Efficient Monitoring Strategy for Active Environments

S. Kami Makki, Bo Sun and Ricky Guidry
Lamar University/Department of Computer Science
Email: {kami.makki, bo.sun, ricky.guidry}@lamar.edu

Jeffery Hill
University of Mary Hardin-Baylor /Department of Computer Science
Email: jdhill2@mail.umhb.edu

Abstract—**The new advances in sensor device technologies make Wireless Sensor Networks (WSNs) effective and economically viable solutions for a large variety of applications. Extensive research efforts have been devoted to various WSN research topics, such as target tracking, data collection and dissemination, localization, and sensing. In this paper, we focus on target tracking problems using WSN – one of the most challenging WSN research topics because of certain real-time constraints, energy consumption, and detection accuracy. We propose an intelligent and independent decision making strategy for sensors to monitor targets in an active environment. This algorithm has the following objectives: 1) Dynamically duty-cycle sensor devices to go to sleep in order to save energy; 2) Quickly wake up sensor nodes to ensure detection accuracy. We illustrate and analyze the details of this algorithm, and demonstrate its performance in terms of WSN lifetime with leader based algorithm.**

Index Terms—**Localization, Network, Scheduling, Synchronization, Tracking**.

I. INTRODUCTION

Recent revolutionary advances in wireless communications and miniature computing have made it feasible to build miniature wireless sensor nodes that integrate processors, sensors, memory, and wireless transceivers together.

Wireless Sensor Networks (WSNs), based on the collaborative efforts of a large number of sensor nodes, have become ideal candidates to provide effective and economically viable solutions for a large variety of applications. The wireless sensor devices are portable and inexpensive, but suffer from a number of drawbacks, including stringent energy supply, limited processing and communications capabilities, and scarce memory. All these factors still severely limit WSN applications. For example, in target tracking applications, a sensor node is usually required to identify a moving target within a specified time limit before the target moves out of the sensing range. Therefore, traditional systems may adopt always-on sensor devices in order to monitor potential targets. Unfortunately, this naive solution severely reduces the lifetime of WSNs because of their battery-powered nature. Therefore, how to conserve sensor node energy while maintaining detection response rate and accuracy poses significant challenges when designing a target tracking system using WSNs. There are a few ways to conserve battery life. One is to simplify the computations a sensor must carry out by implementing more efficient algorithms. Complex and longer computations can cause a sensor to exert more power. Also, the energy performance of a wireless sensor network can be improved by transmitting and receiving less radio messages, because it costs lots of energy to send and receive radio messages. An alternate method is to adopt low-duty-cycle WSNs, in which a sensor node is scheduled to be active for only a very brief period of time and then stays dormant for a long period of time in order to bridge the gap between stringent energy supplies and lifetime requirements [10]. In dormant state, usually only a timer is running to wake up the sensor node at a certain time for processing, receiving and sending messages. Duty-cycled WSNs, however, pose another challenge to WSN target tracking system design. That is, when a target appears and messages need to be forwarded to the base station for processing, the system may suffer from excessive delivery delay because potential relay nodes are still in sleep modes.

This material is based upon work supported by the National Science Foundation under Grant No. 0851912, 0848273 and 0922888.
Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

© 2011 ACADEMY PUBLISHER
doi:10.4304/jsw.6.4.536-543

Since the sender nodes need to wait for the receiver nodes to wake up in order to further transmit the ongoing packet. As a result, the target tracking systems in this category also require an efficient scheduling solution to meet detection time constraints.

Motivated by the above observations, we propose an intelligent and independent decision making strategy for sensors to monitor targets in an active environment, an algorithm that can accurately detect the targets while conserving energy consumption. This algorithm is designed to minimize communications between sensors since transmissions proven to be the most costly operation for a sensor [9]. Fewer communications may not only translate into more battery life, but may also allow for the sensor to spend more time on other operations. Therefore, our proposed algorithm contains stages such as learning, accumulation, and decision making. These three stages work together to form a target tracking systems.

The remainder of this paper is organized as follow: Section 2 discusses the importance of other areas in WSNs that are needed for target tracking. Section 3 examines past target tracking approaches as well as assumptions and definitions used within this paper. Section 4 goes step-by-step through our propose algorithm. Section 5 discusses the mathematical concepts that are used within the algorithm. In Section 6 the performance of our proposed algorithm was analyzed and compared with a traditional solution – the leader based solution. Section 7 concludes the paper and discusses potential future work to help improve the proposed solution.

