Wireless Sensor Network, **2009, 1, 370-382**
doi:10.4236/wsn.2009.15046 Published Online December 2009 (http://www.scirp.org/journal/wsn).

Target Tracking with QoS Support in Large Wireless Sensor Networks

Ghasem Naddafzadeh SHIRAZI, Peijie WANG, Xiangxu DONG, Chen Khong THAM
Department of Electrical and Computer Engineering
National University of Singapore, Republic of Singapore
Email: {*naddafzadeh, g*0700293*, dong*07*, eletck*}*@nus.edu.sg*
Received July 15, 2009; *revised August* 10, 2009; *accepted August* 17, 2009

Abstract

Quality of Service (QoS) is important in the application of target tracking in wireless sensor networks (WSNs). When a target appears, it will trigger an event from one or more sensors. A target can only be accurately detected if a certain number of event packets are received by the sink in a predetermined detection time interval. In this paper, we propose a buffer management scheme based on event ordering to achieve QoS. We also propose a directional QoS-aware routing protocol (DQRP) for the dissemination of the event ordering list. After the dissemination, a priority queue buffer management scheme is used to ensure QoS. Our buffer management scheme works in conjunction with DQRP to ensure accurate as well as energy-efficient target detection in the presence of multiple targets. The novelty of our network architecture is that a distributed admission control scheme is implemented on each node based on a geographic routing algorithm. In our scenario, a target can only be accurately detected if a certain number of event packets are received by the sink in a predetermined detection time interval. Our main performance metric is the number of targets/events being detected. Our protocol maximizes the number of targets being detected.

Keywords: Target Tracking, QoS, Multi-Sink Wireless Sensor Networks.

1. Introduction

With the advancements in wireless communications and the development of small electronic sensing devices, wireless sensor network (WSN) technology was greatly developed in the past few years. A WSN comprises a large number of densely deployed sensing devices to sense the phenomenon or the occurrence of an event. Sensor nodes may be required to do data aggregation and fusion locally. More importantly, the sensor nodes are required to report their measurements to the sink. Similar to the traditional end-to-end networks, communications in WSN suffer from delay and loss. Quality of Service (QoS) support is required to ensure the performance of a network. The QoS in traditional computer networks is generally defined as the performance level of a network service offered to the user, therefore QoS centers on network quantities such as delay, loss and reliability.

However, the communication in WSNs is data-centric and non end-to-end. In WSNs, the main consideration is the number of packets the sink receives about an event and not how many packets the sink receives from any particular sensor node. Therefore, QoS in WSNs is different from the traditional end-to-end networks. Furthermore, depending on different applications, the QoS in WSNs may also include coverage, accuracy, etc. Moreover, minimizing energy consumption is another important consideration in WSNs. Once sensor nodes are deployed in the physical area, it is hard to recharge them. Since replacement is costly, energy efficiency is highly demanded in WSNs.

2. Related Work

QoS in WSNs contains a wide range of issues. In the literature, there are many papers focusing on the explicit definition of QoS in WSNs as well as the specific implementation of QoS. Reference [2] serves as a good survey of the QoS support in WSNs. It generally describes the QoS in WSNs in two different perspectives; one is application-specific QoS and the other is network QoS.

Depending on different applications, the application-specific QoS can be defined using parameters such as coverage, measurement accuracy, and optimum number of active sensors.

From the perspective of network QoS, the main purpose is to efficiently utilize the network resources to deliver the QoS-constrained data. The QoS metrics can be defined as transmission delay or packet loss. Reference

[1] proposes using energy efficiency, system lifetime, latency, accuracy, fault-tolerance and scalability to evaluate sensor network protocols. It also defines the architecture of a sensor network and many basic models in WSNs, such as the communication models, the data delivery models and the network dynamic models. These models can help us to define the QoS metrics more precisely. In [3], the authors present a QoS control for WSN. They assume a broadcast channel for the base station to dynamically control the number of active sensors in virtue of the Gur Game mathematical paradigm. The QoS here is defined as the optimum number of active sensors in the network. Reference [4] devotes to quantify the tradeoff between power conservation and quality of surveillance in target tracking WSNs. It also provides guidelines for efficient deployment of sensor nodes for target tracking applications.

Our motivation is to perform an accurate detection and tracking of moving objects in a WSN where multiple events occur with variable inter-arrival times. We consider a target tracking application in which the events are *appearance* or *movement* of the targets. Some previous works on target tracking focus on the coverage problem [4,5]. A general assumption has been made that once the sensing range of senor nodes covers the trajectory of the moving target, the target can be tracked. Reference [6] proposes to use quality of monitoring (QoM) to ensure a high reporting accuracy in the presence of noises and signal attenuation. The authors of [6] state that both QoM and coverage need to be taken into account when solving target tracking problem.

Instead of restricting the tracking problem in coverage or QoM, we establish the problem from the perspective of network consideration. Accurate tracking and location estimation of a mobile target may require a minimum number of packets to be received by the sinks. It is not enough just to ensure coverage. Loss of data packets will result in loss of accuracy when estimating the actual locations of the targets. Therefore, QoS support is required in order to ensure accurate target tracking.

3. Challenges in Providing QoS in Target-Tracking WSNs

3.1. Definition of QoS

We explicitly define the QoS requirement in the application of target tracking in WSNs as:

A target can only be accurately detected if a certain number of event packets are received by the sink in a predetermined detection time interval.

Our main performance metric is the *number of targets/events being detected*. Our protocol aims to *maximize* the number of targets detected with the events satisfying the user's QoS requirements.

3.2. Problem Description

Traditional QoS schemes used in computer networks, like Intserv or Diffserv [7], do not work well in our problem. This is because the target moves, and the sensor node which is responsible for transmitting the event packets changes. Therefore, it cannot be known in advance which sensors or the resources required to maintain QoS. Besides, reservations of resources require the knowledge of capacity of the network which is difficult to achieve in dynamic WSNs. To solve the above problems, we propose a QoS-aware network architecture designed to be used in multi-hop and multi-sink WSNs. Our network architecture consists of two components, a buffer management scheme and a novel QoS-aware routing algorithm named **D**irectional **Q**oS **R**outing **P**rotocol (DQRP). The main objective of DQRP is to support the use of a distributed admission control scheme for WSNs.

We further illustrate the problem of target tracking if no QoS support is used by carrying an experiment using MICAz motes. The scenario is illustrated in Figure 1. There are two source sensor nodes which have detected one target each and they are sending packets at a rate of 20 packets/s to the sink through an intermediate sensor node. The size of each data packet is 20 bytes excluding the various headers. The sensor which detects event 1 sends data from t=0 to t=180 while the sensor which detects event 2 sends data from t=60 to t=240. The sampling interval is 5 seconds and an event is only detected when 90% of the data packets are received between 2 sampling periods.

Figure 2 shows the results obtained. From the graph, we note that event 1 can always be detected when event 2 has not occurred. When there are two events, there are times when none of the events can be successfully detected (e.g. from t=60 to 80 and from t=160 to 180) although the total throughput remains fairly high. This illustrates the need for a buffer management scheme to give priority to certain events to ensure that we can maximize the number of events detected.

Our buffer management scheme on each sensor node can do priority discarding of the data packets whenever it encounters congestion by giving higher priority to the event which happens earlier. The main remaining challenge

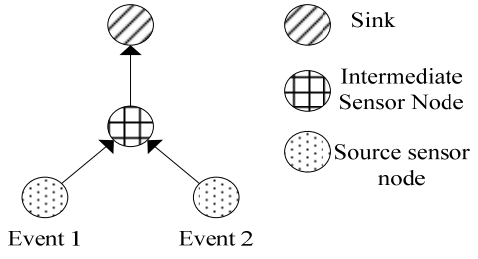

Figure 1. Scenario setup.

