Locating Moving Objects over Mobile Sensor Network

Arvind Nath Rapaka[1], Sandeep Bogollu[2], Donald Wunsch[3], Ravindra Nath[4]

[1,2,3] University of Missouri Rolla
Rolla, MO, USA
{anrb6b, sbd26, dwunsch}@umr.edu

[4]JNV Narla
Kalahandi, Orissa, India
{ravi_bhagya}@rediffmail.com

Abstract-*The purpose of our on going research would be to track entities, which enter their field of vision over the sensor network. Based on their sightings, they maintain a dynamic cache that can be queried by a base station- a stationary or fixed node. Since transmitting messages consumes a lot of energy, as against local processing and moreover the limited energy in sensor node does not allow heavy duty computing, we need to develop algorithms that will ensure efficient computing and minimal transmission without degradation of performance. And also data is stored locally, and memory is finite, the nodes must choose their cache lines carefully and need to dump the redundant data so that it could avoid cache overflow.*

Here we propose a class of algorithms for locating target object moving over a sensor network. The algorithms work with energy and memory limitations of the nodes. Algorithms developed conserves energy by efficiently identifying sensor nodes and use local messages between neighboring nodes to follow the trail of the object, assuming local transmission cost less compared to the long range transmission. Since, we avoid the long-range transmission and maximizing the local granularity security increases. The entire dynamic nature of the algorithm exploits the knowledge management for effective identification of objects through predefined parameters such as the object, object velocity. Hence it effectively exploits the network management for locating the objects over the network with minimal energy conservation and short range transmissions.

1 Introduction

A wireless network is comprised of compact, autonomous nodes equipped with data-gathering, computation, and wireless communication capabilities. Ad hoc and sensor networks are considered to have the potential to be extremely useful in emergency situation where little or no infrastructure is available, which act as an independent-

GPS enabling to locate the objects [1]. For a GPS, line of sight is required, due to unavailability of GPS signals in building and indoor ambient, tracking objects is very difficult, however, it is very important and has potential commercial applications.
The usage of such sensor network systems, however, is now moving towards the consumer market and tremendous potential for applications. For example, the ability to track children is extremely valuable. Think your toddler wanders off without your knowledge. With a location system, we can easily track them among a large crowd. Through the use of a concealed wireless tracking tag, this type of system can aid tremendously if someone takes a child without permission.

Also, the use of location-support applications in retail environments is pretty exciting. You can deploy this type of system in a shopping mall or large retail store and implement electronic flyers and advertisements. The system takes into consideration the physical location of shoppers within the facility and customizes content appropriately.

For example, users can receive an electronic directory and advertisement flyer on their GPS/PDA after entering the mall. The directory would include a map of the facility that identifies the person's exact position on the map. As the shopper clicks on a store, restroom, or ATM machine in the directory, the map indicates directions that take them to the desired selection. If your friend is carrying a GPS/PDA device, then you can also keep track of each other's whereabouts.

2 Design Constrains

2.1 Algorithms

In a highly dynamic and unpredictable network, the network topological state may change more quickly than it can be tracked. Hence, the control decisions and predictable algorithms based on this domain knowledge will always lag behind the actual state of any mobile device. However, that even in cases where real-time tracking of current network state is feasible given the quantity of resources necessary for nodes to keep state information up-to-date should be large enough to preclude the network from performing other functions. Mobile devices may elect not to distribute all state information so that they are able to perform their other necessary functions, and hence mobile devices may have incomplete information about current network state.

Conventional methods used in wired networks cannot be employed in ad hoc network due to the following reasons:

- Its ad hoc and mobile nature.
- Difficult to address node over sensor network.
- Unstructured and undefined network topology.
- Security concerns as information is available in air and can be freely intercepted
- And importantly, limited resources such as computing power, memory and energy resources.
- Data shipping architecture can fail as large number of messages needed to ship all the data causes the sensors to use up their limited power supply, and die very quickly

2.2 Security

Before implementing such a system that tracks object/people, be sure to consider privacy issues. Of course these systems can help people, but unethical operators may take advantage. For example, a system tracking people belonging to a particular group could provide information on when individuals are far enough away from the group to be victimized. Think about implementing security mechanisms.

