Ann. Telecommun. (2017) 72:199–208
DOI 10.1007/s12243-017-0563-x

Virtual coordinate system using dominating set for GPS-free adhoc networks

Shailendra Shukla[1] · **Rajiv Misra**[2] · **Abhishek Agarwal**[2]

Received: 12 January 2016 / Accepted: 20 January 2017 / Published online: 8 February 2017

Abstract Reported work on virtual coordinate assignment (VCA) schemes are iterative-based techniques which rely upon geometric projection (i.e., projecting on circle) or embedding of network topology to low-dimensional Euclidean space (like graph embedding, multidimensional scaling). The performance of existing VCA techniques is constrained by topological situations such as low density or having voids/holes, where greedy forwarding suffers due to local minima when no neighbor is found closer to the destination or low-quality routes comprised of long distance hops. Another drawback of existing VCA techniques is the requirement of thousand iterations for usable coordinate convergence. In order to overcome these drawbacks, we propose a novel virtual coordinate construction technique using graph-theoretic dominating sets. Dominating set (DS) of G is a subset of vertices such that each vertex in G is either in DS or has a neighbor in DS. We found that our virtual coordinate assignment using dominating set algorithm has an approximation ratio $((4.8+\ln 5)opt+1.2)$, where opt is the minimum size dominating set which has the same approximation ratio as minimum dominating set problem. Our algorithm has time complexity $\mathcal{O}(n)$ times and $\mathcal{O}(D)$ rounds and message complexity is $\mathcal{O}(n \log n)$, where D is the radius and n is the number of nodes in networks.

Keywords Wireless sensor networks · Virtual coordinates assignment and Dominating set

1 Introduction

Wireless sensor network (WSN) as a subpart of Internet of Thing (IoT) [1] has been regarded as a promising data acquisition tool, which has been gradually used in the national defense field, academia, industries, and for future smart cities. The WSN uses sensing devices to accomplish smart identification, surveillance, target tracking, and localization. Connectivity and coverage are two most important aspects of WSNs for applications like target monitoring, localization, and surveillance [2, 3]. One of the solutions to achieve connectivity and coverage of networks at minimal cost is to use geographic location [4]. Geographic location systems have drawbacks, and to address these drawbacks, virtual coordinate systems are proposed where each node is assigned with some coordinate which do not project the actual location or geographic coordinates but reflect a relative position of nodes (conforms to connectivity relationship between the nodes).

Virtual coordinate assignments use techniques like graph embedding and multidimensional scaling. Graph embedding is a graph-theoretic technique in which network topology is mapped into a Euclidean space (NoGeo [4], GSpring [5], GEM [6]). Multidimensional scaling (MDS) is used to embed the hop distance metric into a Euclidean space. MDS technique transforms the hop distance matrix to find out the eigen decomposition (TAR [7]). Virtual coordinates find its

✉ Shailendra Shukla
sshukla2000@gmail.com

Rajiv Misra
rajivm@iitp.ac.in

Abhishek Agarwal
abhishek0agarwal@gmail.com

[1] Department of Computer Science and Engineering, Jaypee University of Information Technology, Shimla, India

[2] Department of Computer Science and Engineering, Indian Institute of Technology, Patna, India

application in building network overlays (geographic hash tables (GHTs) to support P2P routing protocols.

The major problem in the reported schemes like NoGeo [4] and GSpring [5] is their slow convergence and a large number of iterations. Another problem of these schemes is that it does not work for sparsely connected networks because sparse networks have holes/voids in topology, which leads to the dead end in the network. Moreover, the greedy forwarding fails in geographic routing (based on virtual coordinates) as the geometric distance in sparse network rarely reflects the true hop distance between the nodes. Therefore, existing virtual coordinate protocols assume fully connected network, highly dense network graph with an average degree of around 16 neighbors (NoGeo [4]), GEM [6], and GSpring [5]. However, these existing schemes report low success rate at lower density with six to eight neighbors (i.e., sparse network). At low density, the hop count becomes high causing low success rate for greedy forwarding.

Thus, the challenge remains to design an efficient virtual coordinate assignment scheme which can also work for sparsely connected network (low density) so that geographic routing can achieve higher success rate with lower hop count.

The paper is organized as Section 2 discusses the related work. Section 3 presents the problem definition and contribution to the paper. In Section 4, we present our proposed algorithm. In Section 5, we present an analysis of proposed algorithm. In Section 6, we perform a simulation of the proposed algorithm and Section 7 concludes our paper.

2 Related work

In this section, we present various reported schemes of virtual coordinate assignment (VCA) in ad hoc and sensor networks. Karp et al. (2000) have earlier proposed the GPSR [8] algorithm which assumes that all the nodes knows their position using either GPS or by any other means.

Rao et al. 2003 in their work proposed coordinate assignment algorithm for geographic routing application without using no-location information (NoGeo [4]) in ad hoc networks. NoGeo assumes a scenario where neither perimeter nor its location is known. NoGeo also assumes two designated nodes as beacon nodes. Non-beacon nodes determine if they are perimeter nodes in distributed manner using their hop distance from the beacons. Each node uses the perimeter distance and computes normalized coordinate of itself and perimeter nodes. Then perimeter nodes are projected onto an imaginary circle and nodes using a relaxation algorithm that iteratively determines their virtual coordinates. The main drawbacks of this approach are the requirement of a large number of anchor nodes and considering the large dimensions. The drawback of having many dimensions is that forming coordinates requires a long time to converge. The mobility support requires a bootstrap node to broadcast the network continuously.

