A New Classification of Backbone Formation Algorithms for Wireless Sensor Networks

Razieh Asgarnezhad
Department of Computer Engineering
Arak Branch, Islamic Azad University
Arak, Iran
raziehasgarnezhad@yahoo.com

Javad Akbari Torkestani
Department of Computer Engineering
Arak Branch, Islamic Azad University
Arak, Iran
j-akbari@iau-arak.ac.ir

***Abstract*—In Wireless Sensor Networks, the most important of challenges is the bandwidth and energy limitations, network topology changes, and the lack of the fixed infrastructures. There is no fixed backbone infrastructure in these networks. Flooding is a kind of broadcasting in sensor networks. But it raises energy consumption because packet retransmission is needed when interference occurs. Also, it will has broadcast storm problem. To solve these circumstances, virtual backbone can be used. A backbone is a subset of active nodes while the rest of the sensors are sleeping. It is able to perform especial tasks and serve nodes which are not in the backbone. For instance, backbone nodes in networks can perform efficient routing and broadcasting. A backbone reduces the communication overhead, increases the bandwidth efficiency, decreases the overall energy consumption, and, at last, increases network effective lifetime in a Wireless Sensor Network. This paper classifies different backbone formation algorithms. We compare performance of these with each other.**

Keywords- backbone formation; clustering; connected dominating set; maximal independent set; wireless sensor network.

I. INTRODUCTION

Wireless Sensor Networks (WSNs) have attracted recent research attention due to wide range of applications they support. These networks consists a number of wireless nodes so that all nodes are energy constrained. Sensors are equipped with data processing and communication capabilities. Each sensor can be used to send the collected data to interested parties. The WSNs can be divided into three parts: data collection, based-station and data management center. In WSN, there is no fixed or predefined infrastructure. Flooding is a kind of broadcasting in sensor networks, where each node retransmits the broadcasting message that it receives. But it raises energy consumption because packet retransmission is needed when interference occurs. Also, it will has broadcast storm problem. [2][15]

The extensive research performed in the past of decades in WSNs. Among the topics that clustering formation and interconnection (referred as *backbone formation*) have received especially attention. Backbone will remove unnecessary transmission links through shutting down some of redundant nodes. Although backbone will still guarantee network connectivity in order to deliver data efficiently in a WSN. [5]

A backbone is a subset of active nodes while the rest of the sensors are sleeping. Backbones are able to perform especial tasks and serve nodes which are not in backbone. Therefore, the backbone construction depends on the task to be carried. The backbone of a network is normally required to be connected. For example, connected backbone node in ad hoc networks can perform efficient routing and broadcasting. The most use of backbones is improving of the routing procedure. A backbone reduces the communication overhead, increases the bandwidth efficiency, decreases the overall energy consumption and at last increases network effective lifetime in a WSN. [20]

There are typically three well known methods to constructed backbones: (1) *grid partitioning-based* (2) *clustering-based,* (3) *connected dominating set* (*CDS*)*-based*. In first method, the area of network is divided into grids and one node in each grid is selected as a backbone node. The size of grid should be carefully determined to guarantee that the backbone is connected. In second method, nodes are grouped into clusters. A node is elected as the *cluster-head* (CH) in each cluster. Any node in the network is either a CH or a neighbor of a CH. Rest nodes are required to be included to make the CHs connected. In third method, routing is easier and can adapt quickly to network topology changes. To reduce the traffic during communication, it is desirable to was constructed a Minimum Connected Dominating Set (MCDS). [7][8][14][17][18][20][24][26]

We try to classify different backbone formation algorithms in these networks and compare performance of these with each other. Based on these methods, we have proposed new hybrid methods in this paper. In Section 2, we exhibited these methods and some examples compared in Section 3. In Section 4, we concluded the paper.

II. CLASSIFICATION OF BACKBONE FORMATION ALGORITHMS

From varied aspects, backbone formation algorithms can be classified into different types. Keeping some classifications in view, we present a few instances of these classifications and we propose new hybrid methods.

A. Grid Partitioning-Based Backbone

In this method, the area of the network is divided into grids and one node in each grid is selected as a backbone node. The size of grid should be carefully determined to guarantee that the backbone is connected.