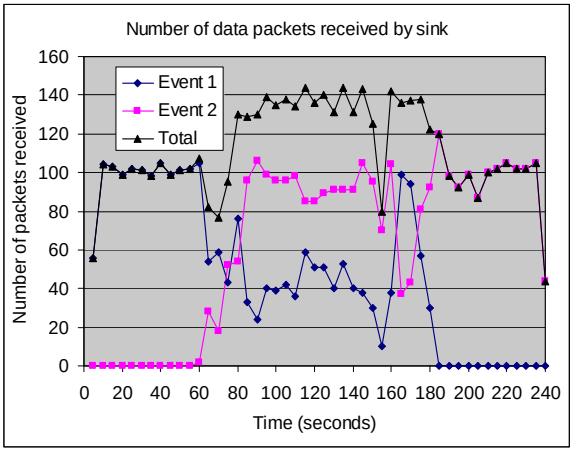

Figure 2. Experimental results.

is how to construct an event ordering list at each sensor node in a distributed way.The solution would be simple if global information is available. Each event packet can be associated with a global timestamp which is the time in which the event first occurs. When congestion happens, the packet with largest timestamp is dropped because we give priority to earlier events. However, this requires global clock synchronization which is hard to achieve in practical WSNs. Since WSNs are deployed over long periods of time, time drifts are a very serious problem because cheap and inaccurate clocks are usually used in sensor nodes.

Another possible solution for this problem is to use broadcasting. We can divide the communication into two phases. In the first phase, whenever a new event happens, the sensor node that detects it first broadcasts a new control message indicating that a new target has appeared and therefore there is a new event. All the nodes which receive the broadcast control message would insert the new event into its event ordering list. All the events that arrive later would have lower priorities. In the second phase, data packets are transmitted. Whenever a node encounters congestion, it would discard the data packet with lower priority, i.e. discard the data packet from the events which happen later.

The authors of [8] propose an efficient broadcast scheme for WSN called Broadcast Protocol for Sensor networks (BSP). The BPS uses an adaptive-geometric approach to reduce the number of retransmission by maximizing each hop length. In ideal BPS, the whole network area is covered by numbers of identical hexagons, where the length of the side of the hexagon is the node transmission range. Reference [8] tries to avoid these retransmissions by defining a transmission threshold *Th*. If a node overhears another node within distance *Th* has transmitted one packet, it would not retransmit the same packet. *Th* is a very important parameter in BPS as it represents the tradeoff between numbers of retransmission (redundancy) and delivery ratio (reliability).

In WSNs, the limited energy is usually expected to be used to transmit useful information (data packets). Therefore, the redundancy in control messages is not a good solution especially in our case where multiple events can happen intermittently. If the new events' arrival rate is high, it would lead to a large amount of redundant control messages occupying the nodes' buffers; therefore the number of events that can be correctly detected would reduce significantly in this case due to congestion.

Another reason we use unicast transmission instead of broadcasting in our proposal is that unicast may be more reliable than broadcasting. Assume the nodes density is D, the channel loss probability is p_l, and let the transmission range of sensor node be t. Suppose the unicast mechanism retransmit a packet for at most k times in case of packet loss. Thus the probability of a packet is lost at a node is given by

$$p_{bl} = p_l^{a_n} \tag{1}$$

in broadcasting, where $a_n = \pi t^2 D$ is the expected number of neighboring nodes of the destination node, and

$$p_{ul} = p_l^{k} \tag{2}$$

in unicast. Therefore, it is clear that unicast will be more reliable than broadcasting if $k > a_n$ is satisfied.

Our solution avoids the use of broadcast packets by making use of geographic routing algorithm to disseminate event ordering information using data packets. Each data packet consists of the event ID and therefore no additional control messages are required. Geographic routing uses nodes' locations as their address, and forwards packets in a greedy manner towards the destination. The greedy manner means that a packet is only forwarded to a node when it is closer to the destination than the current one. References [10] and [11] are two well known proposals on geographic routing algorithms. Our proposed geographic routing algorithm makes use of an angle to implement the greedy routing. We ensure that the density of our network is sufficient so greedy geographic routing works most of the time. The choice of angle would ensure that most of the nodes in the WSN covered area are able to receive packets from every event. These data packets containing the event ID are an important medium for the nodes to know the right ordering of events.

4. System Model

Our main goal is to maximize the number of targets being detected in a WSN where multiple targets appear with different inter-arrival times. If the rate of packets received is less than the desired rate, the event is considered *lost*, and the system is not able to detect it. In the

following subsections, we describe different aspects of our model, namely application, medium access control (MAC), and physical (PHY) levels, as well as transport, routing, and scheduling protocols. A distributed admission control scheme is then proposed in Section 5.

Each target is uniquely identified by a target ID, *i*. The i^{th} target, T_i, causes packet generation at a rate of r_i in the sensor nodes which detect it. Each event requires a packet delivery requirement of d_i. Therefore a minimum packet rate of $d_i r_i$ is required at the sink for detection of the event.

The choice of MAC protocols and physical layer characteristics (PHY) affect the capacity of the network, *C*. However, it does not affect our goal of maximizing the number of detected events given that the network capacity is *C*. A higher *C* would lead to more events detected and vice versa. Therefore, our QoS network architecture does not assume the use of any particular MAC or PHY.

Our novel routing algorithm, DQRP, ensures that each packet takes a different route to the sink such that after β packets have been sent by the source, all the sensors in the network will learn of the new event. Angle geographic routing is used and each data packet is given an angle of routing x degrees. DQRP is further explained in Section 5.

Each node maintains *b* buffers. Congestion occurs when the number of packet arrivals exceeds the number of buffers available, therefore buffer overflow occurs. Each node maintains an event ordering list. Earlier events are given higher priority when deciding which packet to drop in the event of buffer overflow.

We consider a multiple sink architecture in which sinks are able to share their received data. Therefore, sensor nodes can choose any of the sinks as the destination. Moreover, we assume that the location of sinks and neighboring sensors (i.e. sensor nodes within the transmission range) are known. The arrival of events models a Poisson Arrival Process. The targets stay in the WSN for a period which follows an exponential distribution. Only one sensor sends data packets to a sink for an event at any given time.

5. Proposed QoS-Aware Network Architecture

The main objectives and desired properties of our proposed protocol are given as follows:

1) Path diversity: failure of a sink or failure of any sensor node degrades performance gracefully. Any malicious or selfish sensor node can only degrade performance gracefully.

2) Maintaining QoS: in event of congestion, earlier events will get higher priority. This means that if event *i* occurs before event *j*, if event *i* does not achieve the required QoS, event *j* also does not achieve the required QoS.

3) Event Priority: In the event of congestion, event *i* will take priority over event *j* if event *i* occurs before event *j*.

A formal definition of an event being detected is given as: *An event is detected at time t if $d_i r_i$ packets are received in the interval [t-t_{min}, t] where t_{min} is the sampling interval, d_i is the delivery ratio required by the user and r_i is the sending rate of the event.*

This means that an event *i* is only detected at time *t* if the sinks receive a minimum number of packets defined by the QoS requirements of the user. The main performance metric is the number of events detected at a sampling time and our goal is to maximize the number of events detected. In addition, delay, throughput (total amount of traffic received by the sinks), and energy consumption should be considered as auxiliary performance metrics. Note that high throughput does not imply that the number of events detected is high, as illustrated in Figure 2.