In greedy virtual coordinates (GSpring) (2007) [5], to identify nodes which are at the edges of the networks calls a perimeter detection algorithm and assign coordinates using a system of springs and repulsion forces. A method for providing virtual coordinates to boundary nodes begins with the detection of k boundary nodes in the network graph. The k boundary nodes are arranged in the virtual circle and given some reasonable starting coordinates. Once the cyclical ordering of the boundary nodes $n_1, n_2, ..., n_k$ completes, the algorithm determines the number of hops on the boundary of the network graph by summing the hop counts between adjacent nodes to compute the coordinates for k perimeter nodes. All the perimeter nodes are aware of the hop matrix of each other since it is propagated by broadcast. Each perimeter node knows its hop count from each perimeter node, so a node can thus determine that it is on the shortest path between one pair of a perimeter. When this algorithm terminates, a small number of perimeter nodes at the boundary of the network and some nodes in the middle of the network will derived a set of initial coordinates. GSpring was developed for network of non-mobile nodes. The assignment of coordinates in a highly mobile environment without geographic location was not discussed.

Aligned virtual coordinate system (AVCS) 2008 [9] address the issue of quantization error (VCS forwarding void) in geographic routing around voids. Again, the drawback is a long time to converge forming virtual coordinates due to many dimensions resulting from a large size of n. Again, the possible dead ends exist resulting in failure of a greedy forwarding algorithm.

Convex partition, (2009) [10], paper proposed a distributed convex partition protocol that divides the field to subareas with convex polygonal subarea shape and to let every node know which partition(s) it belongs to. The convex partition protocol requires the boundary nodes to be identified first. After running such an algorithm, each hole is assigned a unique ID, and each boundary node that belongs to a certain hole is tagged with the hole ID. The rest of the protocol works as, initially, it identifies the critical points then a bisector-induced convex partitioning is done and finally partitions were recognized. All these operations are performed in a distributed way. But their approach requires a base station for initiating each phase (sends a command to the network). Its drawback is it requires excessive message transmission boundary node identification phase. This phase incurs $O(n)$ message overhead. Critical point identification phase requires $O(n) + O(n)$ message transmission

Table 1 Notation used in algorithms

CN	$\triangleq$	Central node of network
BN	$\triangleq$	Set of boundary nodes
WN	$\triangleq$	Witness node
HN	$\triangleq$	Holder node
IN	$\triangleq$	Independent node
DS	$\triangleq$	Dominating set
$Nb_k(x)$	$\triangleq$	Set of k-hop distance neighbors of node x
N_i $(N_1, N_2...)$	$\triangleq$	Neighbors of node WN
$MNb(x)$	$\triangleq$	Neighbors of node x marked by color <gray>
$UNb(x)$	$\triangleq$	Neighbors nodes of node x marked by color <white>
$MinID(S)$	$\triangleq$	Nodes which have minimum ID in set S
(x_{coor}, y_{coor})	$\triangleq$	Nirtual coordinate derived by proposed algorithm
S_i	$\triangleq$	ith element in set S
ρ	$\triangleq$	Radius of networks

on average. Bisector path identification phase and partition recognition phase requires $O(n \times n_{cri})$, where n_{cri} is the number of concave critical points.

Virtual cord protocol (VCP) (2011) [11] uses hash function to create virtual cord (value in predefined range $[S, E]$) and each node maintains a part of the entire range. Depending on the old position, node gets a new position between the end value and its successor or predecessor. The new node becomes predecessor of the old node if it received position S. Otherwise, it becomes its successor. Finally, locally available neighborhood information is used for greedy routing toward the destination.

Paper [12] investigates unidirectional links in wireless network, develops virtual coordinate assignment protocol ($ABVCap\_Uni$), and supports routing in sensor networks with unidirectional links. Each node is assumed to be static and has a unique identifier (ID). Four anchors and one sink node is selected to assign virtual coordinates. Virtual coordinate is assigned considering eight entries longitude, latitude, ripple, up, down, ring initiator, ring number, and ring order. The longitude and latitude coordinates denote the location of the node, and the other coordinates are used for packet forwarding.

Topology aware routing (TAR) 2012 [7] protocol encodes a network topology to a low-dimensional virtual coordinate area where hop distances between the nodes are maintained. TAR proposes two methodology for scaling MDS in a centralized and distributed manner. In centralized MDS, base station floods the network for gathering connectivity information and for that it uses multipoint relay (which covers the networks with few number of broadcasting) [7]. Once topology is known then the hop distance between pairwise nodes was computed using the shortest-path algorithm. Based on the pairwise node distance, base station applies MDS to embed the network topology into a Euclidean space, which assigns the virtual coordinate. Base station send the virtual coordinate to 1 hop neighbors through multiple relays. In distributed MDS, a network of N nodes randomly selects M nodes as anchors. Anchor nodes broadcast the network with beacon message set as hop count = 0. Based on the received messages, node i constructs its N-dimensional hop distance matrix as $x_i = [x_{i1}, x_{i2}, ...x_{iN}]$ where x_{ij} is the hop distance from node i to j. Each anchor nodes send its hop distance vector to the base station and the base station constructs an anchor distance matrix $X = [x_1, x_2, x_3....x_n]$ then the MDS is used to embed the hop distance metric space to a low-dimensional Euclidean space such that each anchor is assigned a virtual coordinate of m-dimension such that $m << M$. Each nonanchor node calculates its virtual coordinate by itself using the least square fitting method. Centralized MDS incurs higher overhead for detecting the entire network topology because it requires number of broadcastings. Distributed MDS randomly select M anchors to broadcast the network. Both the methods require a designated base station.