Geographical adaptive fidelity (GAF) is a grid partitioning algorithm for backbone construction. In this algorithm, each GAF node uses location information itself. The algorithm divides the network into virtual grids so that nodes are distributed into small virtual grids. Any node in one grid can directly communicate with any node in the other grid. This is why that all nodes in the same grid are *equivalent*. Thus, one node from each grid is enough to construct a connected backbone. According to virtual grid, any node in adjacent grid can communicate with each other. The communication range is supposed deterministic. Assume r is the size of the virtual grid, and also R is the transmission range. Because any two nodes in adjacent grids can be communicate with each other, this equation can be used for grids: [26]

$$r^2 + (2r)^2 \leq R^2 \rightarrow r \leq R / \sqrt{5} \quad (1)$$

B. Clustering-Based Backbone

Clustering is method for partitioning nodes of the network into groups. CHs are used to dominate the other nodes within the clusters. Clustering can provide a hierarchical architecture for efficient routing. At most existing solutions for clustering usually consists of two phases: construction and maintenance. In the first phase, nodes are chosen to act such as coordinators of the clusters. Then, clustering maintenance is required to reorganize the clusters due to mobility and failure of nodes. [7][14][18][24]

Low-energy adaptive clustering hierarchy (LEACH) is a protocol. According to this protocol randomly decide whether or not to become CHs. The parameter used in decision making is the percentage of desired CHs in the network. In this protocol, sensors that decide to become CHs broadcast their decision. Each node reports to the CH with the highest signal strength. Selection of CHs is periodically repeated to balance energy consumption of nodes. The structure of the clusters constructed through LEACH is inefficient because the sink may be very far from many CHs. [14]

A clustering algorithm proved that only clustering schemes that position their resultant clusters within the isoclusters of the monitored phenomenon are guaranteed to reduce the nodes' energy consumption and extend the network lifetime. This was the first clustering algorithm; it employs the similarity of the nodes' readings as the main criterion in cluster formation. [24]

Another algorithm [18] proposed a mechanism as no two CHs could be direct neighbors and any other node should be adjacent to at least one CH. Each node has a unique node *key* and also knows the *key*s of its one hop neighbors. The basic idea behind the CH algorithm is to use the node *key* as a priority indicator when selecting CH in each cluster. Each node compares its *key* with the *key*s of its neighbors. At first, all nodes are undecided. If a undecided node has the lowest *key* among its undecided neighbors, the node decides to create its own cluster and broadcasts the decision and its *key* as the cluster *key*. Upon receiving a message from a neighbor so that announces itself to be a CH, each undecided node will declare itself as a non-CH node and also will inform its neighbors through transmitting a message. [18]

Distributed mobility-adaptive clustering (DMAC) is a distributed clustering algorithm. It uses a mechanism similar to the algorithm in Lin and Gerla [18] to construct clusters. But, it uses the weight (the rest energy in the cluster or the capacity of the nodes) of the nodes instead of node ids as keys. This algorithm is followed with such weight instead of the original lowest id used in Lin and Gerla [18]. The basis behind the DMAC is a protocol for the topology control of large WSNs that Basagni et al. [8] proposed and called S-DMAC. This protocol is used to select a subset of nodes to build a connected backbone and let all other nodes switch to an energy conserving *sleep mode*. A connected backbone includes of backbone nodes and gateway nodes so that interconnect the backbone nodes. Backbone nodes are the CHs computed by DMAC. S-DMAC optimized the overhead at both stages consist of construction and maintenance through limiting the use of *hello* messages. The backbone is reorganized only in two times. First, introducing a new batch of nodes with much higher energy than the current nodes, second backbone nodes deplete their energy. A non-backbone node will join a newly inserted backbone node when the residual energy of the new backbone node exceeds the original one's energy through a predefined threshold. [7]

Virtual Backbone for Energy Saving (ViBES) is a backbone algorithm. It uses the energy efficient construction. The idea behind ViBES was a subset of the sensor nodes that formed a connected backbone (the selected nodes via intermediate nodes and links). A small part of the nodes are selected to be the backbone, and the actual backbone is created through connecting the selected nodes via intermediate nodes and links. ViBES construction included of two important phases: (1) selection of primary ViBES nodes (2) their interconnection to form a connected backbone.The selection of the ViBES nodes is performed at each node according to the algorithm proposed in [8]. Every node has a unique id, a generic weight and also knows about the id and the weight of its one hop neighbors. Nodes that have the biggest weight among their neighbors become primary ViBES nodes. The other nodes decide to be primary ViBES nodes or ordinary nodes corresponding with the decision of all the neighbors with a bigger weight. At last, the process terminated when all sensor nodes be partitioned into primary ViBES nodes and ordinary nodes. A backbone is constructed through connecting the primary ViBES nodes via some ordinary nodes. Keeping this algorithm in view, primary ViBES nodes that are two or three hops away, select interconnection nodes until be part of the backbone. Thus, the backbone paths formed guarantee that the final backbone is connected. Figure 1 illustrates the process of selection of ViBES nodes. [6]

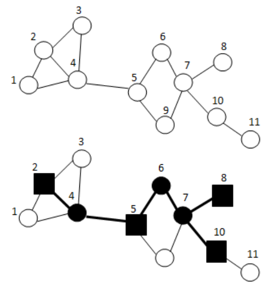

Figure 1. illustrates the process of selection of ViBES nodes. [6] Rectangles and black circles construct the final backbone.