5.1. Priority Buffer Management

We need to implement a priority buffer management scheme in every node to ensure that if congestion occurs and if we have the global event ordering list, we can decide to give priority to certain events. Our priority buffer management scheme is very efficient as insertion and removal takes O(1) time.

5.2. DQRP Protocol

In this paper, we propose using a novel directional QoS routing protocol (DQRP) to disseminate event ordering information. In DQRP, the detection of an event is reported to the entire network in order to provide QoS management in case of congestion. More specifically, when a target is first detected, the corresponding sensor sends different data packets via different routes to the multiple sinks. These routes are determined in such a way that all nodes are informed about the existence of the event after a certain time. The rationale behind informing the entire network is to provide the network with correct global event ordering, and hence enabling distributed admission control in the network.

To ensure that the maximum number of nodes can be informed in an efficient way, it is required to find routes to the sinks with the following two properties:

P1. Each node should be at least in one route.

P2. Number of shared nodes between every two routes should be minimized.

In other words, a new data packet should inform a new node about the event. Figure 3 shows two different set of

routes. The left diagram on Figure 3 shows that some of the intermediate nodes are not included in any route. In contrast, the right diagram in Figure 3 shows all nodes are included in at least one route.

In general, for a network topology where there are 4 sinks located at the edges of the network as shown in Figure 3, we can divide the whole area into four quadrants. The key challenge is to determine the number of packets to send to each sink and the paths of those data packets should cover most of the nodes in the rectangle region. Figure 4 shows a general situation of the four quadrants in a 2 dimensional space. Assume the source node locates at the origin (0,0), the destination sink locates at (x,y). Let the transmission range be t, and $x=m't$, $y=mt$. First, we have the following lemma,

Lemma 1 The largest area of A_i, depending on m', occurs at

$$r_{m'} = m't \text{ or } r_{m'+1} = (m'+1)t \tag{3}$$

where $A_i=I(C(Rj))-I(C(Rj-1))$. $C(Ri)$ is the circle with radius $R_i = it$ and $I(C)$ is defined as the intersected region of C with the rectangle. The value corresponding to $Max(A_i)$ is (whichever is larger):

$$\begin{aligned} & A^* = \pi t^2(2m-1)/4 \\ or \quad & A^* = [r_{m'+1}{}^2 \sin^{-1}(x/r_{m'+1}) - r_{m'}{}^2 \sin^{-1}(x/r_{m'}) \\ & + x\sqrt{r_{m'+1}{}^2 - x^2} - x\sqrt{r_{m'}{}^2 - x^2}\,]/2 \end{aligned} \tag{4}$$

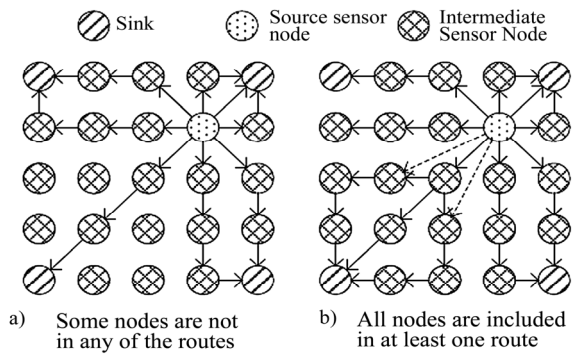

Figure 3. Two different set of routes from the source node to the sinks.

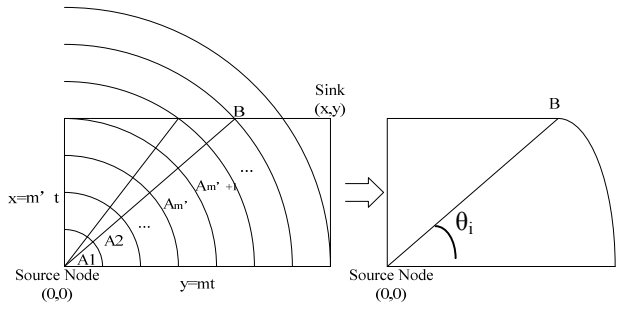

Figure 4. Intersection between transmission range spheres and the rectangle between the source (u) and the sink (s).

Proof: Denote area by | |. If $Ri \le mt$, the area of A_i is given by

$$\begin{aligned} A_i &= (C|R_i| - C|R_{i-1}|)/4 \\ &= \pi t^2(2i-1)/4 \end{aligned} \tag{5}$$

If $Ri > mt$, then the area is computed by dividing it to an arc and a right triangle, and is given by

$$A_i = (|Arc(R_i)| + |T(R_i)|) - (|Arc(R_{i-1})| + |T(R_{i-1})|) \tag{6}$$

where T(Ri) is the right triangle with hypotenuse equal to Ri and Arc(Ri) is I(C(Ri))- T(Ri). It is easy to find $|Arc(R_i)|$ and $|T(R_i)|$, and they are given by

$$\begin{aligned} |Arc(R_i)| &= r_{m'}{}^2 \sin^{-1}(x/r_i) \\ |T(R_i)| &= x\sqrt{r_i^2 - x^2}\,]/2 \end{aligned} \tag{7}$$

Substituting (5) in (4) gives

$$\begin{aligned} A_i &= [r_{i+1}{}^2 \sin^{-1}(x/r_{i+1}) - r_i^2 \sin^{-1}(x/r_i) \\ &+ x\sqrt{r_{i+1}{}^2 - x^2} - x\sqrt{r_i^2 - x^2}\,]/2 \end{aligned} \tag{8}$$

By taking derivatives of (5) and (8), it turns out that (5) is a increasing function of Ri, while (8) is a decreasing function of Ri. Thus, the maximum occurs either when $r_i = r_{m'} = m't$ (max. of (5)) or when $r_i = r_{m'} = m't$ (max. of (8)).

Next, we present our general angle dividing algorithm. The proposed angle dividing algorithm guarantees that in each routing area that divided by angle $\alpha_1, \alpha_2, ..., \alpha_l$, the largest number of nodes that need to receive distinguishing data packets are close to and bounded by

$$a_{upp} = S_{upp} D \tag{9}$$

where a_{upp} denotes upper bound of the number of nodes, S_{upp} denotes the upper bound of area of the region, and D is the nodes density of the WSN. We use A to denote region, S to denote area and assume the transmission range is t. The algorithm is presented in Algorithm 1. We explain Algorithm 1 by a simple example in Figures 5,6. The main idea is that the angles should be chosen such that the shadowed regions share the same area S_{upp}. It should be noted that the value of S_{upp} will affect the delivery ratio as well as the convergence speed of our angle routing algorithm. A larger value of S_{upp} allows larger angles $\alpha_1, \alpha_2, ..., \alpha_l$ to exist. A larger angle may increase the delivery ratio in the sense that if a data packet for a node in the shaded area is lost, other packets that share the same path before reaching the shaded area can serve as the informers. For example, in Figure 6, node 4 and node 5 will share the same intermediate node, node 0 and node 1. If a packet being sent to node 5 is lost at node 0, node 0 and node 1 can still learn

1. Suppose y is always the longer side of the rectangle. Let m=int(y/r), m'=int(x/r) and r_i =ir, $i=1,2,...,m$.

2. Compute $S_{A_m} = (S_{ARC_m}+S_{TRI_m})- (S_{ARC_{m-1}}+S_{TRI_{m-1}})$. In general, we have $S_{A_i} = (S_{ARC_i}+ S_{TRI_i}) -(S_{ARC_{i-1}}+ S_{TRI_{i-1}})$.