Reported works on a virtual coordinate assignment algorithm use thousands of iterations and depend upon geometric techniques, i.e., projecting on circle or embedding network's multiple dimensions on Euclidean space.

3 Problem definition and contribution

3.1 System model

Consider a geographical region in which a large number of wireless nodes are deployed. All the nodes are equipped with an omnidirectional antenna and they are unaware of their position coordinates. Given a graph $G = (V, E)$, where nodes of G are placed in two-dimensional space in such a way that nodes are unaware of their position.

3.2 Problem definition

The virtual coordinate assignment problem that satisfies all of the unit disk graph (UDG) constraints is NP-hard. Hence, the problem statement of this work is to develop a distributed virtual coordinate assignment scheme that uses graph-theoretic dominating sets with guaranteed approximation factor to the size of any optimal dominating set. When virtual coordinate assignment using dominating set algorithm is used with geographic routing, for low density networks, it achieves higher success rate and lower hop count.

3.3 Contribution

Our main contribution through this paper is to design an efficient virtual coordinate assignment scheme which can work for sparsely connected network (low density) so that geographic routing can achieve a higher success rate with a lower hop count.

3.4 Definitions

We describe our network model as a graph $G = (V, E)$, where V is vertices (node) $V = \{v_1, v_2, ..., v_n\}$ in a network. Communication link between any two nodes v_i and v_j is bidirectional in nature. We assumed that the nodes are homogeneously spread and they are equipped with omnidirectional antenna with radio range r. The notations and symbols used in the paper is defined in Table 1.

Definition 1 k-hop dominating set (k-DS): k-hop dominating set of a graph $G = (V, E)$ is a subset k-DS of nodes such that every vertex in $V(G)$ is at distance atmost k from k-DS.

Definition 2 Radius($\rho(G)$): Radius of G is define as the minimum eccentricity of any node in graph. The eccentricity of a given node is the maximum distance of any node.

Definition 3 Witness nodes (WN): These nodes are the common nodes between two dominating sets, $WN = Nb_k(N_2) \cap Nb_k(N_1)$.

Witness nodes WN are identified by merger of dominating set pair DS.

Definition 4 Holder nodes (HN): Holder nodes are the witness nodes or boundary nodes with minimum ID.

Definition 5 Independent node (IN): A subset $IN \subseteq V$ is called an independent node, if for every two vertices $\{u, v\} \in IN$, an edge does not exist $\{u, v\} \notin E$ [13].

4 DVC: distributed virtual coordinate assignment algorithm

4.1 Description of the algorithm

In this section, we give DVC algorithm for virtual coordinate assignment for UDG. Our virtual coordinate algorithm works in three phases. In the first phase, our algorithm identifies center node CN and radius (ρ) of networks. In the second phase, it assigns a virtual coordinates to all 1-hop neighbors of the central node and in the last phase, virtual coordinate assignment is done from 1-hop neighbors of CN to boundary nodes BN.

Algorithm 1 Find node at center of network

Require: Node knows their 1-hop neighbors Nb_i and 1-dominating set (1-DS).

Ensure: Central nodes CN.

1: Detect k-hop dominating set and mark them.
2: Form pairs of DS nodes (using min-hop distance metric). In case of single last node, make its pair with any of its k-hop neighbor.
3: Find out common neighbor nodes of DS-pair (say witness nodes(WN)):
   1. Node N_1 (with smaller id), send message M to other node N_2 (in pair) with the ids of $Nb_k(N_1)$.
   2. Node N_2 finds out witness nodes : $WN = Nb_k(N_2) \cap Nb_k(N_1)$.
4: All witness nodes WN check itself for boundary nodes (BN_i = true), if nodes is at boundary then the witness nodes (WN) which are at boundary nodes send message M_1 to N_2. Message M indicate itself as boundary nodes.
5: N_2 identifies holder node HN.
6: **if** ($WN == null$) **then**
7: $HN = N_1$
8: **if** ($WN - BN == null$) **then**
9: $HN = MinID(BN)$
10: else $HN = MinID(WN - BN)$
11: **end if**
12: **end if**
13: $k = k \ll 1$ *(k multiplied by 2)*
14: **while** ($|DS| > 1$) **do**
15: Central node $CN = DS$ assuming that it always converges to single CN and CN knows the boundary nodes of network mark the k-hop dominating set on the graph $S \cup \in G(V, E)$ for the pair of dominating set makes its *k-hop* neighbor.
16: **end while**

4.2 Algorithm for center node detection in network

Algorithm for center node detection assumes that nodes know their neighbor. Algorithm initiates with the detection of k-hop dominating set k-DS. Each dominator node set identifies a pair in DS using the shortest path. These dominator pairs compute 2-hop DS. After k iterations, it finds k-hop DS. After k = radius (G), rounds of the algorithm terminate with identified center node (CN) of graph. Algorithm 1 gives the detailed description of working.