In Ad hoc networks with resources scarcity, protocols must be designed to consume the minimum amount of computing resources necessary for correct and acceptable operation without performance degradation. Unfortunately, many protocols in networks today are relatively easy to force them into states in which they consume large amount of resources.

And more unhealthy aspect in existing protocols is a malicious node could issue frequent link state changes or could even frequently change the state of one of its links, forcing the routing protocol to generate and distribute many unnecessary updates to the routing information that consumes much of the limited resources.

Hence designing such protocols that effectively manages the resources is a challenging task. Even in the presence of aberrant behavior of external sources, one needs techniques or tools that help identify the weaknesses of a protocol which allow it to enter a state that causes the protocol to deny service to the network [2,3]. Incorporating such techniques for detecting such intrusions in a protocol and in building attack tools for exercising protocols to discover such malicious attacks requires robust protocols, which the existing infrastructure cannot sustain.

Hence, our algorithms avoid long range messages and tracking objects are entirely a local computation with in the Sensor network, which is inherently a decentralized system and enhances the security aspect without development of robust protocols.

2.3 Bandwidth and Management and scalability

One of the most important parameter about network performance is the bandwidth. The objective of this project is to evaluate existing algorithms and to develop a valid measurement technique to estimate the *available bandwidth*. There are many issues that can degrade the performance of the entire network and hence our goal of this project is develop algorithms that effectively use the bandwidth with minimum level of interference.

As a wide spread deployment, it is not possible to administer a system in a scalable when all control functions are centralized. So designing a decentralized system not only provides scalability but also easy to install the system on owners interest. There is no need for a central entity to keep track of object. Also a decentralized system provides better bandwidth management. Hence, dealing with such issues need lot of debate and research for next generation applications. Past few years there has been lot of research to develop next generation technology, light weight computing stack that consumes less energy, security and many more. Here, we are proposing algorithms that are effective in ad hoc and unstructured network.

3 System Parameters

In this paper we struck with same naming convention as [1], and also we introduced new parameters at our requirements.

- Set of sensor nodes which have been distributed over a geographic region, *R: V = {Vl, V2, V3...}*
- *Nid:* Sensor node has a unique identifier. *(Capable of transmitting both long and short range messages; we will discuss the hardware model in later section.*
- *F'(v):* Node's field of vision
- Nbrs(v):Sensor Neighbors
- B: Base station *(which is aware of the topology of the sensor network)*.
- *Lid:* Logical identifier of the object
- C: Cache Memory.
- Cptr: Cache Pointer
- TS: Time Stamp
- LS: Last Access Time Stamp
- Lid: Logical Identifier
- Q: *(Lid, X, T(X, Lid)),* where *Lid* is given, while *X, T* are free.
- *T(X, Lid: The time stamp* of the target by *X*
- HN(Lid): home node of the object
- SR: Short Range distance

Here, any node in the sensor network can communicate to base station and vice versa. But, our algorithms were designed on limited resources existent in a sensor node. Since we consider long range messages consumes more power than short range messages, we avoid long range messages for better resource utilization without performance degradation. Moreover, local computation within the sensor network with less long range message will not only increase security but also better bandwidth management.

Each node in sensor monitors certain area F(v), field of vision. The senor input captured is converted into a signature Lid, logical identifier. The signatures of objects observed by a node are recorded in its cache, along with the time of its observance.