II. RELATED WORKS

The field of target tracking is extremely broad and encompasses many other areas of WSNs as demonstrated in “A Line in the Sand” [4]. Localization, routing protocols and time synchronization are all important components that need to be incorporated in WSN tracking systems.

A. Localization

Localization is an important issue in target tracking, as sensors must be aware of their location and orientation. In addition, the sensors need to know the location and orientation of all their neighbor sensors. This allows sensors the ability to alert the exact neighbors from approaching targets.

Stoleru has presented general ways of localization depending on the needs of the WSN [2]. This type of ad-hoc localization would be ideal as it could also be used in conjunction with routing protocols for relaying information back to sink nodes and cluster heads.

B. Routing Protocols

It is important that routing protocols in WSNs produce as little or no loss of data as possible from one sensor to another. Therefore, it is important to have proper routing protocols to ensure that relevant information is sent in an efficient, timely manner to other sensors, sink nodes, and base stations [1]. There are various routing protocols to choose from, such as the data centric protocol, directed diffusion [5] or the hierarchical protocol LEACH [6].

The routing protocols require to be efficient and timely to ensure that the sensors in the system stay synchronized.

C. Time Synchronizations

Time synchronization algorithms help the system operate in an orderly manner. This helps sensors by limiting packet interference and possibly system deadlock. It also helps to eliminate redundant sensor information and allows for correct data fusion.

A few low cost, energy aware, time synchronization algorithms have been proposed [3, 11, 12]. The algorithm presented in [11] is a post facto time synchronization algorithm where nearby sensors can update their internal clock while in low power modes. The method provides easy time synchronization for a small system or subsystems. This approach would allow sensors to timestamp data on tracked targets to make sure other surrounding sensors have the most up to date information.

III. TARGET TRACKING APPROACHES

One of the more popular approaches in target tracking is to have a sensor act as a leader to certain subsets of sensors within the system. In leader based algorithms a leader sensor may be chosen statically or dynamically. A leader’s subset of follower sensors may be chosen statically or dynamically based on a number of metrics such as region or location, strength of received signals, latency of packet reception, or power reserves [14, 15]. In leader based algorithms all sensors currently tracking a target must report their readings to their respective leader. A leader sensor receives the data from his followers and calculates what actions need to be taken and then transmits directions to all of its followers. The directions may include orders for a sleeping sensor to wake up or in some cases a leader may need to alert its followers that it has passed leadership to another sensor.

There are various ways a new leader may be chosen in leader based algorithms. Some leader based approaches will simply have the current leader sensor pass leadership to another sensor under its command [8]. However, some proposed leader based algorithms will hold new leader elections every time a new leader is needed [7], either due to power constraints or the target is moving out of its subset of sensors. After leadership has been passed the previous leader sensor demotes itself back to follower status.

The past target tracking algorithms work well for low target density areas, but their overall lifetime deteriorates for high density target areas. We propose an algorithm that

© 2011 ACADEMY PUBLISHER

will maintain a lifetime in high target density areas. For our constructed simulator, the sensors have a battery life, a communication range, a detection range and a unique identification number. The sensors for our purposes also have two modes watch and wait. Furthermore, the sensors are able to determine the orientation and distance of targets within their detection range. Our proposed algorithm makes use of the following assumptions:

Assumption 1: A sensor's communication range is twice their detection range.

Assumption 2: Sensors only consume power while in watch mode.

Assumption 3: The system runs until there is a target that is not watched by a sensor.

Assumption 4: There is a pre-defined maximum speed for targets. Otherwise targets could potentially move through a sensor's entire detection range in one time instance.

Assumption 5: Each sensor in the system is on the same stage of the algorithm at any given time instance.

Assumption 6: Once the system has started running no new targets are introduced.

The last assumption does not need to be true for the algorithm, but the algorithm would then need to be adapted. This however is considered beyond the scope of this paper and is ignored. Throughout the remainder of this paper we will continually refer to certain terms which are defined as follows:

Definition 1: The system is defined to be the collection of all sensors.

Definition 2: Two sensors are said to be neighbors if both sensors can directly communicate with each other.

Definition 3: A sub-system is a set of sensors which are connected by fellow neighbor sensors.

Definition 4: A target is any object that can and needs to be tracked within the system. Targets have a variable speed that can accelerate from stationary to the pre-defined maximum speed.