Witness nodes are identified by the intersection of dominating set pairs. All the witness nodes test themselves for boundary nodes (ref algo. 1) and send message (declaring itself as boundary nodes) to one of its neighbor node (lets N_2). Once witness nodes are declared, neighbor node (N_2) detects holder node HN. If the witness nodes is empty then the holder nodes are dominating set. If the boundary nodes are submitted from the witness node and it comes to null then the holder nodes are all boundary nodes with min(ID). If none of the above condition holds then the holder nodes are minimum id node from set (witness node − boundary nodes). Minimum hop distance between CN and BN is the radius (ρ) of graph. Our algorithm converges to single node and that will be the CN of our graph.

4.3 Algorithm for 1-hop coordinate assignment of center node

Algorithm 1 assigns virtual coordinates to 1-hop nodes of CN. Algorithm 1 requires independent nodes for coordinate computation. Independent nodes are defined as any two vertices of a graph $G = (V, E)$ which are not adjacent to one another; as a consequence, the subgraph $G[IN']$ induced by an independent set (IS) contains no edges.

Theorem 1 *IS be the set independent nodes such that $IN \in CN$ but they are not in the scope of other independent nodes IN.*

Proof Two IN independent nodes are not in communication range of each other by definition 1.

Center node (CN) assigns its coordinate as (0,0) and calculate angle $\beta = \frac{360}{IN}$, β is assigned coordinate of anchor nodes. Two anchor nodes b_1 and b_2 ($\{b_1, b_2\} \in BN$) are randomly selected from a set of boundary nodes. Independent nodes are selected in such a way that their hop distances from b_1 and b_2 are aligned in even and odd manner. Even hop distance and odd hop distance between nodes b_1 and b_2 are used to arrange the independent node in order. Rotation between b_1 and b_2 increases the sequence number in ascending order. Independent nodes assign themselves $(\cos(\mu * n)), \sin(\mu * n))$ and mark themselves $gray$. The parameter μ assignment is discussed in the correctness section. Hop count and rotation is used to allocate the coordinates to independent nodes IN as $(\cos(72{*}n), \sin(72{*}n))$. Non-independent nodes assign their coordinates as $\alpha = \frac{(\alpha_i + \alpha_j)}{2}$. Nodes (i, j) are at minimum angle with respect to center node CN node with $(\cos\alpha, \sin\alpha)$ coordinates. Algorithm 1 establishes circulars coordinate (neighbor node) of center node as $Nb_1(CN) = [\sin\alpha + \cos\alpha]$ and angular coordinates as mark node. Parameters μ and α are derived as follows. □

4.3.1 Correctness of algorithm

Angle μ is defined as constant and its value is derived by the help of criteria, given as follows.

1. When five independent nodes were selected then value of $\mu = 72°$ and the value of β is define as

$$\beta \leq 72° \tag{1}$$

2. When four independent nodes were selected then the value of $\mu = 90°$ and the value of β define as

$$72° <= \beta <= 90° \tag{2}$$

3. When three independent nodes were selected then the value of $\mu = 120°$ and the value of β define as

$$72° <= \beta <= 120° \tag{3}$$

4. When two independent nodes were selected then the value of $\mu = 180°$ and the value of β define as

$$72° <= \beta <= 180° \tag{4}$$

Since coordinate assignments initiate from center CN of the network and the number of independent nodes will be greater than equal to 2, so the independent node will be $IN = \tau \in Nb(CN)$, where $\tau \notin Nb(IN - \tau)$. Then β and α are calculated as

$$\beta = \frac{360}{IN} \tag{5}$$

and

$$\alpha = 0, \beta, 2\beta, 3\beta.... \tag{6}$$

Hence, the coordinate provided to independent nodes are.

$$\text{Coordinate} = (\cos(\alpha), \sin(\alpha)) \tag{7}$$

Once coordinates of the central node $(0, 0)$ and some fixed node (anchors) on boundary are known, our proposed algorithm 1 uses triangulation (average of neighbour's known angle α) to obtain the coordinate for 1-hop neighbor of $Nb(CN)$. Our simulation observation reports that each iteration coordinate becomes efficient. Each iteration coordinate other than IN adjusts time to time.

4.4 Algorithm for virtual coordinate assignment

Once coordinate of center node CN and its 1-hop neighbor node are assigned then other nodes compute their coordinate. 1-hop neighbor $Nb(CN)$ of central node passes its α value to a non-assigned nodes. Each non-assign neighbor takes the average of received neighbor's angle (α) and computes its own angle (α). The new computed coordinates of non-assigned nodes are:

$$(P\ \cos(\alpha),\ P\ \ \sin(\alpha))$$

where $P = \frac{level}{level-1}$ and level is hop distance from center to some fixed node (anchors) on boundary. Number of iterations will adjust the redundant coordinates of node (with the same coordinate) and angle α is calculated using neighbour coordinates. Node angle will get distributed at each iteration. Whenever node joins or moves from the network or group due to mobility, a new coordinate is assigned by averaging of αs of its neighboring nodes and checks if any neighbor nodes has the same angle (α), then iterates. Initially, all the nodes are assigned with $white$ color but as node assign their coordinate their color changes to $gray$. To differentiates the center nodes from others, our algorithm assigns a red color to the center node. Iteration keeps the coordinates of independent nodes fixed, changing coordinates of non-independent nodes. As coordinate is assigned to nodes, their color changes to $gray$.