Base station transmits Queries to the sensor network. Unlike [1], the base station queries the sensor node which had reported about the object location on pervious query. And from then the queried sensor node will query the neighbor nodes about the current location of the object. The queries will be of the form Q: *(Lid, X, T(X, Lid)),* where *Lid* is given, while *X, T* are free. In words, *Q* asks for the location of a target with signature *Lid.*

The node with latest time stamp report the base station i.e T(X, Lid) > T(Y, Lid) for all Y in the sensor network. The corresponding node will be queried later whenever the base station wants to track the object again some time later. It can probe the network with queries asking for the location of a specified target. The sensors coordinate with each other to find a suitable answer for the query. Since it is desirable that the sensor network have as long a life as possible, the sensors should transmit as few mes-

sages as possible. In addition, each sensor should have similar duration of life to prevent "holes" in coverage.

4 Working Model

One such solution is uses a *query shipping* architecture [1]. Unlike data shipping, query shipping involves tracking the location of an object through short queries. Queries will be transmitted from an external base station to the node of the cut of sensor network nodes. The nodes coordinate among themselves with short query messages and after detecting the node with latest time stamp it will inform about the latest location to the base station. However, with the data being stored locally, there needs an effective memory management for each node.

Query shipping architecture provides better service than any conventional message shipping solutions for following reasons

- Queries are small in size compared to data passing; and are effective when tracking a location of an object compared to passing bulk data.
- The system will be more fault-tolerant compared to data shipping. The sensor node can retrieve information on the loss of message during transmission.
- Since structured and small messages are transmitted, we can model an effective memory management (discussed in later section). As such, the volume of data transmitted is minimized, and life of the network is increased.

Also, we discuss a resource limitations aware solution to the problem such as memory management, super-light algorithms of detecting the location of a physical target object. As evident our solution would be to ship the query to a node, which later queries the neighbor across the network using hop-by-hop routing. Any sensor node which has seen the object responds back to the base station, with its time of sighting.

We introduced a concept called “Home Node”. Basically, a home node is node like any other nodes in the sensor network expect acts like a home for the object that responds with the latest time stamp to the query from the base station.

We proposed a solution that optimizes the computing resources for a given node without performance degradation. Since physical objects are continuous in space and time, if the target object moved from one portion of the area to another, some node on the boundary separating the two portions must have seen the object. Moreover, objects leave a trail of entries in the caches of sensor nodes along their path. Thus, when ever the base station queries the “Home Node” it takes a follow-through algorithm over the entire network.

Once such a node is found, the trail left by the object is followed to its current position.

We propose a class of algorithms that build on the basic ideas above to minimize the number of long as well as short term message transmissions. Here we adhere to the naming system of [1]. But our algorithms and the data query set are different from the one they have used. And also we proposed a new set of memory management algorithm for effective memory utilization.

Moreover, we proposed a new novel idea of multicasting the short range messages depend on the velocity of the object. We define a cut based on velocity and then multicast the query across the cut, which will be discussed in detail in later section.

5 Related Work

Each sensor node in the Ad hoc network does have a field of vision, F (v), which indicates the area on which the node detects the object. Once detected, the node will retain the signature of the object and updates the cache based on the cache management protocol.

Initially, we discuss the algorithm discussed in [1] and then we propose our algorithm and in later sections we will do performance analysis. Each sensor node is able to retain signatures of all objects detected in its cache, and the cache is effectively unbounded. An object traces a continuous curve as it moves in the area covered by the sensors. The sensors register the object traces from the instant the object entered the monitored area until it left the area. If the sensors cover an area completely, then, at any instant, the object was in the field of vision of at least one sensor node. Correspondingly, as the object moved out of the field of vision of a sensor, it moved into the field of vision of a neighboring sensor node [1].

Sniffer Algorithm Outline: Locate a target object Input: *.Lid:* Logical identifier of the target object over the sensor network [1]:
Input: Lid: Logical identifier of the target object
Output: .x: Identifier for sensor which saw the target object last
T: Time of sighting of target object by X
Method:

- Find a sensor node, O, which has seen the target object last.
- Follow the target object trail to its latest position.
- Return the *Nid* and *T (Nid, Lid)* of the last node in the trail to the base station.