Definition 5: The lifetime of a system is defined as the amount of time that the system can watch all the targets. When a target can no longer be tracked, usually due to a sensor running out of energy, the system is declared dead.

Definition 6: A necessary sensor is a sensor that contains a target that only it can watch.

Definition 7: The intermediary period is a time period within which events occur to keep the sensor system cooperative without jeopardizing collision of transmissions. The less the period is the more accurate readings are. However the system is less power-efficient and there is less room for error with regards to transmissions.

Definition 8: The buffer is the length away from the detection radius that a target will be handed off to a neighbor sensor. The buffer can be thought of as an annulus centered at the sensor with the larger radius being the detection range of the sensor and the smaller radius is the detection range minus the buffer as shown in Figure 1. Consider a sensor S, the thick bar labeled B is the buffer size. R2 is the radius of the sensor's detection range and R1=R2 – B. The larger the buffer, the faster the maximum speed of the targets can be, but the less power-efficient the system becomes.

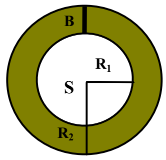

Figure 1. Bar labeled B is the buffer size

IV. THE PROPOSED INDEPENDENT DECISION MAKING STRATEGY

The main concept of proposed strategy is to allow the sensors to make their own decisions which will balance the power of the system. The end result increases the lifetime of the system in high density target areas. The algorithm will be divided into two sections. Section 4.1 will discuss the initialization of the system while Section 4.2 discusses the intermediary period and its three stages.

A. Initialization of the system

When system is initialized, every sensor must use localization to gather information about the location of its neighbor sensors. This localization process will allow each sensor to have the position and identification number of its neighbor sensors by constructing a table. This table contains the list of sensor's neighbors that will be used in the decision process. It will include the identification number, the position, and shared target data set that will be updated during each pass of the intermediary period. The information obtained during the initialization of the system is vital to our proposed algorithm.

B. The Intermediary Period

The intermediary period is the time when the steps of our proposed strategy for efficient usage of energy will be carried out. The period will need to be determined by the amount of sensors used within a system and their ability to communicate with each other without packet collision. There are three stages of the intermediary period: the Learning Stage, the Accumulation Stage, and the Decision Stage. Also, for the purposes of testing of our proposed strategy, it is assumed that all sensors within the system will be on the same stage at the same time. That is, the sensor system must maintain time-synchronization. However, this assumption can be relaxed since the decision making of the sensors for watching the targets solely depends on the detection of the targets by the specific sensors.

C. Learning Stage

The Learning Stage as shown in Figure 2 has three steps which are only performed by sensors who have decided to

© 2011 ACADEMY PUBLISHER

watch during the last intermediary period. The first step is to detect all targets within its detection radius. The next step is to determine whether those targets are shared with any of its neighbors. If any target is not shared with another neighbor, the sensor will declare itself "necessary". Finally sensor will transmit all pertinent data to its neighbors.

```
1. Begin
2.   if sensor is active
3.       target list = Detect Targets
4.    for each target in target list
5.       if target is not in neighbor sensor overlap
6.          necessary = true
7.    Transmit Pertinent Information To Neighbors
8. End
```

Figure 2. Learning Stage

Consider Figure 3 with sensors S1, S2, S3. Suppose sensor S2 has decided to watch during the last intermediary period (i.e. active) so it will go through the learning stage. It will execute the instructions on line 3 and 4 of Figure 2 to detect all the targets present in its detection range. Then it will execute the instruction at line 5 in Figure 2, to discover whether any of its targets are shared by its neighbors.

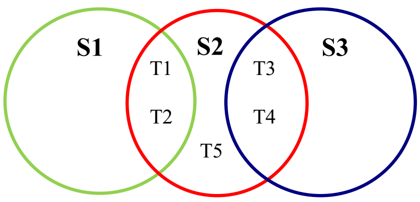

Figure 3. Example - Learning Stage

Suppose it detects that, there are five targets T1, T2, T3, T4, and T5 in its detection range, out of which it shares two targets T1 and T2 with S1, and two targets T3 and T4 with S3, and T5 is not shared by any of its neighbors. At this point, Sensor 2 will execute the instructions of line 6 in Figure 2 and declare itself necessary for targets T5. Finally the sensor will execute the instruction at line 7 in Figure 2 which will transmit all pertinent data such as, its identifier, the detected target information, the identifiers of the neighbors who share certain targets, and its own necessity status to its neighbors.

```
Begin
  Gather Pertinent Info Based On Received Data
   if sensor is idle and sensor targets > zero
       for each neighbor in neighbor list
           Transmit Pertinent Information To Neighbor
End
```

Figure 4. Accumulation Stage

D. Accumulation Stage

The Accumulation Stage contains two simple steps as shown in Figure 4. These steps are only executed by sensors that made the decision to wait during the last intermediary period. The first step is to use the collected information, received from the active sensors during the Learning Stage, to determine the amount of targets in its detection radius. The second step is to transmit all pertinent information to its neighbors which includes its sensor identifier as well as the number of targets in its detection range.

