has a perfect matching between nodes from $L_2$ and nodes from $L_1$.

We relay the messages of the nodes from $L_2$ using their matching nodes from $L_1$. The Property 4.1 ensures that each node from $L_1$ relays messages from at most one more node in $L_2$. Since the congestion of any node in $LP_{2-layers}$ is at most $OPT$, thus after the matching, each node from $L_1$ relays messages from at most one more node from $L_2$. This algorithm yields a factor 2-approximation algorithm.

### 4.2 k-layered graphs

Here we show how to extend the results for 2-layered graphs to a general number of layers, $k \geq 2$. To solve the problem, we transform the input graph $G$ to a special flow network. Let $f_{u,v}$ represent the number of messages sent from $u$ to $v$, and $f(v(out))$ and $f(v(in))$ represent the outflow and inflow per node, respectively. We set $f(r(out))$ to $Q = \sum_{u \in V} q(u)$, the sum of all nodes' messages. Let $C$ be an integer in the range $[\min_{u \in V} q(u), Q]$. The integer feasibility program $IP_{k-layers}$ that represents the data gathering problem is as follows:

$$s.t. \quad f(u(out)) - f(u(in)) = q(u) \quad \forall u \in V$$

$$0 \leq f_{u,v} \leq x_{u,v}C \quad \forall e_{u,v} \in E$$

$$\sum_{u,v \in E} x_{u,v} = 1 \quad \forall u \in V$$

$$\{v : e_{u,v} \in E\}$$

$$x_{u,v} \in \{0, 1\} \quad \forall e_{u,v} \in E$$

This formulation resembles the maximum flow problem with two special constraints:

1. For each node $u$, the out flow is un-splittable, i.e., $f(u(out))$ leaves $u$ using a single edge. Therefore, the flow solution $f$ to $IP_{k-layers}$ is a tree. This problem is known in the literature as the confluent flow problem [2].

2. For each edge $e_{u,v}$, the capacity is bounded by $C$.

We define $OPT$ as the minimum value of $C$, for which $IP_{k-layers}$ is feasible. The feasibility derives that all $Q$ messages are delivered to $r$ (i.e., $f(r(in)) = \sum_{u \in V} q(u)$).

A linear program relaxation to $IP_{k-layers}$, $LP_{k-layers}$, is obtained by changing the 2nd and 4th constraints to:

$$0 \leq f(v(out)) \leq C \quad \forall v \in V$$

$$x_{u,v} \geq 0 \quad \forall e_{u,v} \in E$$

The new constraints enable nodes to relay their messages using more than one parent node (i.e., the flow network is not a tree).

The problem with those linear constraints is that they create an unbounded integrality gap between the programs, $IP_{k-layers}$ and $LP_{k-layers}$. The integrality gap is illustrated in Figure 4. In this figure, we have 4 nodes from layers 4 and 2, each with 10 messages to transmit, and node $w$ in layer 3, with no messages to transmit. Setting the capacity of all edges to 10 solves the linear program, while the integer program solution is always 50. If we extend the number of nodes from layers 4 and 2 to $m$, we can conclude that, for this topology, the gap between the two solutions is always $m + 1$.

This unfair advantage results from the fact that the in-flow of a node is not considered as an optimization parameter in the program (only the flow per edge). To fix this issue, for each node $u$, we create a new node $u'$ with no messages; we then connect $u$ to $u'$ and replace the edges $e_{u,v}$ with $e_{u',v}$, while the capacity of all edges remains $C$ (as illustrated in Figure 3). Note that the transformed graph has $2k$ layers. However, since the congestion of $u'$ is equal to the congestion of $u$, we can treat $u$ and $u'$ as a single node and thus the number of layers, for analysis purposes, remains $k$. In addition, the flow solution takes into account the in-flow cost per node, which means that the maximum cost of a node $u$ in the solution will be bounded by $C$, even if $u$ has more than one parent.

Since $LP_{k-layers}$ has the same formulation as the maximum flow problem when all the edges have a maximum capacity of $C$, we can solve the data gathering problem by searching for a minimum value of $C$ for which all flow demands are met (i.e., $Q$ units of flow are successfully delivered.
to $r$). We define this value as $OPT^*$, the resulting flow as $f^*$, and the actual flow between any two nodes $u$ and $v$ as $f^*_{u,v}$. It is easy to observe that since nodes can relay their messages using more than one parent, $OPT^* \leq OPT$.

We observe that in $LP_k$-layers the variables $x_{u,v}$ represent the relative part of out flow that leaves through edge $e_{u,v}$ (i.e., $x^*_{u,v} = \frac{f_{u,v}}{f(u(out))}$). In $f^*$, we consider an edge $e_{u,v}$ only if $x_{u,v} \geq 0$.

In the case $x_{u,v} = 1$, then node $u$ can use only node $v$ as a relay for its messages. We remove those edges from the graph.

