Intelligent Actor Mobility in Wireless Sensor and Actor Networks

Sita S. Krishnakumar and Randal T. Abler

Georgia Institute of Technology, Savannah GA 31407, USA
{sita,randal.abler}@gatech.edu

Abstract. In wireless sensor and actor network research, the commonly used mobility models for a mobile actor are random walk model, random waypoint mobility model, or variants thereof. For a fully connected network, the choice of mobility model for the actor is not critical because, there is at least one assured path from the sensor nodes to the actor node. But, for a sparsely connected network where information cannot propagate beyond a cluster, random movement of the actor may not be the best choice to maximize event detection and subsequent action. This paper presents encouraging preliminary results with static intelligent mobility models that are found using the inherent clusters' information of a sparsely connected network. Finally, a proposal to develop dynamic intelligent mobility models for the actor based on a mathematical model is presented.
Key words: mobile actor, mobility model, intelligent movement, clusters

1 Introduction

The concept of a mobile actor is relatively new in the area of wireless sensor and actor networks (WSAN). The main advantage of having a mobile actor is that more effective action can be taken, since the actor can get as close as possible to the event location. Additionally, in a densely connected network the presence of a mobile actor eases the problem of overburdened forwarding nodes by balancing the load among all the nodes in the network. In a sparsely connected network, a mobile actor can bridge the connectivity between groups of isolated nodes. But, there is no guarantee that the existing mobility models will work efficiently for a mobile actor in a sparsely connected network. A sparsely connected network can be used in applications where $< 100\%$ event detection is acceptable. An example application is coastal monitoring where sensor nodes may be deployed to monitor water temperature, wave characteristics, water level or meteorological conditions. When an event occurs in this environment, a possible action taken by the actor node could be of collecting a water sample for further analysis.

In a sparsely connected network, there are naturally occurring groups of nodes within which information can be sent. These node groupings are referred to as *clusters*. An event can be detected by one or more sensor nodes belonging to one or more clusters. The event detection information can then be propagated throughout the cluster which has detected the event, but cannot be sent beyond

that cluster. The information can reach the actor only if the actor visited the cluster which has detected the event. It would be valuable if the mobile actor followed a path such that the number of events detected in a timely manner can be maximized. This paper presents a novel way to generate static intelligent mobility models for the actor, using the inherent clusters' information of a sparsely connected network. The rest of the paper is organized as follows: Section 2 discusses related work with mobile actors and Section 3 presents the methodology used to develop the static mobility models. Section 4 presents the simulation results and Section 5 discusses the future work to develop dynamic intelligent mobility models. Section 6 summarizes and concludes the paper.

2   Related Work

The commonly used mobility models in WSAN research are the random walk mobility model, random waypoint mobility model, or a variation of the two. In the case of the random walk model, the actor moves towards a random destination at a certain speed. On reaching the destination, it chooses yet another random destination and continues towards that destination, and so on. In the case of the random waypoint mobility model, random destinations are picked uniformly in the region and the actor moves at a selected speed which is also chosen uniformly in an interval. Upon reaching the destination, the actor pauses for a time period, and the process repeats itself afterwards. Random mobility models are non-intelligent mobility models. When evaluating the performance of algorithms and protocols, they are suitable because the results obtained are then not tailored to specific mobility models. When using random mobility models, evaluating against a fully connected network assures that there is always a path available between the sensor nodes and the actor nodes for information exchange.

One of the prime requirements of a WSAN is that events have to be detected in a timely manner and communicated to the actor at the earliest possible time [1]. Since a centralized sink may not be present in a WSAN, coordination and communication between the actors and between the sensors and the actors have to be managed. The coordination and communication problem has been studied and a hybrid location management scheme has been proposed to handle the mobility of actors in [8]. This approach deals with a fully connected network and the actor moves following a variation of the random waypoint mobility model. Though the action and movement energy of the actor are assumed to be orders of magnitude higher than the energy required for communication purposes, this paper does not analyze the impact of random actor movement on the movement energy.

Another research paper that deals with mobile actors proposes a real-time coordination and routing framework for sensor-actor coordination to achieve energy-efficient and reliable communication [9]. The sensor nodes in the network form clusters based on application parameters and elect a cluster-head, which is responsible for communicating with the actor node. A backbone network of

forward-tracking and backtracking mechanisms is formed, which helps in routing the packets. The main goal of the framework is to deliver event detection information to the actor within a given delay bound. When the actor is mobile, the deadlines miss-ratio does not reach zero even with sufficient delay. This could either be due to congestion in the network or due to the random movement of the actor. These results highlight the fact that when routing is done hierarchically using only cluster-heads, random movement of the actor may not be the best choice for data collection.