E. Decision Stage

The Decision Stage is the longest stage of the algorithm. Every sensor in the system will go through this stage to decide whether it needs to watch (become active) or wait. The Decision Stage will be broken down into two parts.

The first part is the primary decisions of this stage shown in Figure 5. During this part the sensor can categorize itself into taking one of three actions. If the sensor is necessary, it will make the decision to watch. If the sensor does not have any targets within its range, it will wait. Otherwise, it must go through the second part of the Decision Stage for "unnecessary" sensors.

```
Begin
   if sensor is necessary
         Watch
   else if sensor targets equal zero
         Wait
   else
          Decisions For Unnecessary Sensors
End
```

Figure 5. Decision Stage

The second part of the Decision Stage contains a number of steps, as shown in Figure 6. First, the sensor will create a pseudo-neighborhood using the neighbor and shared target information. The sensor traverses its list of neighbors eliminating neighbors and targets as follow: 1) Sensors are eliminated if they have been declared necessary in previous transmission, and they will be removed from the neighborhood list. 2) Targets are eliminated if they are watched by a neighbor that is eliminated. Then, if there are not any targets left to watch, the sensor will make the decision to wait. Also, if any neighbor has more targets than the remaining targets, the neighbor will be removed from the neighborhood list. However, if any neighbor has fewer shared targets than the number of remaining shared targets and does not have unshared targets, then the sensor will decide to watch. Another step is to deal with sensors with equal targets. The sensor will go through every neighbor that contains an equal amount of targets. If the sensor's identifier is higher than that of all other neighbor sensors that contain a target, then it will make the decision

© 2011 ACADEMY PUBLISHER

to watch. Otherwise, all sensors that have gone through the Decision Stage without making a decision will decide to wait.

Figure 7 illustrates the steps followed in the second part of the decision stage. In this figure there are 12 sensors such as S1, S2, S3, S4, S5, S6, ….., S12. In the first step, suppose S6 wants to find out, whether it is required to watch or wait. S6 will execute the instructions on line 2 of Figure 6 and will create a pseudo neighborhood list, using its neighbors and its shared target information and then obtain the target list which consists of the targets of all its neighbors and itself. Suppose the pseudo neighborhood list of S6 consists of the following neighbors: S1, S2, S3, S5, S7, S9, S10, and S11, with the list of shared targets of T5, T8, T9, and T12.

```
1. Begin
2.   neighborhood = new Pseudo-Neighborhood
3.   current sensor's target list = obtain target list
4.   for each neighbor in neighborhood {
5.       if neighbor is necessary
6.          remove neighbor from neighborhood
7.          remove neighbor's shared  targets from
                                      target list
8.   }
9.   if current sensor's target list count is  equal zero
10.        Wait
11. else{
12.      for each neighbor in neighborhood
13.         if neighbor target count > current sensor's
                                     target list count
14.            remove neighbor from neighborhood
15.     for each neighbor in neighborhood
16.         if neighbor targets count equal current
             sensor's target list count and
             sensor id > neighbor sensor id
17.             if  neighbor unshared target equal zero
18.                remove neighbor from neighborhood
19.         else
20.          current sensor's become active
     }
21. for all neighbors in neighborhood{
22.       if target moves out of detection range
23.          if target distance to neighbor >= buffer range
24.              current sensor's become active
25.          if target distance to neighbor < buffer range
26.                Wait
27. }
28. End
```

Figure 6. Decision For Unnecessary Sensors

The first check is performed at line 5 of Figure 6 to find out that if any of its neighbors is necessary senor. If any of its neighbors had transmitted that it is a necessary sensor, then that neighbor will be removed from the neighborhood list. (From Figure 7, target T18 is watched only by S10, therefore, sensor S10 had been declared as a necessary sensor and it will be removed from the neighborhood list of sensor 6.)