C. Connected Dominating Set (CDS)-Based Backbone

From varii aspects, CDS construction algorithms can be classified into different types. Keeping some classifications in view, we exhibited a few instance of these classifications.

1) UDG and DGB

The CDS construction algorithms can classified into two types: *Unit Disk Graph* (UDG) based algorithms and *Disk Graphs with Bidirectional* (DGB) links. In UDG and DGB, the link between any pair of nodes is bidirectional. The nodes transmission ranges in UDG are the same but in DGB are different. The MCDS in UDG and DGB has been shown to be NP-hard. [2][19][20][21]

2) MIS based and Non-MIS based

Independent set (IS) of a graph G is a subset of vertices so that no two vertices are adjacent in the subset. *Maximal Independent set* (MIS) is an IS, so that it is not a subset of any other IS. Note that in an undirected graph, a MIS is also a *Dominating Set* (DS). The MIS based algorithms have two kinds of realization. The optimal node selection is based on some criteria such as node degree, rest energy of node, and node id. [12][20][22][23]

3) Centralized algorithm and Decentralized algorithm

Algorithms that construct a CDS can be divided into two types: centralized and decentralized. The centralized algorithms in general result in a smaller CDS with a better performance ratio than that of decentralized algorithm. The decentralized algorithms also can be divided into two types: distributed and localized. In distributed algorithms, the decision process is decentralized. But in the localized algorithm, the decision process is not only distributed also requires only a constant number of communication rounds. Most of the distributed algorithms find a MIS and connect this set. [3][13][20][22][23]

Two CDS construction approaches were proposed. The first algorithm begins through marking all vertices white. It selects the node with the maximal number of white neighbors. The selected vertex is marked black and also its neighbors are marked gray. The algorithm iteratively seeks the gray nodes and their white neighbors and selects the gray node or the pair of nodes, whichever has the maximal number of white neighbors. The selected node or the selected pair of nodes is marked black, and also their white neighbors marked gray. Finally, the algorithm terminates, when all of the vertices are marked gray or black. All the black nodes form a CDS. This algorithm results in a CDS of size at most $2(1+H(\Delta)).|OPT|$, where H is the harmonic function and *OPT* refers to an MCDS. [13]

The second algorithm also begins through coloring all nodes white. A *piece* is defined to be either a connected black component or a white node. The algorithm includes two phases. The first phase iteratively selects a node that yield the maximum reduction of the number of pieces. A node is marked black and its white neighbors are marked gray when it is selected. The first phase terminates when no white node left. There exists at most $|OPT|$ number of connected black components. The second phase constructs a Steiner Tree until connects all the black nodes through coloring chains of two gray and black nodes. The size of the resulting CDS formed via all black nodes is at most $(3+ln(\Delta)).|OPT|$. [13]

A greedy algorithm was proposed for MCDS in UDGs. At first, all nodes are colored white. The construction of a CDS includes four phases. The first phase is computing an MIS and coloring all its members red. In the second phase, a node selects that it can decrease the maximum number of pieces. This node is colored black and all its non-black neighbors are colored gray. After the second phase, we still have some white nodes left. The third phase will compute a spanning tree for each connected component in the sub graph reduced through all white nodes. All non-leaf tree nodes are colored black but leaf nodes are colored gray. The last phase will scan chains of two gray nodes to connect disjoint black components. [11]

The pruning-based heuristic was proposed. The *S '* CDS is initialized to the vertex set of graph *G(V, E)*. Then each node will be examined to determine whether it should be removed or remained. At first, all nodes in *S* are colored white. The *effective degree* of a node defined to be its white neighbors in *S*. With considering a white node $x \in S$ with minimum effective degree if removing *x* from *S* makes the resulted graph of *S* disconnected, then retain *x* and color it black. Otherwise, remove *x* from *S*. If *x* does not have a black neighbor in *S*, color its neighbor with maximum effective degree in *S* black. With repeating this procedure no white node left in *S*. At first, the algorithm starts from the node with minimum degree, which can be found through modified leader election algorithms in [16]. Let *u* be the node that we consider at the current step. If removing *u* causes the CDS disconnected, we color *u* black. Then, it selects its non-black neighbor with minimum effective degree for consideration in next step. If it is OK to remove *u* and if *u* does not have a black neighbor for next step, then *u* will select a neighbor with minimum effective degree. If *u* does have a black neighbor *v*, therefore *v* will choose its neighbor with minimum effective degree for next step. This procedure will be terminated when all nodes have been examined. This algorithm has time complexity $O(n \log^3(n))$

But, its distributed implementation has higher message complexity. [9]