The above algorithm outlines suffices to find the most recent location of a given target objects but the reduction in the number of messages sent when their algorithm is used depends on the movement patterns of the objects. Their algorithm will end up in sending more messages when the object velocity is high i.e. the network will be flooded with short-range messages across the network and the performance will gradually decrease as the high velocity objects were tracked.

Moreover, the time to track objects across the network will increase since the query will processed over the entire network. The static nature of detection also makes the system less scalable and adaptable to the object defined parameters. Given a cut, if the target object *ever* passed from one part of the area to the other, at least one node on the cut must have seen the target object.

The process of picking up the trail is reduced to picking an appropriate set of cuts, and querying along the cuts for a sighting of the target object [1]. Hence, defining a proper cut is also essential and integral part of the design. Assuming a static cut will degrade the performance since the base station will query each leader on misidentification or detecting a static object.

6 Algorithms

6.1 Memory Management

Here we are proposing a set of algorithm, “care-less”, which effectively manages memory with high hit rate when the base station or any neighbor node sends a query for tracking objects.

For a care-less associative cache management, the replacement algorithm is a bit more complex. Whenever the sensor node references a memory location, the cache uses the unique object ID to select the set that should contain the cache line. Using our algorithm, the caching determines if the data is already present in one of the cache lines in the set. If not, then the sensor node lets the neighbor by sending NULL. If the sensor detects the new object, if cache lines are currently unused, the selection is trivial: pick an unused cache line and enter new data. If all cache lines are currently in use, then the cache must pick one of the cache lines and replace its data with the new data. Ideally, we'd like to keep the cache line that will be referenced first (that is, we want to replace the one whose next reference is later in time). Unfortunately, neither the cache is omniscient; they cannot predict which the best one to replace is. However, remember the principle of temporal locality has been referenced recently; it is likely to be referenced again in the very near future.

A corollary to this is "if a memory location has not been accessed in a while, it is likely to be a long time before the sensor node accesses it again." Therefore, a good replacement policy that many caching controllers use is the "least recently used" or LRU algorithm. The idea is to pick the cache line that was not most frequently accessed and replace that cache line with the new data. An LRU policy is fairly easy to implement in our cache system. All you need is a bit that is set to zero whenever the sensor node accessing one cache line and set it to one when you access the other cache line. This bit will indicate which cache line to replace when a replacement is necessary. Other possible replacement policies include First-in, First-out (FIFO) and random. These are easier to implement than LRU, but they have their own problems.

Algorithm Care-Less: Update the tracked Object by a sensor node.
Input: *Lid:* Logical identifier of the object
C: Cache Memory.
Cptr: Cache Pointer
TS: Time Stamp
LS: Last Access
Output: C Update the cache line

Method:
1) While (C is *Null)*
- Sensor node searches the location of the Search(Lid)