If there is no target left after removing the necessary sensor and its shared targets (i.e. line 6 and 7). Then the sensor waits, otherwise more checks will be performed. The next check is carried out at line 12. If any neighbor of S6 has more target than the remaining target list, then that neighbor is also removed from the neighborhood list, since that neighbor can be a watch sensor when it is tested as the current sensor.

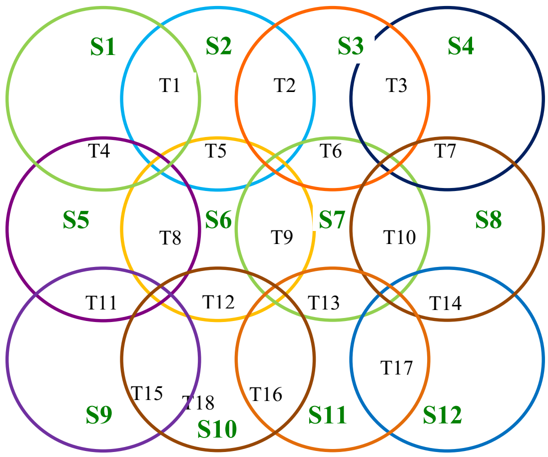

Figure 7. Example - Unnecessary sensors

Finally there will be another check for targets which go out of detection range of a sensor. In this case suppose target T4 goes out of the detection range of S1 and moves toward sensor S6. If the distance of the target T4 from S1 detection range is greater than or equal to the buffer range of sensor S1, then S6 will become active otherwise S6 will wait.

V. Measuring Target Location and Detection

A. Distance of a Target to a Neighbor

In a sensor environment, any sensor S can determine if it is sharing any target T with any neighbor sensor N. Let u be the vector from S to T and let v be the vector from S to N, as shown in Figure 8. Then the following formula will calculate the angle gamma between the two vectors. $\gamma = \cos^{-1}\left(\frac{u \bullet v}{\|u\| * \|v\|}\right)$.

Now we let a represents the magnitude of u and let b represent the magnitude of v. Then let c be the distance from T to N. Combining with the newly obtained angle gamma the Law of Cosines can be applied to determine the value of c, where $c = \sqrt{a^2 + b^2 - 2ab\cos(\gamma)}$. Now the sensor

© 2011 ACADEMY PUBLISHER

will check to see if c is smaller than the detection range of N. If it is then N can detect T, and T is a shared target. Otherwise T is outside of N's coverage area and is not a shared target.

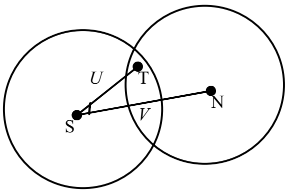

Figure 8. A diagram illustrating the relationship between a sensor S, neighbor N and target T

B. The Buffer

Given a sensor that is currently watching a target the minimum probability of the target being detected as it passes through the buffer can be expressed as W_{Buffer} (the width of the annulus) divide by V_{Target} (the maximum speed of the target per unit of time). Then the result multiplied by $F_{Frequency\text{-}Of\text{-}Period}$ (the number of intermediary periods per second).

$$P = \left(\frac{W_{Buffer}}{V_{target} \times F_{Frequency-Of-Period}} \right)$$

For example, if the checks are made every one second, the buffer has a width of eight meters and the target is moving ten meters per second the probability would be determined as follows:

$$\left(\frac{8m}{10 \, {}^{m}\!/\!{}_{s} \times 1Hz} \right) = 0.8$$

That is, the probability of detecting target in the buffer during an intermediary period is 80%. If the buffer width is less than the targets velocity, then there is a chance the target could become lost as the sensor will not alert any of its neighbors to turn on and watch the target.

It should also be noted that the frequency of the intermediary period also is a factor when computing the probability that a target will be detected passing through the buffer. The larger the frequency translates to the target being detected in the buffer for multiple intermediary periods. While not necessarily a detriment to the system it can prove costly to a sensor's power supply. For this reason, the more frequent the intermediary period, the smaller the buffer needs to be. For example, if the frequency of the intermediary period is 4, the probability would then be

$$\left(\frac{8m}{10 \, {}^{m}\!/\!{}_{s} \times 4Hz} \right) = 0.2$$

That is, the probability of detecting target in the buffer during an intermediary period is 20%. Obviously in this example the frequency of the intermediary period is high in relation to the width of the buffer. Therefore the size of the buffer and the frequency of the intermediary period are inversely proportional. Ideally the width of the buffer multiplied by the frequency would be as close to the target's velocity as possible for maximum efficiency. This would ensure that a target would be detected in the sensor's buffer and guarantee that the target is not lost. This also allows sensing and updating target information to be done as little as possible.