Mobile entities have also been used in wireless sensor network (WSN) and mobile ad-hoc network (MANET) research. Data mules are mobile entities that collect data from sensors as they pass by [10]. They were introduced to help with non-real-time data collection in sparsely connected sensor networks. In simulations, the mules followed a random path or a deterministic path [4]. Mobile elements were also used for data collection in sparsely connected networks [3]. Nodes were partitioned into bins based on geographical locations and buffer overflow times. The schedule for each bin was created by solving the traveling salesman problem. These individual paths were concatenated to form the overall path of the mobile element. Message ferrying is a proactive routing scheme that was introduced in highly partitioned wireless ad-hoc networks where network connectivity can be maintained only when the nodes use their long range transmission radio [11]. The message ferry basically reduces all communication to a single hop at the cost of delay. The route taken by the ferry is found using a traveling salesman problem approximation followed by optimization of average delay.

The mobile entities deployed in WSNs and MANETs were meant to collect data from every single node in the network. In our research, the sparsely connected networks have naturally occurring node groupings within which the nodes can communicate with each other. When an event occurs in a cluster, all the nodes in the cluster will receive the event detection information using propagation techniques. This paper uses the clustering information of the network to help define a path for the mobile actor, with the goal of maximizing the number of events detected in a timely manner. To the best of our knowledge, there is no known research work in the WSAN area where the mobility model of the actor is reflective of the network in which the actor is deployed.

3   Methodology

To devise a mobility model for the actor in a sparsely connected network, the clusters of a given network has to be found. Pseudo code to generate the clusters' information from the sensor node locations and the sensor node transmission range is shown in Fig. 1. Sensor node locations can be obtained using localization techniques such as angle-of-arrival measurements, distance-related measurements, or received signal strength profiling techniques [6].

Based on the clusters' information, a decision has to be made as to which clusters the actor will visit and when. A large number of sensor nodes in a cluster

```
/* find the neighbors of every node in the network */
/* tx_range - transmission range (diameter) of a sensor node */
for(every node i in the network ) {
    for(every node j in the network ) {
        if( node i != node j ) {
            if( distance between node i and node j < tx_range ) {
                node i and node j are neighbors;
            }
        }
    }
}

/* find paths emanating from every node */
for(every node i in the network ) {
    find all possible paths in the network;
    cnt[i] = unique number of nodes reachable from node i;
}

/* cnt[i] denotes the size of the cluster to which node i belongs */
for(every node i that has not been assigned to a cluster ) {
    k = number of nodes that have the same cluster size as node i;
    if( k == cnt[i] ) {
        /*all k nodes are part of the same cluster*/
        assign a cluster_id and associate the nodes;
    }
    if( k > cnt[i] ) {
        go through the paths and find clusters of
                size cnt[i] from the k nodes;
        assign each cluster a cluster_id and associate
                the corresponding nodes with that cluster;
    }
    mark the k nodes as having been assigned to a cluster;
}
```

Fig. 1. Pseudo code to find the clusters in a network

does not automatically qualify the cluster for a visit. The parameter of interest is the area occupied by the cluster. This is because, the larger the area of a cluster, the greater the chance of an event occurring in that cluster. The clusters' area can be found using the pseudo code shown in Fig. 2.

```
for( every square meter of the field ) {
     for( every node n in every cluster m ) {
          if( distance between node n and center of this sq. m. < tx_range/2 ) {
               mark this sq. m. as covered by node n of cluster m;
               increment cluster m's area by 1 sq. m.;
          }
     }
}
```

Fig. 2. Pseudo code to find the area of clusters in a network

Given that the probability of an event having occurred is $P_E(t)$, the probability of an event having occurred in a cluster $P_{E_c}(t)$ is given by

$$P_{E_c}(t) = \frac{A_c}{A_t} * P_E(t) . \tag{1}$$

where A_c is the area of the cluster and A_t is the total area. The clusters with a higher probability of an event occurrence are those that occupy more area and when visited by the actor may lead to better event detection results.