Where $S_{ARC_i} = \theta_i \ r_i^2/2$. $S_{TRI_i} = x\ \sqrt{r_i^2 - x^2}/2$, $\theta_i = \sin^{-1}(\frac{x}{r_i})$

3. k = max i such that $S_{A_k} > S_{upp}$.

4. $\alpha_0 = 2\ S_{upp}/(r_k^2 - r_{k-1}^2)$.

5. While $S_{A_k} > S_{upp}$ do {

$S_{A_k} = S_{A_k} - S_{upp} = S_{A_k} - \alpha_j(r_k^2 - r_{k-1}^2)/2$

$S_{A_{k-1}} = S_{A_{k-1}} - \alpha_j(r_{k-1}^2 - r_{k-2}^2)/2$

.....

$S_{A_1} = S_{A_1} - \alpha_j r_i^2/2$

$\alpha_j = \alpha_0$, j ++.

}

7. While $i > m'$ do {

i--;

If $S_{A_i} < S_{upp}$ .continue;

Else {

$\alpha_j = \theta_i - (\alpha_0 + \alpha_1 + ... \alpha_{j-1})$

$S_{A_i} = S_{A_i} - S_{upp} = S_{A_i} - \alpha_j(r_i^2 - r_{i-1}^2)/2$

$S_{A_{i-1}} = S_{A_{i-1}} - \alpha_j(r_{i-1}^2 - r_{i-2}^2)/2$

.....

$S_{A_1} = S_{A_1} - \alpha_j r_1^2/2$

j ++;

}

}

Algorithm 1. DQRP path-finding protocol.

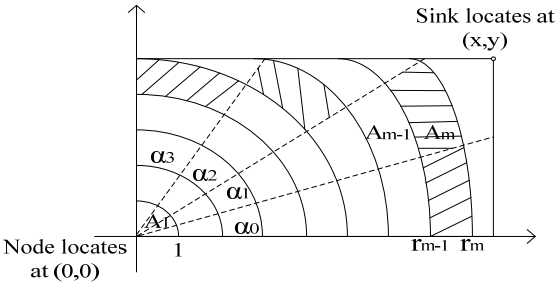

Figure 5. An example of the angle dividing algorithm.

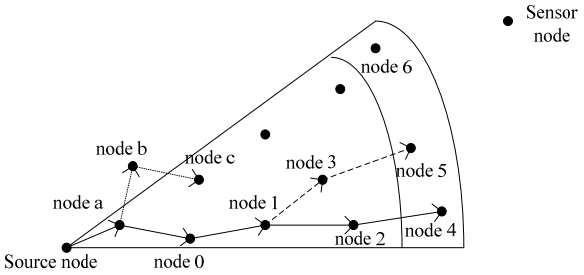

Figure 6. An angle routing example

the occurrence of the event by delivering the packet sent to node 4. In addition, as greedy routing is used with a larger angle, it is more likely that a node can find its subsequent node in that angle's region. This is the case of node c in Figure 6, which is out of node a transmission range, thus node b will help deliver the packet.

However, if the angle is too large, it will degrade the performance of the algorithm. As the routing information is given by angle, the node who has a packet to transmit does not know exactly which next node to pass the packet. It will choose any node that is valid to pass the packet.

In Figure 6, node 1 may pass the packet to either node 2 or node 3. Thus, node 4 may receive the packet twice while node 5 may miss the packet. In this case, we say that node 5 is node covered.

Finding the appropriate value of *Supp*

Note that a_{upp} determines the maximum number of nodes in a single hop region of each route. For example, if we set Supp to A* = **max**(SAm` , SAm`+1), then according to our old proposed algorithm the maximum possible value is **max(**a_{upp}**) = D·A*** . To analyze in more detail, we need to find the approximate area which is covered by each route, $\boldsymbol{A_R}$.

Firstly, it can be easily seen that the number of hops in each route to the sink is $H = \sqrt{m'^2 + m^2}$. Secondly, the area in each hop is bounded by Supp. This gives the following upper bound

$$A_R < HS_{upp} \tag{13}$$

The number of nodes in each route is then given by $\mathbf{N_R} = \mathbf{A_R \cdot D}$. If we apply the above upper bound, then there are approximately

$$\begin{aligned} N_R &= \sqrt{m'^2 + m^2}\, S_{upp} D \\ &= H S_{upp} D \end{aligned} \tag{14}$$

nodes in each route. Recall the properties of a "*good*" set of routes, i.e.

P1. Each node should be in one route from node u to sink s.

P2. Number of mutual nodes between routes should be minimum.

We find out that the minimum number for $\max(a_{upp})$ should be 1. By setting Supp=1/D, we get the minimum of $\max(a_{upp}) = (1/D)D = 1$, which satisfies P1. Hence, we have

$$A^* \geq S_{upp} \geq 1/D \tag{15}$$

Apparently, *1/D* is the best value for Supp if 100% of nodes are to be updated. In other words, increasing S_{upp} may cause dissatisfying P1. However, this minimum may possibly cause some of the hops with smaller areas to be empty which is against P2. As a solution, we approximate each route's area as a triangle as can be seen in Figure 7. Note that corner triangles should also be con-

tinued to the sink, so we assume that each of these triangles has a height equal to S_{upp}/t, and a base equal to Ht. Thus we have

$$A_R \cong HtS_{upp}/2t = HS_{upp}/2 \quad (16)$$

To satisfy P1 and P2, it is required to set $N_R = H$. Therefore we have $N_R = A_R D = HS_{upp}D/2$, which implies

$$S_{upp} = 2/D \quad (17)$$

It should be also noted that for larger values of S_{upp}, P1 might not be satisfied due to the fact that the number of nodes may be increased to more than one and hence, be omitted from the routes. However, for larger values of S_{upp}, it is more unlikely to have same node receives the packet twice (i.e. satisfying P2).

Another important point is that if we assume we have only local information, then density D may not known at each node. We can either assume that we have D (non local info), or estimate it using

$$D^u_{est} = N^u_{neib} / (\pi t^2/4) \quad (18)$$

where N^u_{neib} denotes the number of neighbors of node *u*.

6. Delay Analysis

6.1. Propagation Delay

The propagation delay, the approximate delay is proportional to $H = \sqrt{m'^2 + m^2}$. Therefore, transmission delay is proportional to distance between source and the sink, dt = c H, where c is the average propagation delay for one hop transmission.

6.2. Priority Queue delay

When there is buffer management in nodes, the packet sees an average delay, **ds**, at each hop. **ds** depends on

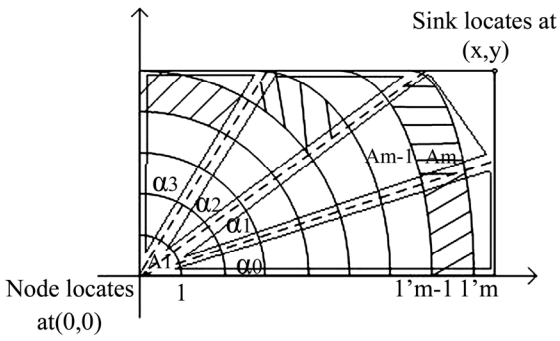

Figure 7. Approximating the routes with triangles.