VI. SIMULATION

A. Simulator

The proposed algorithm was tested using a simulator implemented in the Java programming language. The simulator was built from the ground up using the basic concepts of WSNs.

For the purpose of our simulator targets are all the same size and have random path movements. Targets have uniform size of 1.

The sensor system consisted of placing fifty-four homogenous sensors with a detection range of 80 in a 9x6 grid covering an area of 720x480. This is ensures that a target could be sensed by at least one sensor at all times in the simulation, as shown in Figure 9.

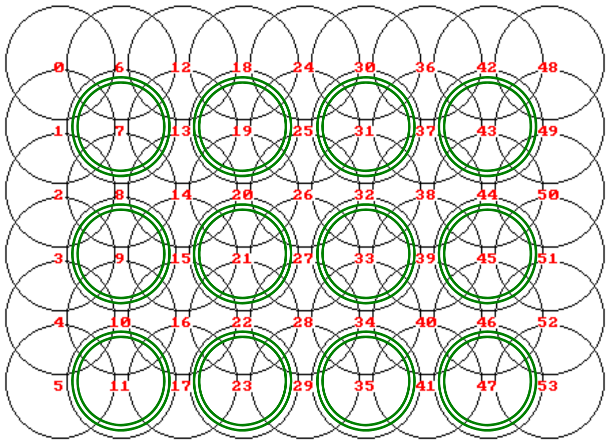

Figure 9. The grid of sensors arranged in a 9x6 format. The numbers in each sensor indicate the location of the sensor and the ring represents the radius of detection

A traditional leader based approach was also implemented in the simulator to compare results against the proposed algorithm. For the leader based approach an additional twelve sensors were added to act as leaders. The leader sensors were placed on top of S7, S19, S31, S43, S9, S21, S33, S45, S11, S23, S35, and S47 sensors and they are highlighted with double lines in Figure 9. These sensors were not used for sensing. The subsystem which a leader sensor was responsible for consisted of all its neighbors. The leader sensors remained as the leader sensors through the lifetime of the system.

© 2011 ACADEMY PUBLISHER

B. Comparison to Leader Based Approach.

As seen in Figure 10 the merits of the proposed strategy against that of leader based approach are more noticeable in scenarios where there are more targets.

In the leader based approach, leader sensors will make fewer transmissions, which will allow for less reception strain on the sensing nodes. However this leads to an increased reception count for leader sensors. The increased reception count will make the leader exhaust its power supply much quicker as shown in Figure 10.

The cost of reception for a sensor is 0.10 and 0.12 for transmission, and the cost of detecting a target is 0.10 [10, 6]. Suppose there is a leader in-charge of eight sensors, which are all necessary sensors. Each of the eight sensors will waste 0.22 per interval. However, the leader will spend 0.80 for all the receptions per interval. Therefore, the usage of the power is not balanced among these sensors, and the power usage of the system has become inherently off-balance.

Now suppose four sensors are on and each sensor is watching a target. Each sensor has neighbor which is in wait mode and could also detect the target. Next the targets begin to move out of each watching sensor's range and into the range of their waiting neighbor. The leader must send a message to tell the four sensors watching a target to enter wait mode and then broadcast a watch order to the four neighbor sensors. The individual cost to the leader is 0.40 for reception and 0.96 for transmission per interval. This comes to a total consumption of 1.36. In the case of a dynamic leader based approach, the leader could be switched off, but then a new leader must be elected. The election process would need to consume some sensors' energy as well and this could result in a shorter system lifetime [13].

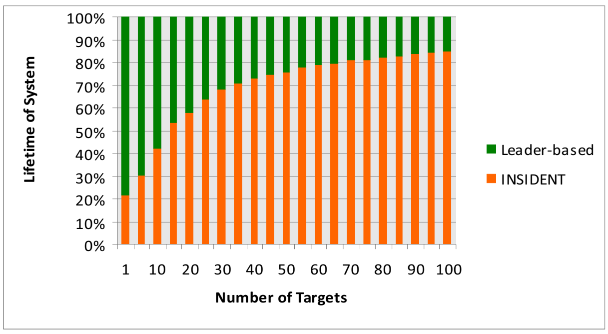

Figure 10: The lifetime of the system using proposed algorithm compared with Leader-based approach.