The distributed implementations of the two greedy algorithms had been proposed. The first algorithm grows one node with maximum degree to be form a CDS. Thus, a node must know the degree of all nodes in the graph. This algorithm produces a CDS with approximation ratio of *2* $H(\Delta)$ in $O(|C|(\Delta+|C|))$ time, using the $O(n|C|)$ messages, where the harmonic function, n is the total number of vertices, and C represents the final CDS. [12]

In the second algorithm, compute a DS and then selects additional nodes to connect the set. Then, an unmarked node compares its effective degree, with the effective degrees of all its neighbors in two-hop neighborhood. The greedy algorithm adds the node with maximum effective degree to the DS. When a DS is achieved, the first stage terminates. The second stage connects the components via a distributed minimum spanning tree algorithm. This is why that each edge is assigned a weight equal to the number of endpoints not in the DS. Finally, the nodes in the resulting spanning tree compose a CDS. This algorithm has time complexity of $O((n+|C|)\Delta)$, and message complexity of $O(n|C|+m+n.\log(n))$. It have the MCDS with a ratio of $2H(\Delta)+1$, where m is the cardinality of the edge set. [12]

Two versions of an algorithm were provided to construct the DS. In these algorithms, they employ the distributed leader election algorithm [16] to construct a rooted spanning tree. Then, a labeling strategy is used to divide the nodes in the tree to be either black or gray according to their ranks (pair of its level and its id). The labeling process begins from the root node and finishes at the leaves. At first, the node with the lowest rank marks itself black and broadcasts a DOMINATOR message. According to the following rules, the marking process continues:

- “If the first message that a node receives is a DOMINATOR message, it marks itself gray and broadcasts a DOMINATEE message.”[3]
- “If a node received DOMINATEE messages from all its lower rank neighbors, it marks itself black and sends a dominator message.”[3]

When it reaches the leaf nodes, the marking process finishes. Just now, the set of black nodes form an MIS. In the final phase, the nodes connect in the MIS to form a CDS through INVITE and JOIN messages. Figure 2 illustrates the operation of these algorithms. Node 0 is the root of the spanning tree so that it is constructed through using the leader election algorithm. This algorithm has time complexity of $O(n)$ and message complexity of $O(n.\log(n))$. [3]

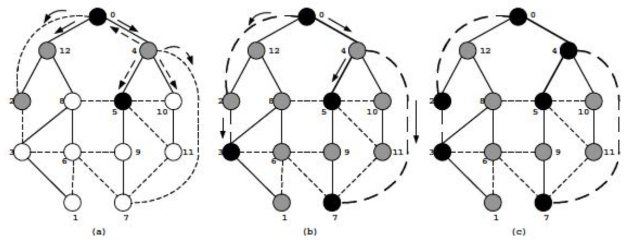

Figure 2. An example of Alzoubi and Wan’s algorithm [3]

A completely localized algorithm was proposed to construct CDS in general graphs. At first, all vertices are unmarked. They exchange their open neighborhood information with their one-hop neighbors. Each node knows all of its two-hop neighbors. The marking process applies the following simple rule: any vertex having two unconnected neighbors so that they were marked as a dominator. At last, the set of marked vertices form a CDS, but it has a lot of redundant nodes. There are two pruning principles so that they are provided to post-process the DS. This pruning idea was expressed to the following general rule [10]. According to this rule, if it exist k connected neighbors with higher ids in S so that it can cover all u’s neighbors then, a node u can be removed from S. [23]

Connected Dominating Set-Hierarchical Graph (CDS-HG) is a novel distributed MCDS approximation algorithm. This algorithm generates smaller CDS sizes compared with the existing algorithms. Algorithm includes of two phases. In the first phase, rule1 (Essential Node Determination) is used. According to this rule, a set of dominators select for each hierarchical level so that all nodes in the next level are dominated by these dominators. A greedy strategy is used to select the dominators for creating a small initial DS. In the second phase, rule2 is used to remove the redundant dominators. This process repeated from the lowest level to the highest level of the hierarchical graph. According to The greedy strategy that created CDS is connected. The size of generated CDS is at most $(\log n|opt|)$, where n is the number of nodes in the network and opt is the cardinality of a minimum DS. The computation complexity of their algorithm is $O(n^2)$. [25]

Because a centralized CDS algorithm is impractical for WSNs, they implemented a distributed algorithm based on competition. It includes three phases: creating the initial CDS through competition and reducing the CDS size through applying rule2 on all dominators. Respectively, the computation and message complexities of their algorithm are $O(\theta^2)$ and $O(\theta)$, where θ is the maximum number of child nodes in graph. [25]

Another algorithm is proposed for finding MCDS by using DS. DSs are connected via Steiner tree. The approximation algorithm includes of three stages. In the first stage, the DS is determined through identifying the maximum degree nodes to discover the highest cover nodes. In the second stage, connects the nodes in the DS through a Steiner tree. In third stage, this tree prunes to form the MCDS. To local repair, rule k [17] is used to find the nodes so that can maintain the MCDS. Eventually in the pruning phase, redundant nodes are deleted from the CDS to obtain the MCDS. They proposed a local repair algorithm to take care of node's deletion. [20]

Approximation Two Independent Sets based Algorithm (ATISA) is a new method for constructing CDS. The ATISA has three stages: (1) constructing a connected set (CS), (2) constructing a CDS, and (3) pruning the redundant dominators of CDS. ATISA constructs the CDS with the smallest size compared with some famous CDS construction algorithms. The message complexity of this algorithm is O(n).