A mobility model generates the path taken by a mobile actor and is defined by

1. A set of points S and
2. An order in which to visit the points in S to generate a path P.

When an event occurs in a cluster, the event detection information is propagated throughout the cluster. All the nodes in a cluster will have the same information over a period of time. The time required for this information to be propagated throughout the cluster will vary depending on the location of the event and the area of the cluster. In this preliminary research, the centers of the clusters are used as points to visit. The center of a cluster can be found by averaging the x and y coordinates, respectively, of all the sensor nodes belonging to the cluster. This ignores the possibility that the geographic center of the cluster may not be part of the cluster. For those clusters where the probability of an event occurrence is high, the centers of the clusters are found and added to S. Now, the points in S have to be ordered to find the path P. Any point in S can be chosen as a starting point. The following approaches are just a couple of ways to find P:

1. Using the *nearest-neighbor* approach: This approach follows a greedy algorithm similar to a solution for the traveling salesman problem. Fig. 3 shows the steps involved in generating a path using this approach. Depending on the speed of the actor, the duration of a simulation may be longer than the time it takes to visit the points in S once. If this is the case, then after all the points in S have been visited once, the last point becomes the new starting point and Steps 2-4 are repeated to generate the path P for the required duration.

2. Using the *nearest-x-coordinate* approach: In this approach, a fixed list denoted by L is generated, as shown in Fig. 4. For a starting point s, scan the list L to find s. Add the point to P. The list L is then browsed forward and points are added to P. When the last point in L is reached, L is browsed backwards and points are added to P. When the first point in L is reached, L is browsed forward again and so on for the duration of the simulation. With this approach, the field is traversed in a zigzag manner along the y-axis. Some points may be visited more than once before all the points in S have been visited once. Similarly the nearest-y-coordinate approach can also be used where the field is traversed in a zigzag manner along the x-axis. In this work, only the *nearest x-coordinate* approach is used in experiments.

---

1. Add the starting point to P and mark it as visited in S.
2. Select the closest unvisited point to visit next. Add that point to P and mark it as visited in S.
3. Repeat Step 2 until there are no more unvisited points in S.
4. Mark all points in S as unvisited.

Fig. 3. Steps to generate a path using the nearest-neighbor approach

---

1. Find the point in S with the smallest x-coordinate value. Add this point to L and mark it as visited in S.
2. Select an unvisited point in S with the closest x-coordinate to the current point to visit next. Add that point to L and mark it as visited in S.
3. Repeat Step 2 until there are no more unvisited points in S.
4. Mark all points in S as unvisited.

Fig. 4. Steps to generate an ordered list using the nearest-x-coordinate approach

4   Preliminary Results

4.1   Simulation Environment

The simulations were run against a network of 100 sensor nodes in a 200 m x 200 m field with one actor node. Sensor node locations were generated using a uniform random number generator. The routing protocol used was an optimized broadcast protocol known as BCAST [5]. This protocol keeps track of one-hop and two-hop neighbors and sends information only to them. Only packets that would reach additional neighbors are re-broadcast. Other simulation parameters are shown in Table 1. The stationary, fixed duration events are uniformly distributed in space and time. The terminating simulations were carried out using the network simulator ns-2 [7]. The events were represented in ns-2 using extensions from the Naval Research Laboratory [2].

Table 1. Simulation parameters

Number of sensor nodes	100
Number of actor nodes	1
Transmission range: Sensor node	20 m
Transmission range: Actor node	30 m
Actor speed	2 m/s
Field dimension	200 m x 200 m
Event diameter	30 m
Event duration	100 seconds
Number of events	10

For the example network, the clusters' information and their area were generated using the steps shown in Figs. 1 and 2 respectively. There were 23 clusters ranging in size from 1 to 24. The largest cluster occupied 11.86% of the total area, while the smallest cluster occupied 0.39% of the total area. The probability of an event having occurred in a cluster was found for all the clusters. For the example network, the clusters with a higher number of nodes corresponded to a larger area and hence a higher probability of an event occurrence. For the simulations, 14 clusters where the cluster size was greater than 1 were selected. The chosen clusters' centers were then calculated to be used as points to visit.

The simulation duration was 400 seconds. Actor paths were generated for the simulation duration using the two approaches defined in Section 3. An actor path following the random walk mobility model was generated using tools available in ns-2 for 400 seconds. The actor speed for random movement was also set at 2 m/s.

Table 2. Simulation results (***italic*** entries highlight values obtained using intelligent mobility models - NN: nearest-neighbor approach, NX: nearest-x-coordinate approach - that are better than values obtained using RW, the random walk model)

	Average time (secs)			Worse time (secs)			Events detected		
	NN	NX	RW	NN	NX	RW	NN	NX	RW
Run 1	*23.61*	*13.09*	42.64	86.00	*40.98*	79.95	*7*	*5*	3
Run 2	*0.04*	*18.32*	28.08	*0.11*	*50.14*	56.15	*4*	*4*	2
Run 3	59.74	*4.09*	18.35	100.11	*20.12*	35.89	*5*	*5*	3
Run 4	*22.76*	*14.69*	25.30	68.01	*47.95*	56.25	3	*5*	4

4.2   Simulation Results

For each of the three mobility models, four simulations were run. The starting point of the actor was varied in each run. For each run, the time to detect every event was noted. From this, the average time to detect an event was found. This, along with the worse time to detect an event and number of events detected are tabulated, as shown in Table 2.