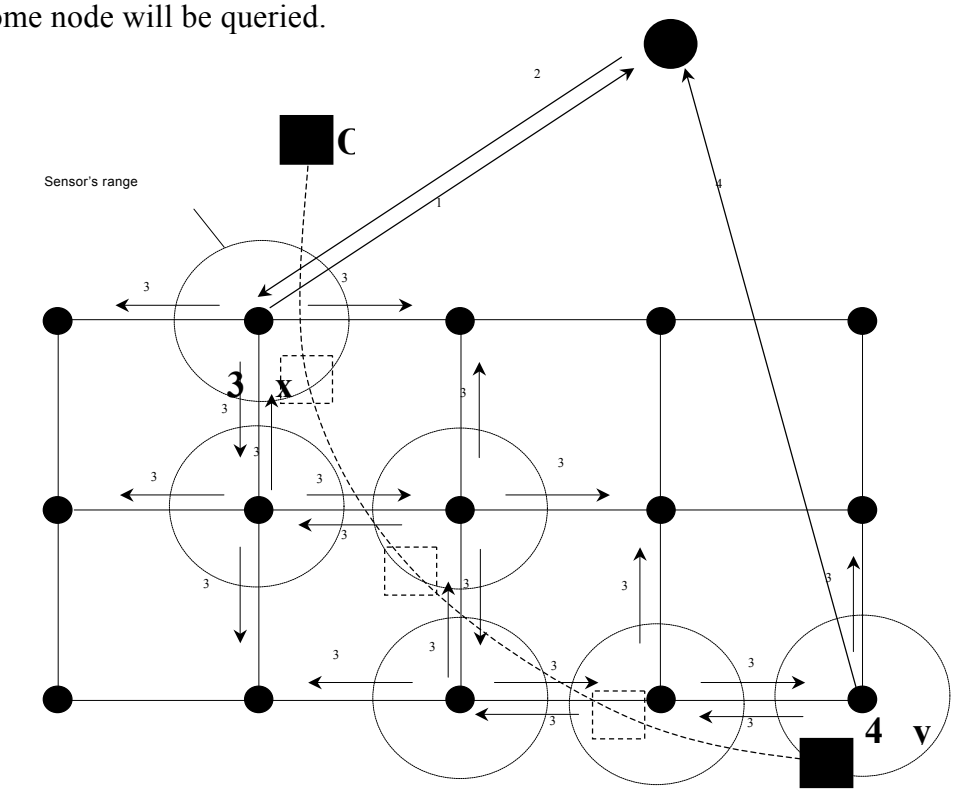
2) if (search(Lid)==Null)
- *(Cptr)= Id;
- Enter the TS and LS= TS against Cptr;
- Cptr++;

else
- Calculate the least used cache line on LS; LRU ()
- Enter the Id against the cache line;
- Enter the TS and LS= TS;

6.2 Tracking Objects

Here we describe algorithm that will track the objects. The base station receives a query *Q* that looks for an entity with signature *Lid.* The base station chooses the node

as the home of object, and later whenever the base station needs to locate the object the home node will be queried.



B. Base station for the remote sensors
O. Object passing through the deployed sensor network
1. Node x sends a message about locating the object O
2. Server responds by requesting the current location
3. Messages to the neighbors requesting latest time stamp of spotting O
4. Node y sends the message about the final location of the object O

Figure1: Tracking objects

The figure above describes the basic function of the location tracking system. The Base station queries the HN, home node, for tracking the latest location of the object. Later the HN, queries its Nbrs for its latest position. The above process ends when the below network finds the latest location of the object. The node will report the base station and in turn the base station updates the home node against the Lid.

Algorithm TrackObject: Find a home node of the object.
Input: *Lid:* Logical identifier of the target object
HN (Lid): Home Node of the Lid.
Output: X: Identifier of a sensor which saw the target object
Method:
(1) Ask all nodes in Nbrs of HN (Lid)
(2) if (no node in Nbrs of HN(Lid) has seen Lid later) send(Nid, T(Nid, Lid)) to base station.
Else ask the node with the latest T(Nid,Lid) to do TrackObject.
(3) Return(X);

The phase of the algorithm returns the identifier of the node with the most recent sighting time and is assigned as the home node of the object. The Base station will

contact the HN to continue to track objects, and sends it an *elect* packet. The elected node, HN, then starts the process of following the trail of the target object. If the target object moved out of range of the sensor, it must have gone into the range of one of its neighbors. By finding the particular neighbor, who saw the target object the last the elected node can determine the next step in the trail of the target object. Such a neighbor is then elected as the home node to follow the target object, and so on.

The trail stops when a node on the trail had seen the target object after *its entire* neighbors. The trail stop implies that either the target object is still in the range of this existing node, or the elected node is an edge node and the target object has left the grid of sensors. In either case, we have found the most recent sighting of the target object. Such a node then responds back to the base station giving its identifier and sighting time as the answer to the query. Once returned the base station updates the HN with the node that responded with the latest time stamp against the object.