The ATISA has two kinds of implementations: centralized implementation and distributed implementation. The centralized algorithm consists of three stages, which are CS construction stage, CDS construction stage, and pruning stage. In the centralized algorithm, the initial node is selected randomly and then, the algorithm executed several rounds. When the first stage is ended, there are no black nodes generated in the network. The generated black node set is formed a connected set. If a white node has black neighbors, then it will select the black neighbor with the minimum id as its dominator, and also change its state into gray. If a white node only has the gray neighbors; then, it will send an invite message to the gray neighbor with the minimum id and also change its state into gray. Finally, in the second stage, constructs a CDS and all the nodes are either black or gray. At last, there is no white node left in the network. According to the third stage, if a black node with no children and also if the neighbors of the black node are all adjacent to at least two black nodes, then the black node is put into connected set. [19]

But, in the distributed implementation, all the nodes exchange their positions information with their neighbors. At first, all nodes are initialized white. After the first stage, there are white nodes, gray nodes, and black nodes. Then, in the second stage, there are black nodes, gray nodes and sometimes white nodes. According to the first stage, white nodes can change their states into gray and also gray nodes can change their states into black. At last, in the third stage, the redundant black nodes are deleted. [13]

Energy-Aware Virtual Backbone Tree (EVBT) is a distributed algorithm for constructing a backbone in WSN. It chooses only nodes with enough energy levels as the member of the virtual backbone. Also, it introduced a concept of threshold energy level for members of virtual backbone. Only nodes with energy levels above a predefined threshold are included in the EVBT. The EVBT can be dynamically reconstructed with changing energy levels and also changing state (on/off) of nodes. Data packet can be delivered along another EVBT, when an EVBT breaks down due depletion of energy of one or more members. All sensor nodes are fixed but, the SN is static. They used a simple graph G(V, E) to represent a WSN, where V and E represents set of all sensor node and all edges, respectively. The graph will be an undirected graph. Hence, sensor node that does not belong to the backbone is termed as *leaf node*. Every node in the network has an EVBT node. They term this EVBT node as the dominator of the corresponding leaf node. They presumed each node v knows its N (v). They check two types of vertices. A tree node is a *fixed vertex* so that it cannot be removed from the EVBT. It means that this vertex will be a part of the final solution. If energy level of *Non-fixed* vertices is not above threshold energy level or its removal does not disjoin the resulting sub graph, then *Non-fixed* vertices will be removed. Therefore, at each step of the algorithm, at least one vertex is either fixed, or removed. It is presumed that at first, all the nodes in the network form the EVBT. At last, these non-removed and fixed vertices form the EVBT. They presumed, the sink node is leader to starts execution of algorithm.

At first, the leader will check its degree. If the degree is greater than one, then it verifies whether removing itself from the graph would disjoin the sub graph. Keeping this in view, criteria for being a member of EVBT are the node must have energy level greater than the threshold energy level, and also highest degree among all the neighbors of the node. When the algorithm terminated that result of iteration is an empty set of each node. At the first iteration, this list is empty. The EVBT computed at the end of all iterations. It at once updates its list of dominators, ever when a node chooses any node as its dominator. In this algorithm, every node in the network has one virtual backbone node, which it selects as its dominator. This dominator will be parent node for that node. Any node in the network will forward its packet to its dominator. In this way the packet eventually reaches the sink node. [1]

A CDS-based backbone was constructed to support the operation of an energy efficient network. That focused on three key ideas in their design: (1) a realistic weight matrix, (2) an asymmetric communication link between pairs of nodes, and (3) a role switching technique to prolong the lifetime of the CDS backbone. This algorithm is distributed in nature. It is deterministic.