The average time to detect an event with the proposed intelligent mobility models fared better than the random walk mobility model in most of the individual runs. The worse time to detect an event was not as conclusive. This is because the mobility models used in the simulations were preliminary in design with no optimization performed. The proposed mobility models also detected more events in most runs than the random walk model, as shown in Table 2. The clear winner is mobility model *NX* - the mobility model going zigzag along the y-axis using the nearest-x-coordinate approach with center point of clusters - and it outperformed the random walk model in every run.

5   Future Work

The preliminary results validate the assumption that a random walk model is not the best mobility model for the actor in a sparsely connected network. The intelligent mobility models that have been defined in this paper are static in nature and do not take into account the time an actor may spend acting. The paths generated by these static mobility models can be calculated in advance and can be used in situations where the actor's movement has to be pre-loaded. The future goal is to optimize the actor's path in a dynamic environment where, when an event occurs, the actor may be involved in acting for an unspecified duration of time. In order to be able to incorporate the dynamic acting pattern of the actor, we plan to define a mathematical model for fixed duration events. This mathematical model would be implemented at the actor node and would help in defining the expected number of detectable events in a cluster and the expected number of lost events (events that occurred but went undetected) in a cluster. Depending upon an application's requirements, different dynamic mobility models will be generated for the actor node using the mathematical model.

The results of the dynamic mobility models will then be compared against the static mobility models that have been presented in this paper.

With a sparsely connected network and sparse number of actors, approaching 100% detection may not be possible. We plan to conduct extensive simulations to study how close to 100% detection we can reach with optimization.

6   Summary

Static intelligent mobility models for the actor were generated using the clusters' information of a sparsely connected network. Even though random walk is not an intelligent mobility model, we used it for a baseline comparison because of it's prevalence in literature for use in both fully connected and sparsely connected networks. With one of the proposed intelligent mobility models, on an average 60% more events were detected than with a random walk model. Also, the average time to detect an event was on an average 60% lesser than the results with a random walk model. These results have validated the idea of using the inherent characteristics of a sparsely connected network to generate intelligent mobility models for the actor, which helps in detecting more events in a timely manner. A simple yet effective mathematical model will be developed, that would aid in generating dynamic intelligent mobility models for the actor based on application requirements. These dynamic mobility models will be evaluated against different application parameters and the results will be reported in a future paper.

References

1. Akyildiz, I., Kasimoglu, I.: Wireless sensor and actor networks: Research challenges. Ad Hoc Networks Journal (Elsevier) (2004) 351–367
2. Downard, I.: Simulating Sensor Networks in NS-2, NRL/FR/5522–04-10073, Naval Research Laboratory, Washington, D.C., May 2004
3. Gu, Y., Bozdag, D., Ekici, E., Ozguner, F., Lee, C-G.: Partitioning based mobile element scheduling in wireless sensor networks. Proceedings of IEEE Secon'05 (2005) 386–395
4. Jain, S., Shah, R., Brunette, W., Borriello, G., Roy. S.: Exploiting mobility for energy efficient data collection in wireless sensor networks. Mobile Networking Applications (2006) 327–339
5. Kunz, T.: Multicasting in mobile ad-hoc networks: achieving high packet delivery ratios. CASCON '03: Proceedings of the 2003 conference of the Centre for Advanced Studies on Collaborative research 156–170
6. Mao, G., Fidan, B., Anderson, B.: Wireless sensor network localization techniques. Computer Networks 2007
7. McCanne, S., Floyd, S.: ns–Network Simulator, Available from http://www-mash.cs.berkeley.edu/ns/
8. Melodia, T., Pompili, D., Akyildiz, I.: A Communication Architecture for Mobile Wireless Sensor and Actor Networks. Proceedings of IEEE Secon'06 (2006)

9. Shah, G. A., Bozyigit, M., Akan, O. B., Baykal, B.: Real-Time Coordination and Routing in Wireless Sensor and Actor Networks. 6th International Conference, NEW2AN 2006
10. Shah, R., Roy, S., Jain, S., Brunette, W.: Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks. IEEE SNPA Workshop, 2003
11. Zhao, W., Ammar, M.: Message Ferrying: Proactive Routing in Highly-Partitioned Wireless Ad Hoc Networks. FTDCS '03: Proceedings of the The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems