

## 1 Lecture topic

This lecture is about the presentation of the papers

1. “*LOCI: Local Clustering Service for Large Scale Wireless Sensor Networks*” by Vineet Mittal, Murat Demirbas, and Anish Arora.
2. “*Sparse Partitions*” by Baruch Awerbuch and David Peleg.

The papers were presented to the class by Puviyarasan Pandian.

## 2 LOCI paper

### 2.1 Introduction

With large-scale wireless networks, the associated challenges of self-configuration of the network, self-healing in the case of any failure and self-maintenance assume significant importance. Without even one these features, the sensor network formed cannot be used any further. A possible solution to this problem is to use clustering of the nodes in the network.

### 2.2 Clustering

A geographical-radius based clustering has several advantages.

1. The network structure obtained by clustering is independent of the number of nodes in the network.
2. Efficient communication is possible within the geographical region, which is very essential for energy consumption in wireless sensor networks.
3. Efficient evaluation of spatial queries, which is normally the case with wireless sensor networks, as we discussed in the class.

A lot of research has gone into the problem of clustering in static networks. The problem of clustering in wireless sensor networks presents us several more challenges to address, which are as follows:

1. *Almost equal sized clusters.* This ensures that the network is load balanced and there is uniform energy dissipation.
2. *Minimize cluster overlap.* This is to ensure that the energy lost by the overlapping nodes is minimized and hence lesser energy requirements.
3. *Minimize orphan nodes.* This maximizes resource availability for the network and hence allows load balancing and saves energy.
4. *Minimize communication overhead,* which saves energy.
5. *Minimize number of clusters.* The lesser number of clusters, we store less overhead information about clusters and hence decreases memory requirements.
6. *Scalable cluster formation and self-healing.* This is to make the network fault tolerant.

LOCI(Local Clustering Service for Large Scale Wireless Sensor Networks), the distributed algorithm presented in this paper partitions the network into clusters with the following properties.

1. *Locality* - Each node only needs information about nodes that are at most distance  $2R$  away.
2. *Scalability* - Each node maintains a constant amount of state information and each node completes its role in clustering in  $O(R^4)$  time, independent of the network size. Hence, expansion of the network does not affect the clustering formation time and only nodes in the neighborhood of the newly expanded network get affected.
3. *Fault-tolerance* - The clustering in the network is locally self-stabilizing.

### 2.2.1 Fixed-Radius clustering

The direct and first step would be to come up with a clustering where the radius of the cluster,  $R$ , is fixed. Let us consider one such clustering scenario as shown in Figure 1.

In such a scenario of chain network, with fixed radius clustering, a new node addition causes a cascading effect on all the clusters in the network making it not a scalable solution. If this approach is adopted, every node addition and deletion from the network will modify the whole clustering and it is a costly operation. Hence, such an approach will not be preferred in any large-scale network.

Hence, to come up with a network which re-organizes itself with node additions and deletions, an algorithm with bounded range radius is proposed in this paper. The range for the radius values of the network is  $[R, mR]$ .

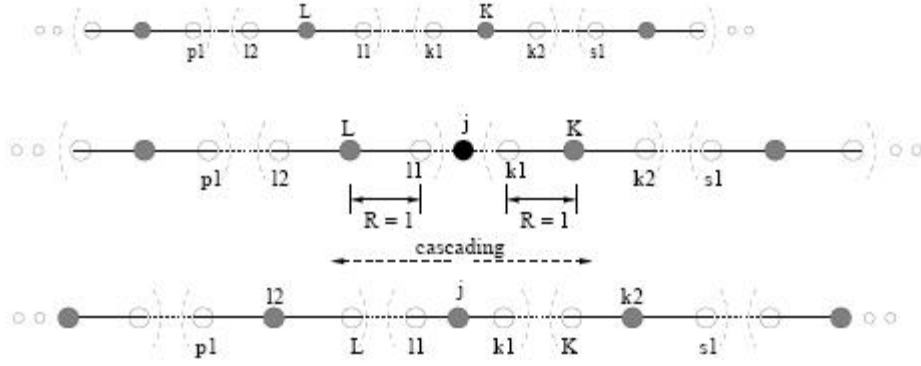


Figure 1: Fixed Radius clustering cascading scenario

### 2.3 LOCI

First, let us consider a case of this algorithm to get an idea on how this works. When radius( $R$ ) is 1 in the network and a value of  $m$  as 2, the scenario is shown in Figure 2.