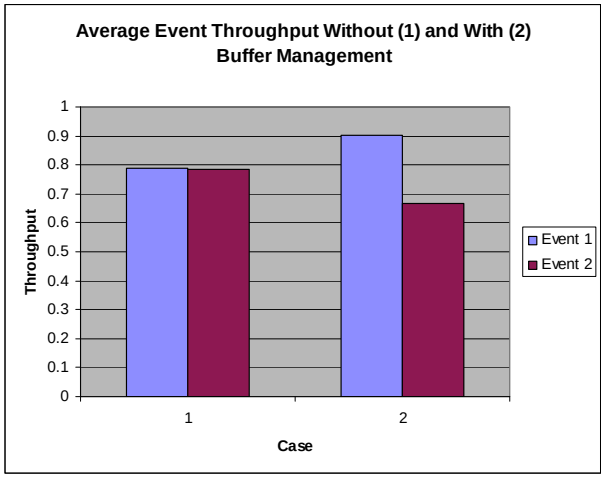

Figure 8. Normalized throughput of 2 events with simple queue management (case 1) and priority queue (case 2).

congestion level and also the priority of the packet. In [12], an analysis for the average delay in a non- preemptive Head of Line (HOL) priority queue is given. A non-preemptive HOL queue has the property that the packet in the head of line, which is being processed by the server, should be completely served before other packets can be processed. Using the results (Equation (9) in [12]) and simplifying it for a HOL M/M/1 priority queue, we get the following formula. [1]

$$ds_i = \mu_i + \frac{\sum_{m=1}^{i}\rho_m}{1-\sum_{m=1}^{M}\rho_m} + \frac{\sum_{m=1}^{i-1}\sum_{n=1}^{i-1}\rho_m\rho_n}{2(1-\sum_{m=1}^{i-1}\rho_m)(1-\sum_{m=1}^{i}\rho_m)} \quad (19)$$

where μ_i is the average service time for ***i***'th priority class; **M** is the total number of priority classes; and ρ_i is the offered load of packets in priority class ***i***. Denoting the packet arrival rate of priority class ***i*** by λ_i, ρ_i is defined as $\rho_i = \lambda_i\mu_i$. In general, a smaller index for a priority class implies a higher priority, and hence less delay for that class. In fact, from the Equation (10), it can be seen that the first class packets observe the least delay (i.e. $ds_1 = \mu_1$). More importantly, this delay is less than the delay in a queue without prioritizing ($ds' = \frac{1}{1-\rho}\mu$). The fact that $ds_1 < ds'$, shows that the more important packets leave the queue quickly. On the other extreme hand, the delay for last priority class ds_M is obviously larger than ds', and hence, the packets from less important class observe more delay compared to non- prioritized mode.

[1]The second and third terms in the right hand side of the Equation (1) should also have the same unit as μ_i by multiplying them into 1 time unit.

ds_i is the queuing delay for one hop only. Similar to what we had for the propagation delay, the total queuing delay will be $dq_i = H\, ds_i$.

Figure 8 shows the effect of implementing a simple priority queue scheme on the throughput from two different priority classes. The setting is similar to Figure 2 when both events are active. As can be seen, after applying the buffer management, the class 1 packet will have more throughput compared to normal queue management. For example, if $\alpha_i = 0.9$, then the first event can be detected with the priority queue management in contrast to normal queue management.

6.3. Total Delay

The total delay is given by $d_i = d_t + dq_i = H\,(c + ds_i)$. Therefore, for the packets in a certain priority class, the nearest sink to the node observes the least delay. Hence, after disseminating of the information about the existence of one event by the proposed angle routing, the nearest sink can be used in order to achieve the least possible delay. The data dissemination phase guarantees that the maximum number of events is detected, because of the global event ordering knowledge in the network.

7. Analysis of Our Protocol

At first glance, our DQRP protocol seems to reduce the throughput or the number of events detected because the average number of hops per event packet increases. We should prove using linear programming to show that DQRP can actually increase our performance metric which is the number of targets detected. Shortest-path greedy geographic routing is not optimal in multi-sink sensor networks.

We consider a $n \times n$ grid topology of sensor nodes with 4 sinks located at the corners of the grid. The sensor network can be modeled using the representation of a graph $G=(V,E)$ where V is the set of vertices and E is the set of edges. The sink and all sensor nodes are in the set of vertices. Each vertex is associated with a location information given by (x_i, y_i). Each sensor node has a maximum transmission range of t There is an edge (u, v) in E if the nodes are within transmission range of each other. Formally, this is stated as

Edge $u,v \in E$ *iff* $\sqrt{(x_u - x_v)^2 + (y_u - y_v)^2} \le t$

Data is sent from a sensor node through intermediate sensor nodes to the sink if the sink is not within direct transmission range of the sensor. We let $f(u,v)$ be the total amount of data transmitted from sensor node u to sensor node v. These data includes data from other sensor nodes and data originating from the node itself. We let G be a weighted graph with a weight function w. The weight $f(u,v)D$, $w(u,v)$ of the edge $u,v \in ED$ is the cost of transmission from node u to node v. The transmission cost is dependant on the distance between the nodes as well as the propagation model used. In general, $w(u,v) \infty d^k$ where d is the distance between 2 nodes and k is the path loss exponent. We let k be 2 in our scenario. As we consider only a static grid topology, $w(u,v)$ is a fixed value. We consider 3 types of routing:

1) Nearest-sink Greedy Geographic Routing: This means that the node will choose the nearest sink to send the data packets to and the next hop it chooses will maximize the distance gained towards the sink.

2) Nearest-sink Non-Greedy Geographic Routing: This means that the node will choose the nearest sink to send the data packets to and it can choose any neighbor which is nearer to the sink than it is to forward the data packets.

3) Multi-sink Non-Greedy Geographic Routing: This means that the node can choose any sink to send the data packets to and it can choose any neighbor which is nearer to the target sink than it is to forward the data packets. We minimize the maximum energy consumed by any sensor node by linear programming:

7.1. Nearest-Sink Greedy Geographic Routing

Minimize p subject to the following constraints:

$$f(u,v) = 0 \quad \text{for each} \quad (u,v) \notin E \tag{20}$$

$$f(u,v) \ge 0\text{D} \quad \text{for each} \quad (u,v) \in E \quad \text{and } v \text{ maximizes distance gained towards the nearest sink of } u \tag{21}$$

$$\sum_{v \in V} f(u,v) \le c \quad \text{for each} \quad u \in V \text{ - } \{\text{sink}\} \tag{22}$$

$$\sum_{v \in V} f(u,v) - \sum_{v \in V} f(v,u) = L_u \quad \text{for each} \quad u \in V \text{ - } \{\text{sink}\} \tag{23}$$

$$\sum_{v \in V} f(u,v) = 0 \quad \text{for u} \in \{\text{sink}\} \tag{24}$$

$$\sum_{u \in \text{sink}} \sum_{v \in V} f(v,u) = \sum_{u \in V - \{\text{sink}\}} L_u \ \text{u} \in \{\text{sink}\} \tag{25}$$

$$\sum_{v \in V} f(u,v) w(u,v) \le p \quad u \in V \text{ - } \{\text{sink}\} \tag{26}$$

The variable c is the maximum amount of data transmitted by any sensor node among all the sensor nodes in the network. Constraint (20) means that if two sensor nodes are not within the transmission range, the flow between each other is 0. If two sensor nodes are within transmission range, the flow between each other must be non negative as stated in Constraint (21). Constraint (21) will vary based on the type of routing algorithm used. Constraint (22) states that the total amount of transmission data by any sensor node cannot exceed c. We set c to be 10 packets of data. Constraint (23) states that all received data are to be forwarded and every sensor node has L_u units of data to be sent to the sink. This value

depends on the number of event packets assigned to that node. Constraints (24) and (25) states that the sink is not sending any data and should receive all the data from the sensor nodes. Constraint (26) states that if the linear programming problem can be solved, the solver should produce a solution that minimizes the energy consumption of the sensor nodes.