6.3 MR Algorithm

Here we want to propose a different method of tracking objects over a sensor network that will track on object parameters such as velocity. Since the velocity is very important parameter to define the object location in time frame, we calculate the radius, a threshold, which later defines the most probable area of object location.
The corresponding node wills multicast the query to the entire sensor nodes that fall in the area.

Procedure MinRadius Track target object over the sensor network
Input:

- Lid: Logical identifier of the target object.
- Nid: Identifier of this sensor node.
- V1: Velocity of object at time T1.
- V2: Velocity of object at time T2.
- A: Multicast Area.
- SR: Minimum distance between the two sensors

Output:

- T: Time of sighting of target object by X.

Method:
(1) A= 3.14*([(V1-V2)/(TI-T2)]*[([T2-T3]) POWER2] +SR) POWER2;
(2) Ask all nodes in A;
(3) If (no node in A has seen Lid later) send (Nid, T (Nid, Lid)) to base station.
Else ask the node with the latest T(Nid,Lid) to do TrackObject

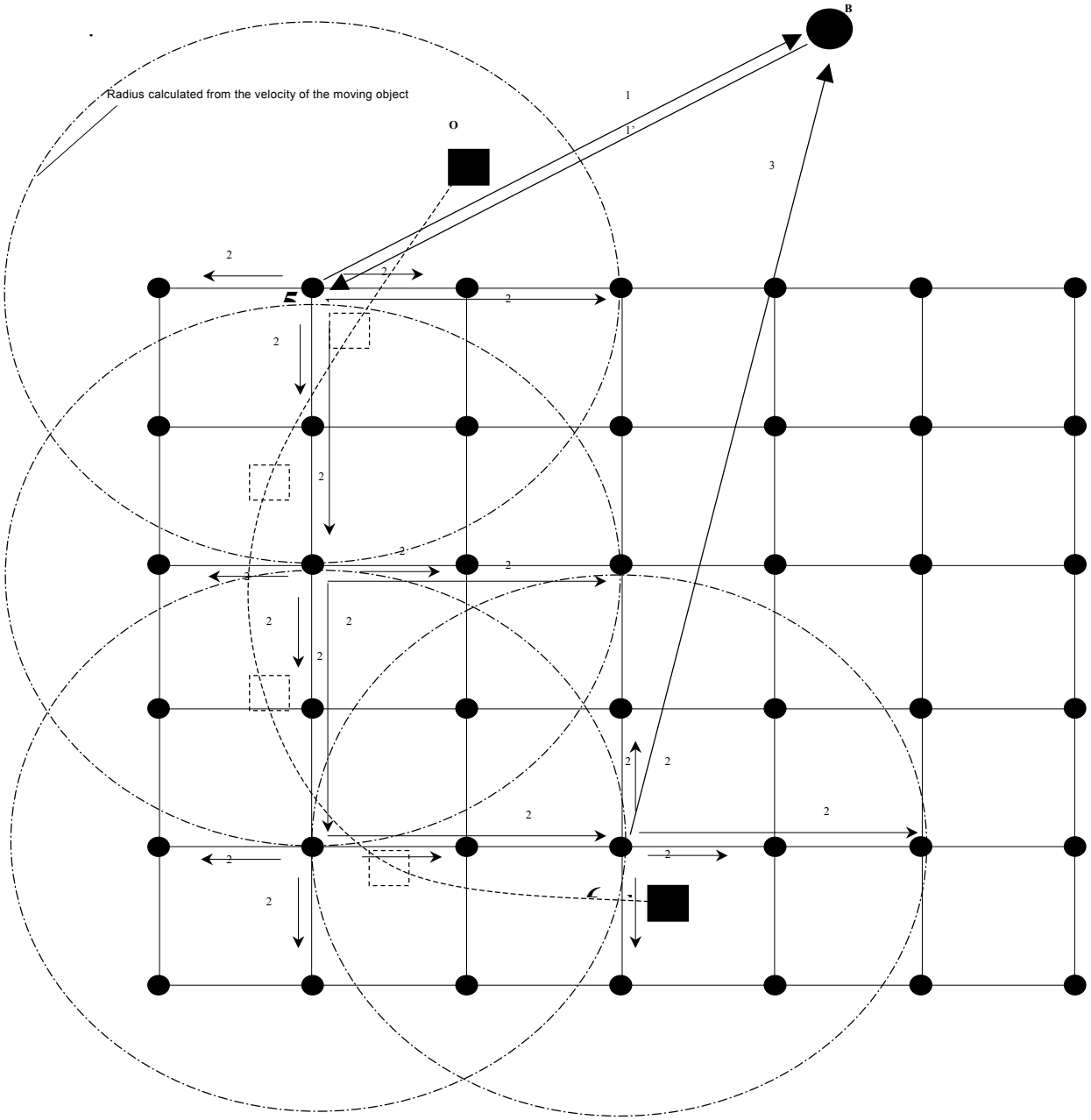
6.4 Randomized Algorithm for short range communication