According to Figure 10 the proposed algorithm begins to outperform the leader based algorithm around 20 targets. This is also when the target to sensor ratio for the system is approximately 1:4. As the targets become more numerous the sensor system using our algorithm is shown to have a longer lifetime. Any overhead incurred by a leader is non-existent in our algorithm. Our algorithm allows for the sensor system to function properly without a leader, or cluster head. In a static leader based approach if the leader fails the system fails.

VII. CONCLUSION

The traditional leader based approach delivered better system lifetimes when there were fewer targets. However the proposed algorithm was shown to outperform a traditional leader based approach in scenarios with numerous targets. Our algorithm's low system lifetime for fewer targets is overshadowed by the consistent performance when the number of targets increases.

The next objective would be to combine our algorithm with a routing protocol such as Directed Diffusion. This would allow information on targets to be fused together and sent to a base station for further analysis, such as predicting paths or target identification. However, the use of path prediction with multiple targets is only possible when targets can be uniquely identified.

ACKNOWLEDGMENT

The authors are very grateful for the anonymous reviewers' comments that helped significantly improve the readability of the paper.

REFERENCES

[1] M. Chu, H. Haussecker, and F. Zhao, "Scalable Information-Driven Sensor Querying and Routing for ad hoc Heterogeneous Sensor Networks," Xerox Palo Alto Research Center Technical Report P2001-10113, May 2001.

[2] R. Stoleru, J. Stankovic, and S. Son, "Robust Node Localization for Wireless Sensor Networks". In the Proceedings of the Fourth Workshop on Embedded Networked Sensors (EmNets'07), June 25-26, 2007, Ireland

[3] R. Akl and Y. Saravanos, "Hybrid Energy-Aware Synchronization Algorithm in Wireless Sensor Networks." In the Proceedings of the 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'07), September 3-7, 2007, Greece.

[4] A. Arora et al., "A Line in the Sand: A Wireless Sensor Network for Target Detection, Classification, and Tracking". Computer Networks, Vol. 46, No. 5, Dec. 2004, pp. 605–34.

[5] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed Diffusion for Wireless Sensor Networking". IEEE/ACM Transactions on Networking (TON), Vol. 11, No. 1, 2003, pp. 2-16

[6] Ioan Raicu, "Efficient Even Distribution of Power Consumption in Wireless Sensor Networks." Eighteenth International Conference on Computers and their Applications (ISCA'03), 2003.

[7] Liqian Luo, Tarek Abdelzaher, Tian He, and John A. Stankovic, "Design and Comparison of Lightweight Group Management Strategies in EnviroSuite." In the Proceedings of the International Conference on Distributed Computing in Sensor Networks (DCOSS'05), 2005.

[8] F. Zhao, J. Shin and J. Reich, "Information-Driven Dynamic Sensor Collaboration for Tracking Applications," IEEE Signal Processing Magazine, March 2002.

© 2011 ACADEMY PUBLISHER

[9] C-Y. Chong, F. Zhao, S. Mori, and S. Kumar, “Distributed Tracking in Wireless Ad Hoc Sensor Networks.”

[10] H. Liu, X. Jia, P-J. Wan, C-W. Yi, S. K. Makki, and N. Pissinou, “Maximizing the Lifetime of Sensor Surveillance Systems,” IEEE/ACM Transactions On Networking, Vol. 15, No. 2, April 2007.

[11] J. Elson and D. Estrin, “Time Synchronization for Wireless Sensor Networks”. UCLA CS Technical Report 200028.

[12] M. Sichitiu and C. Veerarittiphan, “Simple, Accurate Time Synchronization for Wireless Sensor Networks”. In the Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC’03), March 2003.

[13] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, “Adaptive protocols for information dissemination in wireless sensor networks,” In Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking, August 1999.

[14] Masoumeh Karimi and Deng Pan, “Challenges for Quality of Service (QoS) in Mobile Ad-Hoc Networks,” IEEE Wireless and Microwave Technology Conference (WAMICON’09), April 2009.

[15] W. Chen, and J. Hou. "Dynamic clustering for acoustic target tracking in wireless sensor networks", IEEE Trans. on Mobile Comp., vol. 20, pp. 258-71, 2004.

© 2011 ACADEMY PUBLISHER