Corresponding with the weight comparison among neighbors, some suitable nodes get selected as dominators. The set of dominators is a MIS. Those selected dominators are in conjunction with some Connector nodes (dominator2 nodes), then they form the dominating set of the network. Nodes that are not part of the dominating set remain as dominates and use neighboring dominators as next hops for data communication. This algorithm presumed that all nodes know two hops away neighborhood information and they have equal transmission range. Therefore, the weight matrix used in r-CDS algorithm is: $W_i(r_i, deg_i, id_i)$. Node i is more suitable to be a dominator than neighboring node j, if any of the following is true: [15]

deg (u)- The effective node degree of node u

r(u)- The number of 2-hop away neighbors

- $r(i) < r(j)$
- $r(i) = r(j)$ and $deg(i) > deg(j)$

- *r(i) = r(j) and deg(i) = deg(j) and id (i) < id (j)*

According to this algorithm, sensor nodes in the r-CDS algorithm can have three different colors: white, gray and black. At first, all nodes are white. In continue, all nodes change their color to either black or gray. Black nodes form network backbone, but gray nodes remain as dominatees. In their algorithm, nodes can broadcast the following messages: BLACK, GRAY and d(u) messages. After each node knows about its two hop away neighborhood, all nodes broadcast their r values. A node u can become dominator1, if it wins in the weight comparison. Then, node u turns black and broadcasts a BLACK message in the neighborhood. If a white node v receives BLACK message from its neighbor u, so v becomes gray and broadcasts GRAY message. This GRAY message includes the pair (v' s id, u' s id). If a black node w receives GRAY message from a gray node v and also the id of another black node u, and if w and u are not connected yet, then v becomes dominator2 node to connect u and w. In that case, after receiving a BLACK message from a node w, if a gray node u has already received a notification so that there is a two hop away black neighbor v sent through a neighbor x and v has not been connected to w yet, then both u and x become dominator2 nodes to connect node v and node w. [15]

An intelligent backbone formation algorithm was proposed according to distributed learning automata. The worst case running time and message complexity of the backbone formation algorithm has a *1/(1-ε)* optimal size backbone. This was why that it was shown that through a proper choice of the learning rate of the algorithm, a trade-off between the running time and message complexity of algorithm with the backbone size can be made. [2]

At its implementation, a network of the learning automata isomorphic to the UDG was used. It is formed through equipping each host to a learning automaton. At each stage of this approach, the learning automata randomly choose one of their actions so that a solution can be found in the CDS problem. The created CDS is evaluated via the random environment and also the action probability vectors of the learning automata are updated depending on the response received from the their environment. At last, in an iterative process, the learning automata converge to a common policy so that it constructs a minimum size virtual backbone for us. The network graph is presumed to be undirected. Each host has a unique id and also requires that know its neighbors' id. With comparing the results of proposed algorithm with the other of the best known CDS-based backbone formation algorithms, the results show that their algorithm always outperforms the others in terms of the backbone size and also its message overhead is only a few more than the least cost algorithm. [2]

D. Hybrid Algorithms

Several backbone formation algorithms have been created so that they used from two or more categories such as clustering and CDS. We call their as *Hybrid* Algorithms. At first, these algorithms use clustering and then CDS. In blew some of algorithms have been shown.

One algorithm was proposed for constructing virtual backbone in Wireless Ad-hoc Sensor Networks. According to this algorithm, the sensor network is divided into clusters. This algorithm includes of two phases: (1) clustering nodes, (2) the CDS algorithm for intra clusters. It assumes all vertices are unmarked. Then, exchange their open neighborhood information with their one-hop neighbors. With using two pruning rules are provided to post-process the DS. If there exists a node v with higher id so that the closed neighbor set of u is a subset of the closed neighbor set of v, node u can be taken out from the CDS. [4]

Clique Clustering (CC) is the definition of a protocol for building and maintaining a connected backbone in WSN. In this protocol, the network is partitioned into clusters that are cliques. Thus, removing a node does not disjoin a cluster, and adding one needs simple operations for checking node acceptance to the cluster. The protocol includes three phases: (1) partitioning the network into clusters as *cliques,* (2) connection Clusters to form a backbone, (3) maintains the backbone connected. The cluster formation phase of the CC protocol produces a clustering that includes the following properties: (1) every non-cluster-head node has at least a cluster-head (2) every node in a cluster can communicate directly with every other node in the cluster, and (3) every non-cluster-head node affiliates to the cluster of the first cluster-head inviting it. In their opinion, every node knew its own unique id, its own weight and also the id and weight of each of its neighbors. [5]

The protocol is started through nodes that have the biggest weight among all their neighbors. These nodes send a message so that they will be cluster-heads. Upon receiving this message from one of its heavier neighbors, a node exchanges with the sender information. According to the received information, a cluster-head selects all smaller neighbors that can be affiliated to its own cluster so that maintaining the clique property and invites them to join it. A node decides to be a cluster-head itself, when whose heavier neighbors have joined other clusters or have finished inviting nodes and also that has not been invited to be part of any cluster. When the protocol terminates that every node belongs to a cluster being either a cluster-head or an ordinary node and also knows the role and cluster-head of all its neighbors. At last, to build these cluster connections, each cluster-head needs to know all its *neighboring cluster-heads*. With terminating the cluster formation phase, every node knows the id and weight of each neighbor and also the id and the weight of the cluster-head to which each neighbor is affiliated. Then, each node sending this information to its own cluster-head to select paths for a connected backbone. Figure 3 illustrates the final connected backbone. [5]