Given the highly probable interference, sampling over a time window per transmitter, the sensor node, would be a good idea to estimate the correct location and also resolves the ambiguity in time stamp.

For example, when the object is equidistance from the neighboring sensor nodes, there is always a high probability that neighboring nodes duplicate the same time-stamp and thus leads to bad memory management. And also issues like carrier-sense-style channel to avoid collisions need intensive recourses and heavy computing, which the sensor node cannot sustain.

Instead, we handled these problems using a randomized algorithm. Rather than transmitting the signals continuously over time, each node will transmit the signals over an interval [R1, R2]. Thus the broadcast of different sensor node are independent, which avoids the issues like time stamp ambiguity, interference.

In this system, the allocation of the timeslots is on a synchronous basis. The timeslots are shared on a equal basis in strict rotation (below figure).



B. Base station for the remote sensors
O. Object passing through the deployed sensor network
1. Node sends a message about locating the object O
1'. Server requests for the latest location
2. Messages sent to all the nodes within the radius value calculated from the velocity of the object
3. Node sends the message about the final location of the object O

Figure 2: MR Tracking

Algorithm:

1. For each sensor nodes;
   Detect the timestamp of the object over the random time period [R1, R2] ms
2. The unique time stamp will be registered against the object in the cache.

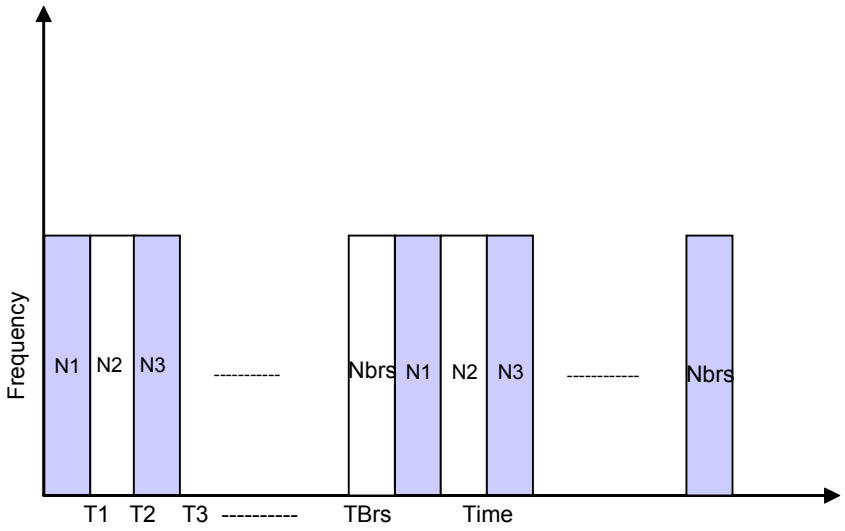


Figure 3: Randomized TDMA

7 Performance Analysis

In this section, we will describe the performance and compare against competing algorithm. Initially, we will analyze our set of algorithms for the number of elementary target movements. Since any node in the sensor network can communicate to base station and vice versa, and the node that reported the latest timestamp for the query will be assigned as home node of the object and further query for the object will be directed to home-node of the object.