7.2. Nearest-Sink Non-Greedy Geographic Routing

The linear program is similar to the previous case but (28) is modified.

Minimize p subject to the following constraints:

$$f(u,v) = 0 \quad \text{for each} \quad (u,v) \notin E \tag{27}$$

$f(u,v) \geq 0$ for each $(u,v) \in E$ and v is nearer to the nearest sink than u is (28)

$$\sum_{v \in V} f(u,v) \leq c \quad \text{for each} \quad u \in V - \{\text{sink}\} \tag{29}$$

$$\sum_{v \in V} f(u,v) - \sum_{v \in V} f(v,u) = L_u \quad \text{for each} \quad u \in V - \{\text{sink}\} \tag{30}$$

$$\sum_{v \in V} f(u,v) = 0 \quad \text{for} \quad u \in \{\text{sink}\} \tag{31}$$

$$\sum_{u \in \text{sink}} \sum_{v \in V} f(v,u) = \sum_{u \in V - \{\text{sink}\}} L_u \quad \text{for} \quad u \in \{\text{sink}\} \tag{32}$$

$$\sum_{v \in V} f(u,v) w(u,v) \leq pd \text{ for each } u \in V - \{\text{sink}\} \tag{33}$$

7.3. Multi-sink Non-Greedy Geographic Routing

The linear program is similar to the previous case but (35) is modified.

Minimize p subject to the following constraints:

$$f(u,v) = 0 \quad \text{for each} \quad (u,v) \notin E \tag{34}$$

$$f(u,v) \geq 0 \quad \text{for each} \quad (u,v) \in E \tag{35}$$

$$\sum_{v \in V} f(u,v) \leq c \quad \text{for each} \quad u \in V - \{\text{sink}\} \tag{36}$$

$$\sum_{v \in V} f(u,v) - \sum_{v \in V} f(v,u) = L_u \quad u \in V - \{\text{sink}\} \tag{37}$$

$$\sum_{v \in V} f(u,v) = 0 \quad \text{for u} \in \{\text{sink}\} \tag{38}$$

$$\sum_{u \in \text{sink}} \sum_{v \in V} f(v,u) = \sum_{u \in V - \{\text{sink}\}} L_u \text{ , u } \in \{\text{sink}\} \tag{39}$$

$$\sum_{v \in V} f(u,v) w(u,v) \leq p \text{ for each } u \in V - \{\text{sink}\} \tag{40}$$

We vary the number of packets generated by each event and determine the maximum number of events which can satisfy the QoS. Each event is randomly assigned to sensor nodes. The transmission range is set to $\sqrt{2}$, therefore each node has 8 neighbors. The maximum number of events is reached when the solver cannot find a solution to the linear program. We use the solver in MATLAB for this analysis. Figure 9 shows the maximum number of events satisfying the QoS requirements for a 5 by 5 grid topology and Figure 10 shows the results for a 6 by 6 grid topology. These results show that multipath non-greedy geographic routing can indeed improve network performance.

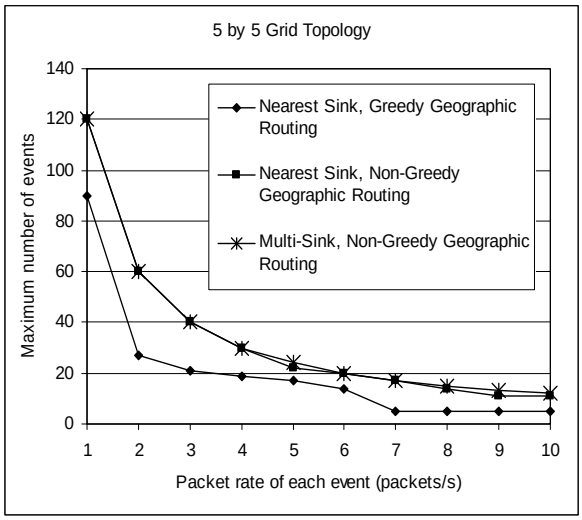

Figure 9. Maximum number of events satisfying QoS requirements in a 5 by 5 grid topology.

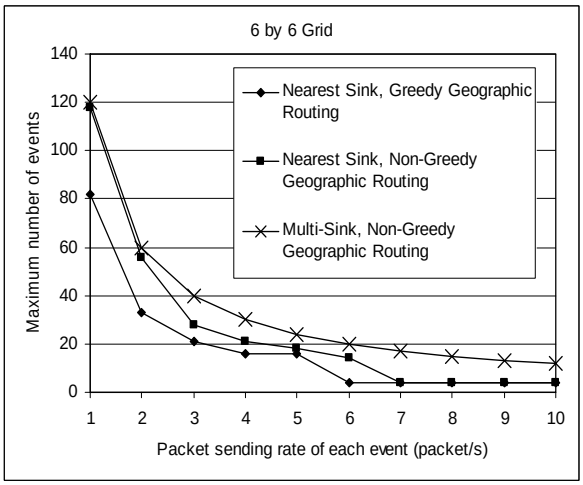

Figure 10. Maximum number of events satisfying QoS requirements in a 6 by 6 grid topology.

8. Implementation Results

We test the effectiveness of our priority queue buffer by implementing the scheme on the MICAz motes. In the first scenario, we use 3 motes in which there is 1 sink and 2 traffic sources as shown in Figure 11. In the second

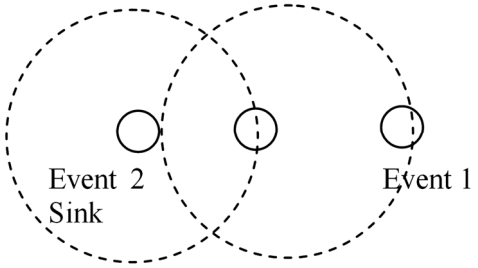

Figure 11. Network topology in scenario 1.

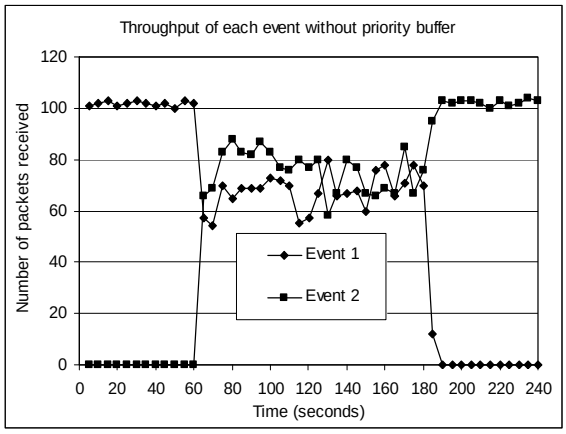

Figure 12. Throughput of events without priority buffer.

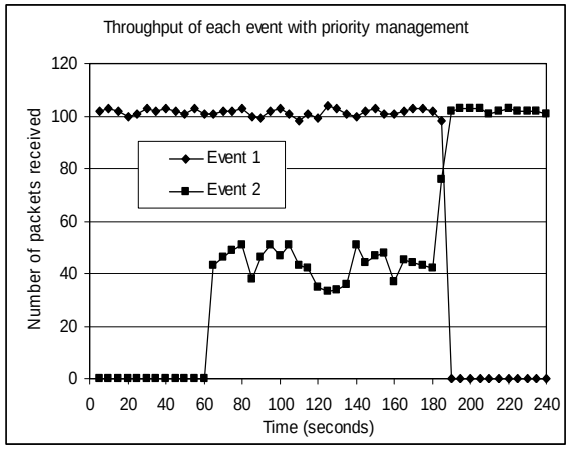

Figure 13. Throughput of events with priority buffer.

scenario, we use a total of 18 motes in which there is 1 sink and 5 traffic sources. The nodes are randomly deployed in a lab. Each mote is between 1 to 5 hops away from the sink. Each event sends packets at a rate of 20 packets per second for the first scenario and 10 packets per second. The sampling interval is set to 5 seconds.