 ISBN: 978-1-61208-166-3

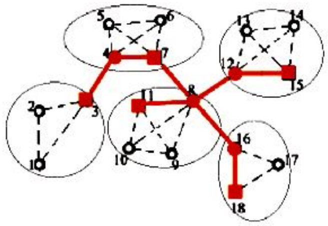

Figure 3. A WSN, the CC-induced clustering and a backbone connecting the cluster [5]

III. COMPARISON OF SOME ALGORITHMS

We have surveyed some well-known backbone formation algorithms in term of time and message complexity. Performance comparison of some algorithms is shown in the table below. We can see that proposed algorithms in [3], [11], [20], [22] have the less time and proposed algorithms in [11], [19], [20], [25] have the less message complexity among other algorithms in this table.

Also, time complexity of proposed algorithms in [13], [25], and message complexity of proposed algorithms in [3], [22] are equal. According to the table below, time and message complexity [2] is only slightly more than the least cost algorithm.

TABLE I. PERFORMANCE COMPARISON

Ref.	**Performance comparison**		
	Approximation factor	***Time complexity***	***Message complexity***
[2]	-	$O(\Delta)$	$O(n\Delta^2)$
[3]	*8opt+1*	$O(n)$	$O(n \log (n))$
[9]	-	$O(n \log^3 (n))$	$O(n^2 \log^3 (n))$
[11]	*147opt+33*	$O(n)$	$O(n)$
[12]-I	$2H(\Delta)+1$	$O((n+\lvert C\rvert)\Delta)$	$O((n\lvert C\rvert + m + n \log(n))$
[12]-II	$2H(\Delta)$	$O(\lvert C\rvert(\Delta+\lvert C\rvert))$	$O(n\lvert C\rvert)$
[13]	$O(n \log(n))$	$O(n^2)$	$O(n^2)$
[19]	-	-	$O(n)$
[20]	$O(n)$	$O(n)$	$O(n)$
[22]	*8opt*	$O(n)$	$O(n \log (n))$
[23]	$O(n)$	$O(\Delta^3)$	$\Theta(m)$
[25]	-	$O(n^2)$	$O(n)$

(n and m are the number of vertices and edges respectively, *opt* is the size of MCDS, Δ is the maximum degree, $\lvert C\rvert$ is the size of the computed CDS, H is the harmonic function.)

IV. CONCLUSIONS AND FUTURE WORKS

The backbone has proven to be an effective construct within which to solve a variety of problems that arise in WSNs. In this paper, we classified backbone formation algorithms and a few instances of these classifications and proposed hybrid approaches of these classifications. Also, we have surveyed some famous backbone formation algorithms in term of time and message complexity. Significant attention has been paid to backbone formation algorithms yielding a large number of publications. Backbone construction depends on the task to be carried. A backbone reduces the communication overhead, increases the bandwidth efficiency, decreases the overall energy consumption and at last increases network effective lifetime in a WSN. The important issue that we can be reached is selection algorithm according to our use.

ACKNOWLEDGEMENT

We would like to thank the reviewers who helped us to improve the quality of the current paper.

REFERENCES

[1] T. Acharya, S. Chattopadhyay, and R. Roy, “Energy-Aware Virtual Backbone Tree for Efficient Routing in Wireless Sensor Networks,” in Proc. of Int. Conf. on Networking and Services, (ICNS '07), IEEE, pp. 96-102, Athens, Greece, June 19, 2007.

[2] J. Akbari Torkestani, M. R. Meybodi, “An intelligent backbone formation algorithm for wireless ad networks based on distributed learning automata,” Computer Networks 54, pp. 826–843, 2010.

[3] K. M. Alzoubi, P. J. Wan, and O. Frieder, “New Distributed Algorithm Connected Dominating Set in Wireless Ad hoc Networks,” Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Vol. 9, pp. 297-304, 7 January 2002.

[4] R. Azarderakhsh, A. H. Jahangir, and M. Keshtgary, “A New Virtual Backbone for Wireless Ad-hoc Sensor Network with Connected Dominating Set,” Third Annual Conference on Wireless On-demand Network Systems and Services (WONS), pp. 191-195, 2006.

[5] S. Basagni, R. Petroccia, and Ch. Petrioli, “Efficiently reconfigurable backbones for wireless sensor networks,” Computer Communications, Vol. 31, Issue 4, pp. 668-698, 5 March 2008.

[6] S. Basagni, M. Elia, and R. Ghosh, “ViBES: virtual backbone for energy saving in wireless sensor networks,” Military Communications Conference (MILCOM), IEEE Press, Vol. 3, pp. 1240–1246, 31 October, , 2004.