Algorithm 2 Virtual coordinate assignment from 1-hop neighbor of center node to boundary nodes.

Require: Coordinates of 1-hop neighbors of center node
Ensure: Virtual coordinate assignment in the networks.

$j = 2$
while $(j++ \leq k)$ **do**
  for i : **do**
    while $(j++ \leq k)$ **do**
      Send $< x_{coor}, y_{coor}, node\ id > to\ UNb(i) - CNb(i)$
      On receiving message from marked nodes increment counter (C) for each node $S = S \cup UNb(i)$
      change color of node $i = red$
    end while
  end for
end while
for $i : S$ **do**
  $x_{coor} = (\frac{i}{i-1}) * \frac{(x_{received})}{C_i}$
  change the color of node i to $gray$
end for
$C = 0$
for all $gray\ nodes$ **do**
  for $i :< gray > nodes$ **do**
    Send $< x_{coor}, y_{coor}, node\ id >$ to $MNb(i)$
    On receiving message from marked nodes increment counter (C) for each node
    If one or more coordinate received from are equal to node's coordinates then assign $x_{coor} = (x_{coor} + \frac{\sum x_{received})}{(C+1(msg_{received}+1))}$, $y_{coor} = (y_{coor} + \frac{\sum y_{received})}{(C+1)})$
  end for
end for

4.4.1 Correctness of algorithm

Distributed algorithm correctly finds out central node (CN) and radius (ρ) and correctly assigns the virtual coordinate using local algorithm for static as well as dynamic topology.

Figure 1 illustrates the holder node HN detection when two 1-hop dominating sets are merged.

Lemma 1 *Combining two 1-hop nodes, we get at most 2-hop radius circumference with 2-hop k-DS at its center.*

Proof Fusion between two 1-hop dominating set DS_{N_1} and DS_{N_2} forms a 2-hop radius circumference of 2-hop

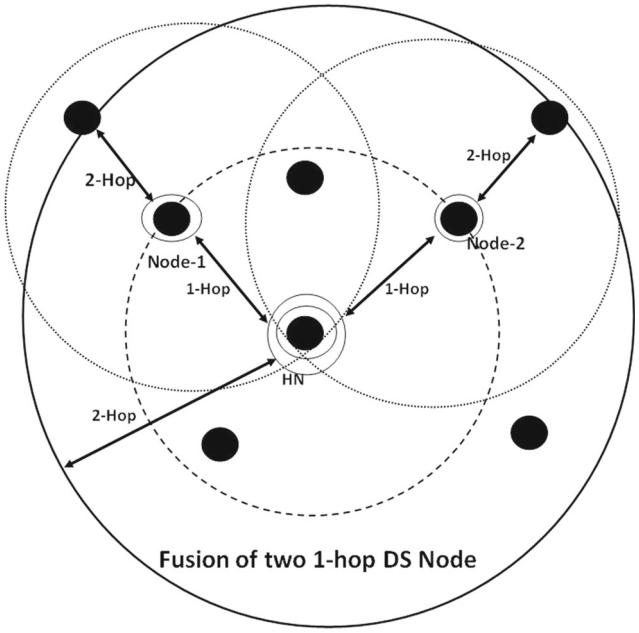

Fig. 1 Proof of lemma: 1

dominating set (DS), and holder node (HN) becomes the dominator node (by definition 3 and definition 4) as illustrated in Fig. 1. By mathematical induction law, dominating sets recursively keeps on concatenating to form a k-hop dominating set with single holder node at center and that will be the CN of the network (by Definition 1). □

Lemma 2 *Witness nodes always form the center (intersection) of the two nodes combined*

Proof Same as Lemma 1 □

5 Algorithm analysis

Consider a homogeneous WSN, modeled as UDG. It is NP-hard to recognize whether a given graph is UDG [14]. Consequently, it is also NP-hard to determine a set of virtual coordinates that satisfy all of the UDG constraints in the given unit disk graph.

5.1 Approximation analysis of virtual coordinate using dominating set algorithm

1. For the algorithm 1, the size of every maximal independent set (IS) in step 1 can be computed with approximation of $|IS|=(3.8opt + 1.2)$ where opt is the size of a minimum dominating set in the unit disk graph, from the reported result [15].
2. For the algorithm 2, the size of witness node in step 4 can be computed with approximation of $|WN|=((1 + \ln 5)opt)$ where opt is the size of a minimum dominating set, from the reported result [16].
3. Thus, the size of minimum dominating set available for virtual coordinate assignment algorithm is

   $|IS| + |WN| = 3.8opt + 1.2 + (1 + \ln 5)opt = (4.8 + \ln 5)opt + 1.2$.

   Thus, the virtual coordinate assignment using dominating set algorithm has the approximation ratio $((4.8 + \ln 5)opt + 1.2)$, where opt is the minimum size dominating set [17].

5.2 Complexity analysis

The algorithm for virtual coordinate assignment has time complexity $\mathcal{O}(n)$ times and $\mathcal{O}(D)$ rounds where D is radius and message complexity is $\mathcal{O}(n \log n)$.