Next, we examine the following feasibility problem $FP_i$ between any two layers $i$ and $i - 1$ ($1 < i \leq k$):

$$
\sum_{\{v : e_{u,v} \in E_i\}} x_{u,v} = 1 \quad \forall u \in L_i
$$

$$
\sum_{\{v : e_{u,v} \in E_i\}} x_{u,v} f(v(out)) + q(v) \leq OPT^* \quad \forall v \in L_{i-1}
$$

$$
0 \leq x_{u,v} \leq 1 \quad \forall e_{u,v} \in E_i
$$

In this program, $E_i$ represents the edge set between nodes from $L_i$ to nodes from $L_{i-1}$. We claim the following:

**Lemma 4.2.** The flow solution to $LP_k$-layers, $f^*$, is a feasible solution to all $FP_i$ ($1 < i \leq k$).

**Proof.** By setting $x^*_{u,v} = \frac{f_{u,v}}{f(u(out))}$, we get a basic feasible solution since:

$$
\sum_{\{v : e_{u,v} \in E_i\}} x^*_{u,v} = \sum_{\{v : e_{u,v} \in E_i\}} \frac{f_{u,v}}{f(u(out))} = 1
$$

$$
\sum_{\{v : e_{u,v} \in E_i\}} x^*_{u,v} f(u(out)) + q(v)
$$

$$
= \sum_{\{v : e_{u,v} \in E_i\}} \frac{f_{u,v}}{f(u(out))} f(u(out)) + q(v)
$$

$$
= f(v(out)) \leq OPT^* \square
$$

The following lemma shows a special property of the solution to $LP_k$-layers.

**Lemma 4.3.** In the solution of $LP_k$-layers, after removing nodes that have only one parent, for the remaining nodes in all layers $i$ ($1 < i \leq k$), there is a perfect matching between each node in layer $i$ to a node in layer $i - 1$.

**Proof.** Lemma 4.2 shows that $f^*$ yields a feasible solution for any $FP_i$ ($1 < i \leq k$). Since $FP_i$ has the same linear programming formulation as $LP_2$-layers, property 4.1 applies to it as well. Thus, we have a perfect matching between any two adjacent layers, and the lemma holds. $\square$

After finding the flow $f^*$ that minimizes $C$, the maximum cost of any node in the network is less than or equal to $OPT^*$. We remove all nodes that have only one parent in the flow network. From Lemma 4.3 we know that for the remaining nodes, between any two adjacent layers $i$ and $i - 1$ ($1 < i \leq k$), there is a matching between nodes from layer $i$ to nodes from layer $i - 1$.

We assign each node $u$ from layer $i$ to its matching node $v$ from layer $i - 1$, and relay node’s $u$ messages using the edge $e_{u,v}$. Since each node is assigned to a lower layer node at most once and since the maximum cost before the assignment is $OPT^*$, we derive that the cost of a node in layer $i$, can increase by at most $(k - i + 2)OPT$. For example, after the assignment, the cost of a node in layer $k - 1$ increases by at most $OPT^*$. Then, at the worst case, the cost of a node in layer $k - 2$ increases by at most $2OPT^* + OPT^*$, and so on. Thus, the maximum cost of a node in any layer is at most $OPT + (k - 1)OPT$. This leads us to the following theorem:

**Theorem 4.4.** There is a $k$-approximation algorithm for the data gathering problem in $k$-layered graphs.

The Data Gathering Algorithm (DGA), which is given below, summarizes the different steps of our solution.

**Data Gathering Algorithm (DGA)**

**Input:** A $k$-layered graph $G$, a message size vector per node $q$, and a root $r$

**Output:** A data gathering tree $T$

1. $T \leftarrow \emptyset$
2. Create a flow network $f$ on $G$ as follows:
   - Set the flow demand per node $u$ as $q(u)$
   - Create a source node and connect it to all other nodes in the network
   - Set the capacity of all edges to $m$
3. Search for a minimum $m$ such that in the resulting flow $f$, all demands are satisfied
4. If a node $u$ delivers its flow using only one parent $v$ in $f$, add $e_{u,v}$ to $T$ and remove $e_{u,v}$ from $f$
5. For $i \leftarrow 1$ to $2$ do
   - Select edges between layers $i$ to $i - 1$ that transfer positive amount of flow
   - Using those edges, find a perfect matching between the nodes in layer $i$ to the nodes in layer $i - 1$
   - Add the matched edges to $T$ and remove them from $f$
6. Return $T$

**5. Numerical Results**

In this section, the performance of the DGA algorithm is evaluated through simulations. We compare between $OPT^*$, the total cost of $LP_k$-layers, to the cost of DGA under different topology conditions. Recall that $OPT^* \leq OPT$ as we explained in Section 4.2.

We obtained the result of $OPT^*$ by searching for a minimum value of $C$, for which the flow algorithm yields a total flow of $Q = \sum_{u \in V} q(u)$ on the input graph. The solution for our approximation algorithm was obtained using Algorithm 1.

First numerical results, which are depicted in Figure 5, show the approximation ratio of the DGA as the number of layers, $k$, varies. The number of nodes per layer is fixed and set to 5. Each node in layer $i$, has between 1 to 5
neighbors from the upper layer (chosen with uniform distribution). The low number of nodes per layer leads to a network with smaller density. Our results show that for such networks, even when $k$ increases, the approximation ratio remains fixed.

In Figure 6 we evaluate the approximation ratio when $k$, the number of layers, remains fixed. We set $k$ to 6 and progressively increase the number of nodes per layer. The growing number of nodes leads to a denser network. We can observe that even as the number of nodes per layer increases, the approximation ratio remains within the theoretical bounds.

Figure 5: Approximation ratio for input graphs with different number of layers.

Figure 6: Approximation ratio for input graphs with different number of nodes.

6. CONCLUSIONS AND FUTURE WORK

This paper studies the data gathering problem in hierarchical ad hoc networks. We presented DGA, an approximation algorithm for the problem, which yields an approximation ratio of $k$, the number of layers. That is, the performance of the algorithm depends on the depth of the hierarchical input graph. We also provide several numerical results to support our theoretical analysis.

It would be interesting to explore the characteristics of the problem under dynamic topologies that evolve over time, where graph properties, such as nodes per layer or edges between nodes, change between communication rounds.

Another interesting research question is what effect does the hierarchical graph structure has on the network capacity, delay, and interference.

7. REFERENCES