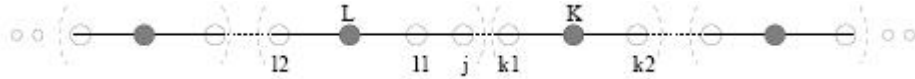


Figure 2: Bounded radius when  $R = 1$  and  $m = 2$  and each cluster with radius 1

The network chain includes all the clusters with  $R = 1$ . In such a network, a new node  $j$  when it comes up joins one of the clusters adjoining its current location and this cluster's radius becomes 2.

When the network chain is full of clusters each with radius,  $R$ , as 2, as shown in Figure 3, and when a new node  $j$  joins the network, it elects itself as the leader of the cluster and takes in its neighbors to form a cluster of radius 1. This is shown in Figure 3.

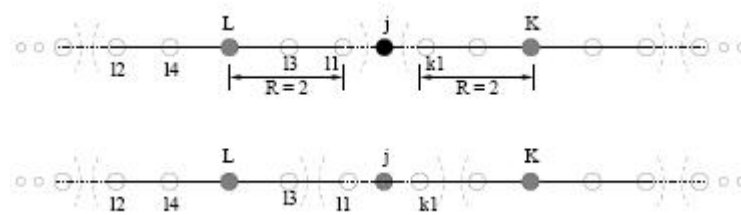


Figure 3: Bounded radius when  $R = 1$  and  $m = 2$  and each cluster with radius 2.

#### 2.3.1 Model and Definitions

The model assumes that the nodes in the network are in a 2-dimensional coordinate plane. A node in this graph  $G = (V, E)$  is connected to another node iff the nodes are at unit distance.

A *cluster* is defined as a subset of vertices,  $S \subseteq V$  in  $G$ , such that the graph  $G(S') = (S, E')$ , where  $E'$  consists of all the edges in  $E$  such that both vertices of the edge lie in  $S$ .

A node  $v$  is in the  $r$ -neighborhood of another node  $u$ , if node  $v$  is distance  $r$  away from node  $u$ .

### 2.3.2 Assumptions

1. Each node has a unique ID for unique priority assignment to clusters
2. Nodes have distance estimation capability relative to their neighbors, for cluster construction based on geographic radius.
3. Nodes are distributed as per a homogeneous spatial Poisson process of intensity  $\lambda$ .

### 2.3.3 Problem Statement

Given a cluster radius interval  $[R, mR]$ , where  $R$  is a natural number and  $m$  is a constant  $\geq 1$ , construct a partition  $\{S_1, S_2, \dots, S_p\}$  such that

- A unique node is designated as a leader of each cluster.
- All nodes in the  $R$ -neighborhood of each leader belong to that cluster.
- Maximum distance of a node from its leader is  $mR$ .
- Each node belongs to the cluster of the closest clusterhead.
- There exists a path from every node to its leader which consists of only the nodes that belong to the same cluster.

### 2.3.4 Algorithm Intuition

A node  $j$ , joining the network, is eligible to become a cluster head if all the following conditions are satisfied.

- After a random waiting time has expired.
- Node  $j$ , does not belong to any cluster.
- Node  $j$ , is not within  $mR$ -neighborhood of the cluster heads(if any) of its neighboring nodes.

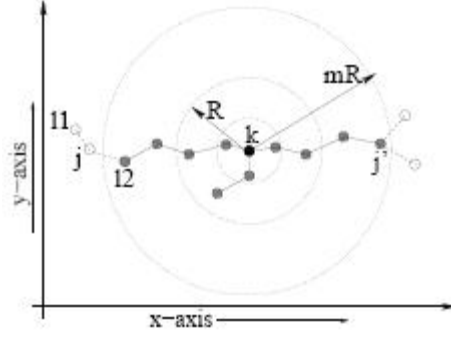


Figure 4: Node joining in LOCI

Node  $j$  starts a cluster by electing itself as the clusterhead and subsumes its neighboring nodes, within distance  $R$ . Thus node  $j$  forms a  $R$ -radius circular disc as shown in Figure 4.

In Figure 4, node  $k$  is a clusterhead of a legitimate cluster and  $l_1$  does not belong to any cluster. The new node joining the network,  $j$ , is not within  $mR$ -neighborhood of any of the adjoining clusterheads. Thus node  $j$  can become a clusterhead but  $j'$  which is within the  $mR$ -neighborhood of clusterhead  $k$  cannot become a clusterhead.

If the node  $j$  does not belong to any cluster and is within  $mR$ -neighborhood of some neighboring clusterhead  $k$ ,  $j$  joins the cluster of  $k$ . If node  $j$  already belongs to a cluster and is not within  $R$ -neighborhood of its clusterhead and is within  $mR$ -neighborhood of  $k$ 's clusterhead,  $j$  joins the cluster of  $k$ . This allows nodes not in the within  $R - mR$  distance from the clusterheads to leave the cluster and join another cluster. In Figure 4,  $l_1$  can join  $j$ 's cluster thus enabling the node to join their closest clusterheads at any point of time.