Figure 12 shows the number of packets received by each event when there is no priority buffer management scheme and Figure 13 shows the number of packets received by each event when there is priority buffer management scheme. Figure 14 shows the number of events detected based on our QoS requirements. It can be clearly seen that the priority management scheme improves QoS performance. In the random deployment scenario, each event is gradually introduced into the network until there are 5 events.

After that, events gradually leave the network. Figure 15 shows the number of events detected. The results show that our priority buffer management scheme performs better most of the time.

One of the main problems that we face when carrying out the experiments is that the experiment was done in a lab where shadowing and fading effects are more serious compared to an open field. This causes the link quality to vary rapidly. As a result, there are a lot of fluctuations in the number of packets received and routes between nodes change quite often. Our scheme could potentially improve the performance much higher if the experiment is carried out in an open environment (e.g. open field) where there are less obstacles and the link quality is more stable.

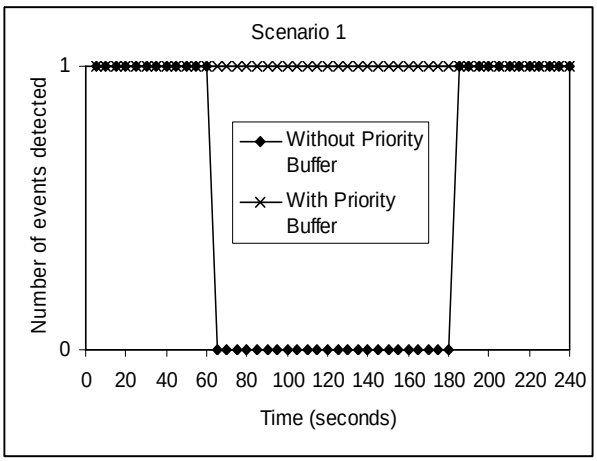

Figure 14. Number of events satisfying our QoS requirements. The required data delivery ratio is 90%.

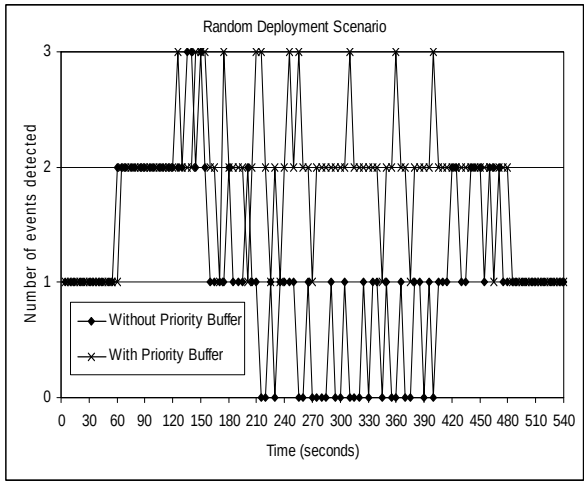

Figure 15. Number of detected events in a random deployment scenario satisfying our QoS requirements. The required data delivery ratio is 70%.

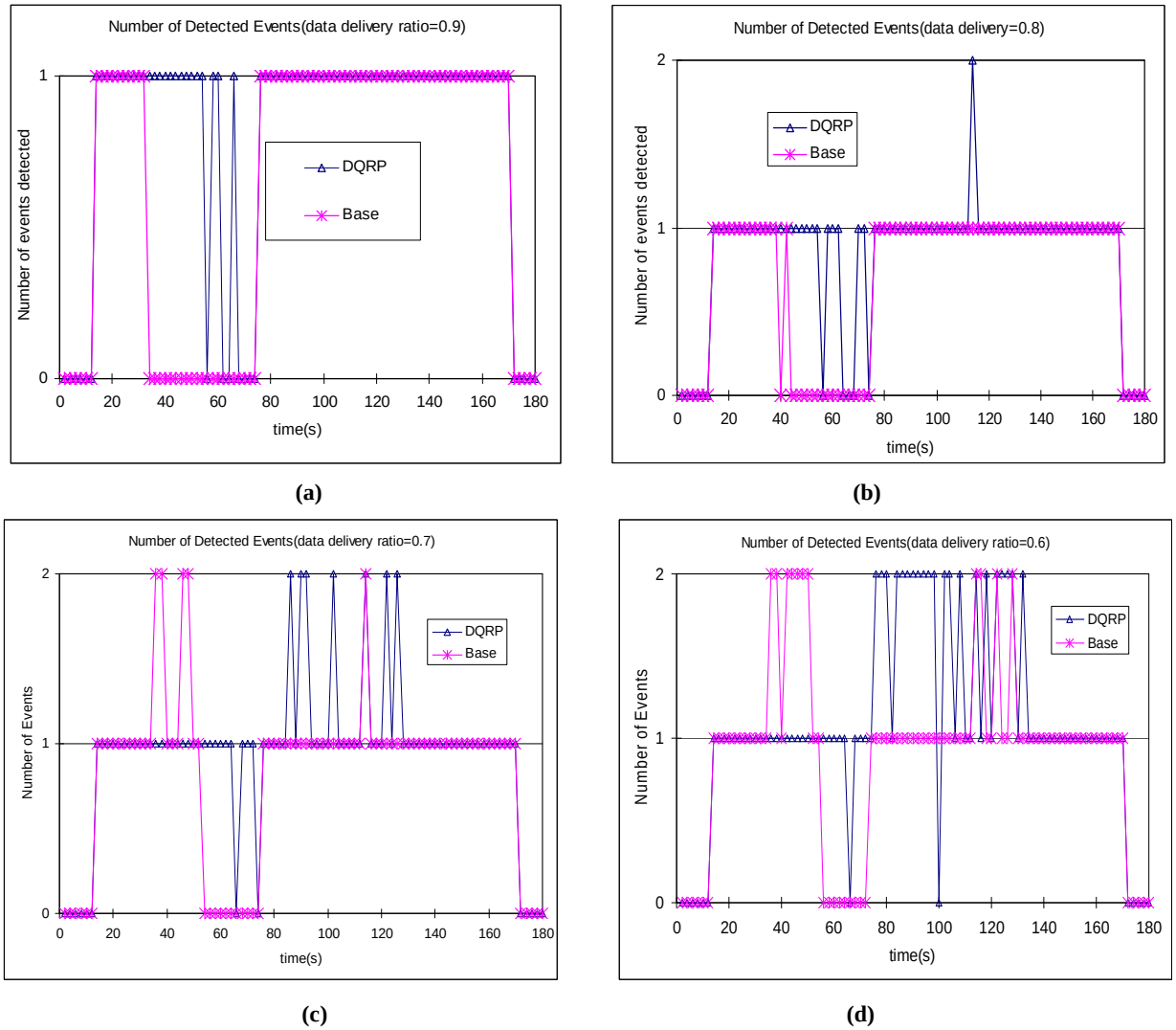

Figure 16. Number of detected events, out of max. 4 events, in the basic setup (stars) and DQRP (triangles). Vertical axis shows the number of events detected, and horizontal axis shows the time. The required data delivery ratio for an event detection at sinks is a) 0.9, b) 0.8, c) 0.7, and d) 0.6.