The search algorithm follows an n-array tree. The order of such a tree is defined by the neighbors of the sensor network. Such a communication, between two neighbors, is a short range message. Moreover, local computation within the sensor network with short messages will not only increase security but also better bandwidth management against communication on common channel, in our case, the channel to base station.

Case	Tracking Objects	Algorithm[1]	MR algorithm
Best	Nbrs	N+Nbrs	4
Worst	$\lceil Log_{Nbrs}(NxN) \times Nbrs \rceil$	$\lceil Log_{Nbrs}(NxN) \times Nbrs \rceil$	N^2

Table1: Algorithm Complexity

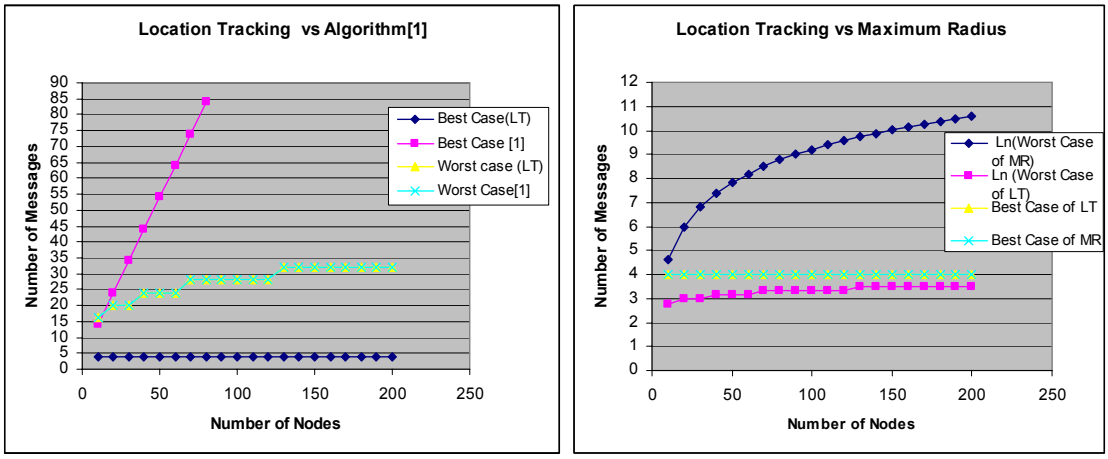


Figure 4: Best Case Analysis for LT vs. Algorithm [1] (left graph),
Worst Case Analysis for LT vs MR (right graph)

8 Conclusions and Future Work

The model defined in this paper aims at locating an object in a sensor network and the sensor that is currently holds latest time stamp of the object acts as the home node and communicates with the base station, the details related to the object. The query shipping approach was followed which allows for smaller messages to be transmitted, thus saving on energy and channel usage against the conventional data shipping. Another major advantage of our model is that, long-range messages, between base station and the nodes, are minimized by the local computation within the senor nodes of the network; this further reduces the power usage in the nodes. The location of the object can be retrieved by only two long-range messages, base station query and corresponding time stamp message. The short-range messages to all the neighbours querying for the latest time stamp and then finally, the node with the latest time stamp of the object reports to the base station will be the home node for the object. The short-range messages will ensure greater security.

9 References

[1] M. Anant, M. Bawa, J. Krishnan, and S. Merchant. Locating an object over a wireless sensor network. Project proposal, MIT, Boston, MA, USA, 2001.

[2] J. Kahn, R. H. Katz, and K. Pister. Next century challenges: mobile networking for 'Smart Dust'. In ACM MOBICOM Conference, pages 188-196, Seattle, WA, August 1999.

[3] S. Marti, T. J. Guili, K. Lai, and M. Baker. Mitigating routing misbehavior in MANETs. In ACM MOBICOM Conference, pages 122-132, Oakland, CA, 2000.