This algorithm could cause an overlap occurring between  $R$ -neighborhoods of clusterheads, due to a local clusterhead election. A priority is assigned to each cluster and this priority value is used for resolving such conflicts. This priority given by the lexicographical ordering  $\langle Rad, -O, ID \rangle$  where  $Rad$  is the radius of the cluster,  $O$  is the number of overlaps of the cluster with other equal radius clusters, and  $ID$  is the identity of the clusterhead. The rule for cluster prioritization is that the cluster with maximum radius has highest priority, followed by the cluster with minimum number of overlaps, and finally by the cluster with maximum ID.

If priority of clusterhead  $j$  is lower than that of clusterhead  $k$ , clusterhead  $j$  reduces its radius by one thus allowing higher priority clusterhead  $k$  to subsume the overlapping nodes of clusterhead  $j$ . Thus either, the lower priority cluster is either completely subsumed. This cluster grows again if all the nodes neighboring  $j$ 's cluster can join the cluster of  $j$ . Cluster  $k$  in turn can lose to some other higher priority cluster decrementing its radius and hence priority, and thereby allowing cluster  $j$  to grow again. In Figure 5, Node  $l_2$  moves to clusterhead  $k$  from clusterhead  $j$ .

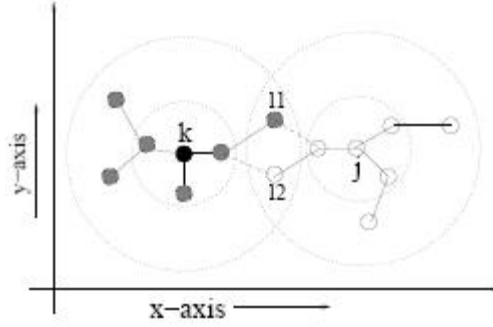


Figure 5: Overlapping of clusters in LOCI

## 2.4 Extensions and Future work

A hierarchical level of such clustering would serve several applications that can be built on top of LOCI. In such a scenario, a multilevel partitioning can be done in a bottom-up fashion as follows.

1. A normal  $[R, mR]$  partitioning is done first in the network. Let this be called as the lowest level or, *Level0partitioning*.
2. For the next level of partitioning, clusters at level 0 are represented by clusterheads. Now, the clusterheads of *Level - 0* are separated by  $[2R, 2mR]$  and this is used as the bounded radius range for *Level - 1partitioning*. Thus, unit distance at level 1 is  $2mR$ .
3. This level of partitioning is repeated to get further higher levels of partitions.

Stability of such a hierarchical clustering technique is important for the reason that this hierarchical construction cannot be done every time when a node joins or leaves the network, which is an expensive operation. A node at *Level - i* clusterhead is also a clusterhead for every level up to  $i$ . Hence, when *Level - i* clusterhead fails, a new cluster is formed at the lowest level and a similar bottom up construction is followed for the new clusterhead.

Future work as suggested by the authors is a geometric, local clustering, which would help in designing efficient data structures for evaluation of spatial queries in sensor networks.

## 3 Sparse Partitions

### 3.1 Introduction

A distributed system presents several challenges and a lot of these challenges have been addressed in several papers that we discussed in the class. This paper addresses the problem of control and management in distributed systems. A naive implementation would be to use a global approach. This would be expensive since the network can be really huge and storing data about the topology of the network might take a huge amount of memory. Storing the status of the network would also be expensive. This also presents the other twin problem of the huge complexity in updating and maintaining this global data.

### 3.2 Locality-based approach

The idea is to develop a clustered representation of networks, where the entire network is divided into clusters. Then, we obtain a cover of the entire network so that we ensure that every node in the network is covered by some cluster or the other. This locality-based approach also ensures that limited knowledge on the state of the network or rather the local cluster is sufficient to maintain in the node.

A protocol developed assuming the clustered representation of a network should ensure the following.

1. Cost of communication between any two nodes depends on locations of the source-destination pair in the actual network and not just the clustered network.
2. Any activity between nodes in the same cluster involves local information of the cluster and should be cheap
3. Any interaction between nodes across clusters is not preferred since that requires more information than just the cluster and hence should be costly.

A simple naive implementation is neighboring nodes form clusters. In such an implementation, it is possible that two nodes which lie in the border of adjacent clusters, though neighbors in the original network, belong to different clusters in the clustered network and hence any interaction between these two nodes is a costly operation. Hence, any such clustering cover obtained should preserve the locality information present in the original network.