9. Simulation Results

We simulate a network consisting of 25 nodes with DQRP and using priority buffer management scheme based on event ordering. The simulation tool used is TOSSIM. There are 4 sinks located at each corner of the simulation area. We simulate two scenarios, one with 4 events and another with 8 events. Events are gradually inserted into the network. Events can also leave the network after some time. The sampling interval is 2 seconds. We compare the number of events detected with varying QoS requirements from 60% delivery ratio to 90% delivery ratio. The results for the scenario with 4 events are shown in Figure 16 and the results for the scenario with 8 events are shown in Figure 17. The results show that DQRP with priority buffer management scheme based on event ordering achieves better performance.

10. Conclusions and Future Work

In this paper, we have presented a QoS network architecture for target tracking WSNs consisting of a priority buffer management scheme based on event ordering and a routing algorithm to disseminate the event ordering.

We believe that better performance can be obtained from our system by using cross-layer design. For example, once we have the global event-ordering list, at the MAC layer, we can assign more time slots using a TDMA MAC protocol to events that have a higher priority. This can provide more assured QoS than the CSMA MAC protocol that is used in our simulations and experiments.

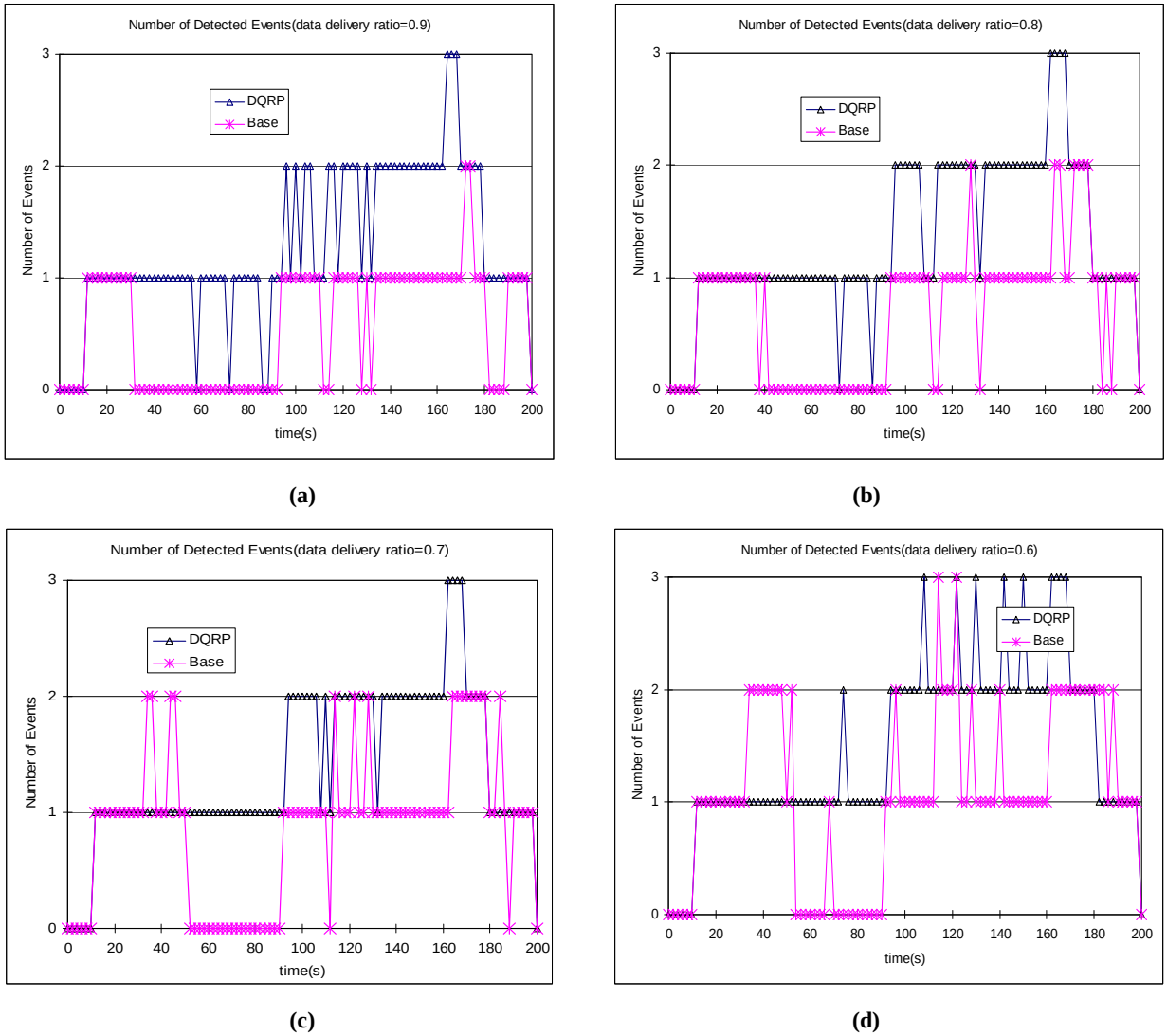

Figure 17. Number of detected events, out of max. 8 events, in the same format as Figure 16.

11. References

[1] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of sensor network communication models," ACM Mobile Computing & Communication Review, Vol. 6, No. 2, pp. 1−8, 2002.

[2] D. Chen and P. K. Varshney, "QoS support in wireless sensor networks: A survey," Proceedings of the 2004 International Conference on Wireless Networks (ICWN 2004), Las Vegas, Nevada, USA, June 2004.

[3] R. Iyer and L. Kleinrock, "QoS control for sensor networks," Proceedings of IEEE International Conference on Communications (ICC'03), Vol. 1, pp. 517−521, May 2003.

[4] C. Gui and P. Mohapatra, "Power conservation and quality of surveillance in target tracking sensor networks," ACM International Conference on Mobile Computing and Networking (MOBICOM), 2004.

[5] Q. X. Wang, W. P. Chen, R. Zheng, K. Lee, and L. Sha, "Acoustic target tracking using tiny wireless sensor devices," In International Workshop on Information Processing in Sensor Networks (IPSN), 2003.

[6] G. H. He and J. C. Hou, "Tracking targets with quality in wireless sensor networks," proceedings of the 13th IEEE International Conference on Network Protocols, 0-7695-2437−8/05, 2005.

[7] K. I. Park, "QoS in packet networks," Springer, pp. 159−179, 2005.

[8] A. Durresi, V. K. Paruchuri, S. S. Iyengar, and R. Kannan, "Broadcast protocol for sensor networks," Technical Report, LSU-CSC, October 2003.

[9] A. Durresi, V. K. Paruchuri, L. Barolli, and R. Jain, "Qos-energy aware broadcast for sensor networks", Proceeding of the 8th IEEE Int. Symp. on Parallel Architectures, Algorithm and Networks (ISPAN'05), 1087−4089/05, 2005.

[10] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in Ad-Hoc wireless networks," ACM Wireless Networks, November 2001.

[11] B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," ACM/IEEE Int. Conf. on Mobile Computing and Networking (Mobicom), 2000.

[12] J. Walraevens, B. Steyaert, M. Moeneclaey, and H. Bruneel, "Delay analysis of a HOL priority queue," Telecommunication Systems, Vol. 30, No. 1–3, pp. 81–98, 2005.

[13] G. N. Shirazi, P. Wang, D. Xiangxu, Z. A. Eu, and C. K. Tham, "A QoS network architecture for multi-hop multi-sink target Ttracking WSNs," IEEE Inte Conference on Communication Systems (ICCS), pp. 17−21, 2008.