[7] S. Basagni, “Distributed clustering for ad hoc networks, ” Proceedings.IEEE, Fourth International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN '99), pp. 310 -315 , Australia, 23 Jun 1999.

[8] S. Basagni, A. Carosi, and Ch. Petrioli, “Sensor-DMAC: Dynamic Topology Control for Wireless Sensor Networks,” In Proceedings of IEEE VTC 2004 Fall, Vol. 4, pp. 2930-2935, Los Angeles, CA, 26 September 2004.

[9] S. Butenko, X. Cheng, and Carlos A. S Oliveira, and P. M. Pardalos, “A New Heuristic For The Minimum Connected Dominating Set Problem On Ad Hoc Wireless Networks,” Recent Developments in Cooperative Control and Optimization, pp. 61-73, Kluwer Academic Publishers, 2004.

[10] I. Cidon, O. Mokryn, “Propagation and Leader Election in Multihop Broadcast Environment,” Proc. 12th Int. symp. Disrt. Computing, pp. 104–119, Greece, September 1998.

[11] X. Cheng, M. Ding, and D. Chen, “An approximation algorithm for connected dominating set in ad hoc networks,” Proc. of International Workshop on Theoretical Aspects of Wireless Ad Hoc, Sensor, and Peer-to-Peer Networks (TAWN), 2004.

[12] B. Das, V. Bharghavan, “Routing in Ad-Hoc Networks Using Minimum Connected Dominating Sets *International Conference on Communications*. IEEE Int. Conf. Communications (ICC 97), Vol. 1, pp. 376-380, Montreal, Canada , June 1997.

[13] S. Guha and S. Khuller, “Approximation algorithms for connected dominating sets,“ Algorithmica, 20(4), pp. 374-387, April 1998.

[14] W. R. Heinzelman, A. Chandrakasan, and H. BalakrishnanD, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks,” Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS), Vol. 8, pp. 10-20, January 2000.

[15] S. Hussain, M. I. Shafique, and L. T. Yang, “Constructing a CDS-Based Network Backbone for Energy Efficiency in Industrial Wireless Sensor Network,” In Proceedings of HPCC, pp. 322-328, 2010.

[16] K. Islam, S. G. Akl, and H. Meiher, “A constant Factor Localized Algorithm for Computing Connected Dominating Sets in Wireless Sensor Networks,” Proc of 14th IEEE International Conference on Parallel and Distributed Systems, (ICPADS), pp. 559-566, Melbourne, VIC, December 2008.

[17] B. Jeremy, D. Min, and T. Andrew and C. Xiuzhen, “Connected Dominating Set in Sensor Networks and MANETs,” Handbook of Combinatorial Optimization, Springer, US, 2004.

[18] C.R. Lin, M. Gerla, “Adaptive clustering for mobile wireless network, ” IEEE J Sel Areas Commun, Vol. 15, Issue 7, pp. 1265–1275, September 1997.

[19] Z. Liu, B. Wang, and Q. Tang, “Approximation Two Independent Sets Based Connected Dominating Set Construction Algorithm for Wireless Sensor Networks,” Inform. Technol. J., Vol. 9, Issue 5, pp. 864-876, 2010.

[20] M. Rai, Sh. Verma, and Sh. Tapaswi, “A Power Aware Minimum Connected Dominating Set for Wireless Sensor Networks,” Journal of networks, Vol. 4, no. 6, August 2009.

[21] M.T. Thai, W. Feng, and L. Dan, and Z. Shiwei and D. Ding-Zhu, “Connected dominating sets in wireless networks with different transmission ranges,” IEEE Trans. Mobile Comput. , Vol. 6, pp. 721-730, 2007.

[22] P.J. Wan, K.M. Alzoubi and O. Frieder, “Distributed construction of connected dominating set in wireless ad hoc networks,” Proc. of IEEE Conf. Computer and Communications Societies, pp. 1597-1604, New York, June 23-27, 2002.

[23] J. Wu, H. Li, “On calculating connected dominating set for efficient routing in ad hoc wireless networks,” Proc. of ACMDIALM’1999, pp. 7–14, August 1999.

[24] D. Xia, N. Vlajic, “Near-Optimal Node Clustering in Wireless Sensor Networks for Environment Monitoring,” In Proceedings of CCECE, pp.1825 – 1829, 2006 .

[25] R. Xie, D. Qi1, and Y. Li, and J. Z. Wang, “A novel distributed MCDS approximation algorithm for wireless sensor networks,” Mobile & Wireless Communications, Vol. 9, Issue 3, pp. 427–437, March 2009.

[26] Y. Xu, J. Heidemann, and D. Estrin, “Geography-informed Energy Conservation for Ad Hoc routing ,” In Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom), pp. 70-84, Rome, Italy, July 16-21, 2001.