### 3.3 Cluster cover

The quality of a cover obtained thus should have a small cluster size. Lesser radii clusters involve less internal communication within the cluster and hence better. Sparsity of the cover can be intuitively defined as the overlap of clusters and they provide a measure of the interaction of

vertices of one cluster with the neighboring cluster. Sparsity is provided by  $\Delta(S)$ , the maximum number of occurrences of a vertex in clusters  $S \in \mathcal{S}$ . Lower sparsity would hence mean low memory requirements in the node, since overlap of clusters is less and hence the clustering is sparse. Communication complexity of cover-based protocols typically are in the order of  $O(\Delta(S) * Rad(Cover))$  and hence this would provide us with the conflicting goals of lower radii of clusters and lower sparsity.

### 3.3.1 Coarsening of covers

Coarsening of a cover is the process where, given some initial cover of network,  $S$ , repeated cluster merging is performed in this cover and a coarsened cover,  $T$ , is constructed. The coarsened cover  $T$  should guarantee that each original cluster in  $S$  is fully subsumed in one of the clusters of  $T$ , thus reducing sparsity. This coarsening is done with care taken such that the cluster radii in the coarsened cover,  $T$ , are not much larger than those of  $S$ .

```

 $U \leftarrow \mathcal{R}; DT \leftarrow \emptyset; DR \leftarrow \emptyset$ 
repeat
  Select an arbitrary cluster  $S \in U$ .
   $Z \leftarrow \{S\}$ 
  repeat
     $\mathcal{Y} \leftarrow Z$ 
     $Y \leftarrow \bigcup_{S \in \mathcal{Y}} S$ 
     $Z \leftarrow \{S \mid S \in U, S \cap Y \neq \emptyset\}$ .
  until  $|Z| \leq |\mathcal{R}|^{1/k} |Y|$ 
   $U \leftarrow U - Z$ 
   $DT \leftarrow DT \cup \{Y\}$ 
   $DR \leftarrow DR \cup Y$ 
until  $U = \emptyset$ 
Output  $(DR, DT)$ .

```

Figure 6: Algorithm *Cover*

The paper proposes an algorithm for coarsening of covers. The algorithm, shown in Figure 6, takes the following inputs: a graph  $G = (V, E)$  and  $R$ , a collection of possibly overlapping clusters and an integer  $k \geq 1$ , which is used to bound the number of runs and hence the radius of the clusters in the coarsened cover. Its output is:  $DT$ , a collection of disjoint clusters that subsume a subset of  $R$  ( $DR$ ). The algorithm ensures us the following, as stated in *Lemma 3.2* of the paper.

- $DT$  coarsens  $DR$
- $Y \cap Y' = \emptyset$ , for every  $Y, Y' \in DT$

- $|DR| \geq |R|^{1-\frac{1}{k}}$
- $Rad(DT) \leq (2k - 1)Rad(R)$

The algorithm *Cover* does not guarantee us that each of the original clusters in  $DR$  is covered by some cluster  $Y \in DT$ . Also several clusters present in the initial collection are thrown out of consideration without being subsumed by any cluster in  $DT$ .

```

R ← S; T ← ∅
repeat
  (DR, DT) ← Cover(R)
  T ← T ∪ DT
  R ← R \ DR
until R = ∅

```

Figure 7: Algorithm *MAX\_COVER*

The paper also comes up with a *MAX\_COVER* algorithm, shown in Figure 7, that calls the *Cover* algorithm repeatedly till a sparse cover is obtained which ensures that every cluster in the initial setup has the locality information preserved and the cover is as sparse as possible. The algorithm takes the following inputs: a graph  $G = (V, E)$  and  $S$ , a cover of the network denoted by  $G$  and an integer  $k \geq 1$ . Its output is:  $T$ , a coarsening cover of  $S$ . This algorithm provides us the following guarantee, stated in *Theorem – 1* of the paper.

- $Rad(T) \leq (2k - 1)Rad(S)$
- $\Delta(T) \leq 2k|S|^{\frac{1}{k}}$

### 3.4 Applications

This algorithm for generating a sparse cover can be used in several related problems such as distributed regional matchings, generating tree covers and in generating sparse partitions. The part we discuss here is the Extended Abstract of the original paper and the actual sparse partition algorithm is provided in the full paper.

A hierarchical level of clustering can also be performed with this technique. A hierarchy of covers with higher levels using larger radii clusters is constructed. Communications and data are organized at the appropriate level and costs are according to the locality level of application. Between neighbors, it uses the lowest level and hence costs less. Between distant nodes, higher level of the hierarchy is used and this costs accordingly.

A sparse partition generated using such an algorithm can be used for several applications such as routing, online mobile tracking of users.