6 Simulation results

In this section, we present simulation result of our proposed virtual coordinate assignment (DVC) algorithm. We evaluated the performance of our proposed DVC algorithm with geographic routing without location information (NoGeo) [4] and physical coordinate or true coordinate system. The simulation is performed using high-level event driven simulator Netsim [18]. All the nodes have a uniform radio range and two nodes can only communicate if the line of sight is not obstructed by obstacle.

Each node is independently and randomly placed on a two-dimensional simulation area and uniform random

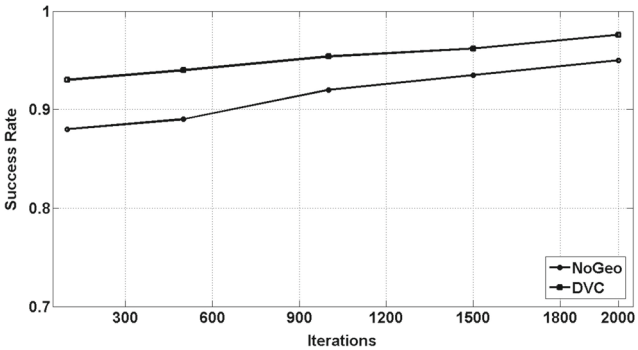

Fig. 2 Success rate comparison of greedy routing using DVC and NoGeo at different iterations

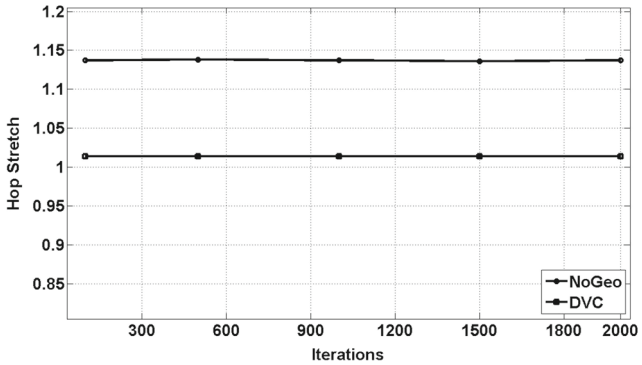

Fig. 3 Hop stretch comparison of greedy routing using DVC and NoGeo at different iterations

distribution [19] is used. Communication range is adjusted according to radio range equation [19], so that the connectivity of network is maintained even as low density.

Greedy forwarding mechanism is used as routing protocol over virtual coordinate systems. Evaluation of our algorithm is done in terms of number of iterations for successful convergence (hop stretch, success rate) and average number of geocast message generated (network overhead), end to end delay in greedy forwarding and node/link failure. The network parameter used in the simulation is stated below.

Area	L × L	200 × 200 grids
Range	R	10m
Number of nodes	N	300 to 1500

6.1 Convergence

Fast convergence rate is the requirement for any virtual coordinate-based algorithm. In this subsection, we evaluate the convergence rate of proposed DVC algorithm in terms of number of iteration required for convergence and its effect on success rate and hop stretch. In this scenario, a total of 1500 nodes were deployed randomly in an area of 200 × 200 $units^2$ and a number of iteration were increased from 100 to 2000 rounds. Since true coordinates get unaffected by iterations, hence, true coordinate comparison is ignored in this simulation. Result in Fig. 2 shows that after 100 iterations, the success rate of our proposed DVC approach reaches to .93 whereas NoGeo still shows greedy routing success rate below .90. As the iteration was increased, success rate reaches to almost .99. Result also shows that increase in iteration increases success rate up to .04 times for DVC whereas .07 increase in success rate for NoGeo is noticed, which clearly shows that DVC requires less number of iteration for convergence than NoGeo.

When hop stretch is measured in the same scenario, it is found that there is a drop of 12.5 % in hop stretch of DVC as compare to NoGeo. Figure 3 shows the result. Result obtained in Figs. 2 and 3, concludes that the number of iteration can impact VCS convergence which effect the success rate and hop count. It is also found that DVC takes less time in convergence as compare to the NoGeo, since our algorithm provides coordinates with circular overlay.

6.2 Network overhead

Network overhead plays an important role in resource constraint devices like sensor networks. In this section, we evaluate network overhead in terms of average number of message generation during greedy forwarding. Result obtained in this simulation is plotted in Fig. 4 where X axis represents the average number of packet generation and Y axis represents the number of iterations. Simulation scenario consists of 1000 randomly deployed nodes in an area of 200 × 200 $units^2$. The results obtained after simulation, in Fig. 4, show that DVC produces less number of packet as compare to NoGeo. It is also found that their is almost 2.5 % less network overload in case of DVC algorithm. Result in Fig. 5 shows that there is more than 5 % fall in average number of packet generated in case of our proposed DVC algorithm with respect to NoGeo . This

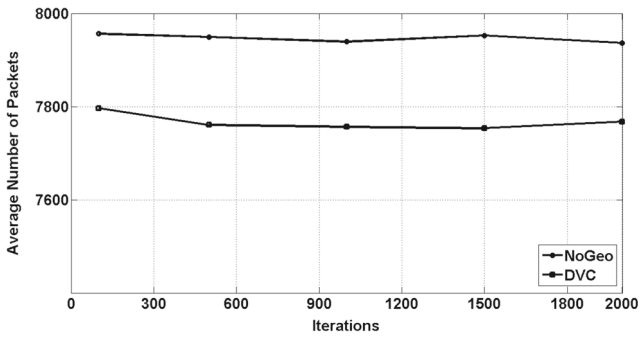

Fig. 4 Result shows the average number of packet generated in DVC and NoGeo, when number of iteration is increase

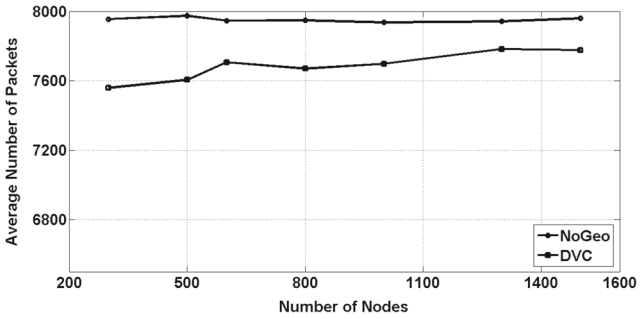

Fig. 5 Result shows the comparison of average number of packet generated in DVC and NoGeo, when number of nodes is increase

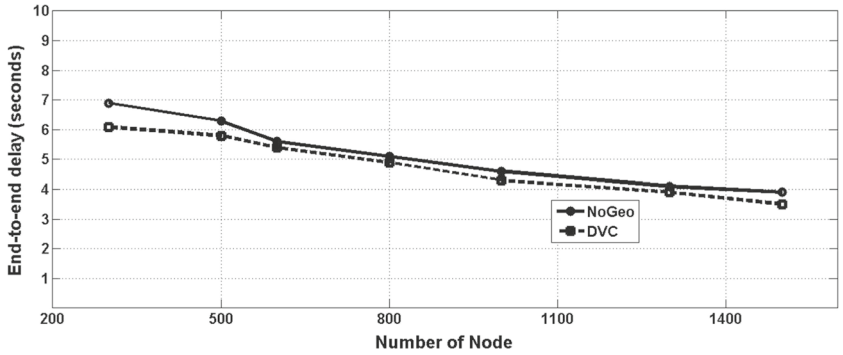

Fig. 6 Result shows the end to end delay between DVC and NoGeo (when number of nodes is increased in networks)

is reasonable because NoGeo algorithm at bootstrap time assigns two nodes as beacon nodes. Then, other nodes in the networks determine if they are perimeter nodes using heuristic approach on hop count from the beacons (this process requires to flood the network). Once the perimeter nodes are determined, then $\mathcal{O}(p^2)$ (where p is the number of perimeter nodes) messages are exchanged, so that each node uses these inter-perimeter distances to compute normalized coordinates. Finally, one beacon periodically floods the entire network, and each node periodically broadcasts within its radio range to share the neighborhood information. Whereas our proposed DVC algorithm requires only $\mathcal{O}(n \log n)$, number of messages for coordinate construction and each node periodically share their coordinate information with neighbors.

Result in Fig. 5 also shows that performance of the proposed DVC algorithm is better than NoGeo when network density (i.e., 300 nodes in $200 \times 200\ units^2$) is low.

6.3 End to end delay

Figure 6 illustrates the result concerning the end to end delay in the networks having number of nodes ranging from 200 to 1500. In this scenario, we assume that interference level along the route is negligible. In the simulation, we found that DVC and NoGeo algorithms are able to successfully set the path for every source to destination nodes pair. The result of Fig. 6 shows that DVC algorithm is better than the NoGeo in terms of end to end delay. Initially, when the number nodes are fewer in number then the end to end delay is higher for both the algorithms but as the number of nodes are increased, the end to end delay has fallen. This is reasonable because when the numbers of nodes are fewer (300) then there are chances of holes and dead ends in the networks (greedy routing deteriorates as nodes have no neighbor close to destination.

6.4 Node and link failure

Since WSN suffers from high energy drainage and frequently node and link failure, therefore, we performed an additional simulation to evaluate impact of link failure on coordinate assignment. Simulation results in Fig. 7 show the average number of extra hop stretch requires to route the packet in the network where frequent node failures may occur. We keep the scenario the same (1500 nodes in $[200 \times 200\ units^2]$ at $R = 8\ units$) but we randomly removed the deployed 25 % (i.e., around 375) of nodes. Then we randomly add the same number of new nodes in the network. The simulation is repeated for the 2000 iterations.

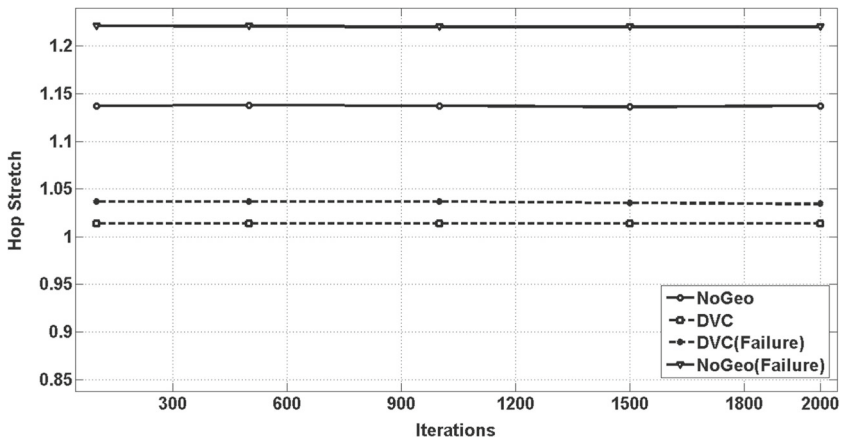

Fig. 7 Result shows the comparison of impact of link failure on coordinate assignment

Result shows that NoGeo algorithm has around 16–20 % increase in hop counts as compare to our proposed DVC algorithm. Result also shows that the proposed DVC algorithm is 0.41 % better than the without node failure scenario of DVC algorithm.

7 Conclusion and future work

In this paper, we propose a virtual coordinate assignment protocol, DVC, to assign virtual coordinates to nodes that have no geographic information. The proposed virtual coordinate system works in three phases: in the first phase, DVC algorithm identifies center node CN and radius (ρ) of networks; in the second phase, it assigns a virtual coordinates to all 1-hop neighbors of the central node and in the last phase, virtual coordinate assignment is done from 1-hop neighbors of CN to boundary nodes BN. Virtual coordinates could be made more robust by setting up coordinates of all the nodes in the network according to their angle α with center node CN as the origin. For future work, we are planning to extend our work to the three dimensional spaces with slight variations in some constraints and elimination of redundant and useless ID minimization. An important research goal for the future is to determine whether the virtual coordinate obtained through this algorithm can support various geographic and other routing algorithms. It is NP-hard to determine a set of virtual coordinates that satisfy all of the UDG constraints in the given unit disk graph (since, it is NP-hard to recognize whether a given graph is UDG). Our virtual coordinate assignment using dominating set algorithm has an approximation ratio $((4.8 + \ln 5)opt + 1.2)$, where opt is the minimum size dominating set. And our algorithm has time complexity $\mathcal{O}(n)$ times and $\mathcal{O}(D)$ rounds where D is the radius and message complexity is the $\mathcal{O}(n \log n)$.

References

1. Abreu DP et al (2016) A resilient Internet of Things architecture for smart cities. Annals of Telecommunications: 1–12
2. Mirsadeghi M, Mahani A (2015) Energy efficient fast predictor for WSN-based target tracking. Ann Telecommun 70(1-2):63–71
3. Bouet M, Pujolle G (2010) RFID in eHealth systems: applications, challenges, and perspectives. Ann Telecommun 65(9-10):497–503
4. Rao A et al (2003) Geographic routing without location information. Proceedings of the 9th annual international conference on Mobile computing and networking. ACM
5. Leong B, Liskov B, Morris R, Liskov B, Morris R (2007) Greedy virtual coordinates for geographic routing
6. Newsome J, Song D (2003) GEM: Graph EMbedding for routing and data-centric storage in sensor networks without geographic information. Proceedings of the 1st international conference on Embedded networked sensor systems. ACM
7. Huang P, Wang C, Li X (2012) Improving end-to-end routing performance of greedy forwarding in sensor networks. IEEE Trans Parallel Distrib Syst 23(3):556–563
8. Karp B, Kung H-T (2000) GPSR: Greedy perimeter stateless routing for wireless networks. Proceedings of the 6th annual international conference on Mobile computing and networking. ACM
9. Ke L, Abu-Ghazaleh N (2008) Aligned virtual coordinates for greedy geometric routing in WSNs. Int J Sensor Netw 3(4):252–265
10. Tan G, Bertier M, Kermarrec A-M (2009) Convex partition of sensor networks and its use in virtual coordinate geographic routing. INFOCOM 2009, IEEE. IEEE
11. Awad A, German R, Dressler F (2011) Exploiting virtual coordinates for improved routing performance in sensor networks. IEEE Trans Mob Comput 10(9):1214–1226
12. Lin C-H et al (2008) Virtual-coordinate-based delivery-guaranteed routing protocol in wireless sensor networks with unidirectional links. INFOCOM 2008. The 27th Conference on Computer Communications. IEEE. IEEE
13. Nieberg T (2006) Independent and dominating sets in wireless communication graphs
14. Breu H, Kirkpatrick DG (1998) Unit disk graph recognition is NP-hard. Comput Geom 9(1):3–24
15. Wu W, Du H, Jia X, Li Y, Huang SCH (2006) Minimum connected dominating sets and maximal independent sets in unit disk graphs. Theor Comput Sci 352(1):1–7
16. Li Y, Thai MT, Wang F, Yi CW, Wan PJ, Du DZ (2005) On greedy construction of connected dominating sets in wireless networks. Wirel Commun Mob Comput 5(8):927–932
17. Misra R., Mandal C (2010) Minimum connected dominating set using a collaborative cover heuristic for ad hoc sensor networks. IEEE Trans Parallel Distrib Syst 21(3):292–302
18. Leong B Geographic routing network simulator, 2004. http://web.mit.edu/benleong/www/netsim
19. Bettstetter C (2002) On the minimum node degree and connectivity of a wireless multihop network. In: Proceedings of the 3rd ACM International Symposium on Mobile ad Hoc Networking & Computing, pp 80–91. ACM