Chapter 3

Robust and Real Time Data Delivery Mechanism (RRTD)
3.1 INTRODUCTION

Over the last years, sensor networks are being widely used in many smart applications, including military applications and earthquake response systems. While these applications remain diverse, one common point they all share is the need of an efficient and reliable real-time communication mechanism. However, the potential contention in MAC protocols (e.g., IEEE 802.11 and 802.15.4), the node mobility nature of the sensor networks, and the interference between the transmitting nodes, all make it difficult to achieve good quality real-time communication (data delivery). Providing real-time data delivery in wireless sensor networks is a challenging research problem.

Real time communication is divided into two categories: Soft real time communication and hard real time communication. Soft real time communication provides QoS guarantee to the applications and is responsible for meeting the message delivery deadlines. Hard real communication deals with proper network utilization. A lot of work has been done in both the directions but none of them provides both hard as well as soft real time communication at the same time.

Ours is the first work that provides both hard as well as soft real time guarantee at the same time. So, in this chapter, we are proposing a Robust and Real Time Data Delivery in Wireless Sensor Networks (RRTD) mechanism which uses centralized control plane incorporating the timed token protocol in the MAC layer for wireless token ring architectures for providing hard real time guarantee, and in advance bandwidth reservation method to provide soft real time guarantee. This mechanism is proposed to support real time communication on real time routing layer of the proposed architecture. The reason to adapt the timed token protocol to wireless networks is that it has special timing properties and solid mathematical foundations [12] [13] [14] [15]. While a task is executing, RRTD reserves enough bandwidth between the source and destination nodes.
We consider a sensor network with a set of mobile nodes $N = \{n_1, n_2, n_3, \ldots\}$. Nodes may join, leave, move or fail at any time, resulting in unexpected packet losses. Real-time data delivery is often triggered by the completion of a real-time task. A real-time task executes on an individual node, with a given termination time $T_{\text{trm}}$. A task starts at time $T_{\text{st}}$. After a given execution time $T_{\text{et}}$, it finishes at time $T_{\text{fn}}$ and has some data which is required to be delivered to another node before $T_{\text{trm}}$. The time for real time data delivery is the task’s slack time $T_{\text{sll}}$, $T_{\text{sll}} = T_{\text{trm}} - T_{\text{fn}}$. Before task execution starts, $T_{\text{et}}$ and the size of delivered data can be estimated through application profiling techniques [16] [17]. After the task completes, RRTD executes real-time data delivery within the required time period $T_{\text{sll}}$. In addition, to deal with network failures, RRTD simultaneously transmits data in multiple paths.

### 3.2 SYSTEM ARCHITECTURE

With the timed token protocol, a synchronous bandwidth allocation (SBA) scheme must also be used to allocate synchronous bandwidth to the stations properly for guaranteeing the deadlines of real-time messages. Various SBA schemes have been proposed in the literature and also the non-optimality of the most famous schemes has been shown [18] [19] [20]. In the proposed control plane, Enhanced Minimum Capacity Allocation (EMCA) [21] is used as the SBA scheme due to its good performance and simplicity.

To achieve hard real-time communication, three important functions are implemented in the control plane, namely the Request Status procedure, the Priority management procedure and a traffic differentiation mechanism. The Request Status procedure determines whether a connection request should be accepted and as a result the requesting station could be admitted into the ring network. This decision is actually based on the current network state, such as the current load, the number of connections established and the class of traffic carried over these connections. If the expected QoS of the connection request can be satisfied, the requesting station is accepted to join to the ring. If the connection cannot be accepted, the management station executes the priority management procedure. This procedure tries to remove the best possible station in the token ring with a lower traffic class than the requested
connection to make it possible that the new connection can be accepted. The last contribution of the proposed control plane is the traffic differentiation mechanism. The traffic classes as proposed by 802.11e are supported in this control plane. And as a result of the traffic differentiation mechanism and the priority management procedure, the system tries to admit as much high priority traffic into the network as possible and becomes more responsive to the high priority traffic.

To achieve soft real-time communication, RRTD executes bandwidth reservation (or BR) before the real data delivery (or RD) begins. In this way, when a real-time task completes and data is ready for delivery, it can immediately transmit data with desired sending rate. RRTD uses an existing wireless routing protocol (DSDV) [22] to provide immediate data delivery path. RRTD’s basic strategy is shown in figure 3.1

![Figure 3.1: RRTD Strategy](image)

RRTD also needs to reserve enough bandwidth before data is delivered; it achieves bandwidth reservation by controlling its own and its neighbors’ packet sending and receiving rates. In addition, it delivers data through multiple paths, in order to robustly transmit data, or separately transmit large data chunks.

Reactive protocols are not considered by RRTD, because path finding time might be very long, especially when several intermediate nodes exist in the path, i.e., long path finding time might cause data delivery violating its time constraint. RRTD therefore adopts a proactive routing protocol called “Destination-sequenced Distance Vector”
Robust and Real Time Data Delivery Mechanism (RRTD)

(or DSDV) [22]. DSDV achieves proactiveness by letting nodes periodically discover and maintain paths. With DSDV, when a packet needs to be delivered, the path is already known in the forwarding table of DSDV and can be immediately used. Each node maintains a forwarding table for all the reachable destinations. The table contains the next hop, number of hops, and sequence number for each destination. A sequence number shows the freshness of a path, and is used to help nodes distinguish stale paths from the new ones, and thus avoids formation of path loops. Nodes broadcast routing updates periodically or at the time the network topology changes. Each path is tagged with a Time-to-Live (TTL) field to indicate its freshness. Before a real-time data delivery starts, RRTD will reduce the TTL of the path between the source and destination nodes, in order to enhance path maintenance frequency. In this way, it remains a reliable path when delivery begins.

3.3 THE NETWORK AND THE MESSAGE MODEL FOR THE CONTROL PLANE

In the proposed architecture the timed token protocol [23] is adapted into the MAC layer and EMCA [24] SBA scheme is used for synchronous bandwidth allocation. The dynamic ring network is assumed to consist of \( n \) nodes at an instant. Message transmission is controlled by the timed token protocol. Token walk time \( \tau \) includes the ring latency, the token transmission time and other network dependent overheads, and thus represents the portion of TTRT (Target Token Rotation Time) that is not available for message transmission.

3.3.1 Target Token Rotation Time

The messages generated in the network are classified as synchronous and asynchronous messages. These are \( n \) streams of synchronous messages at a certain moment

\[
S_m = \{ S_{m_1}, S_{m_2}, \ldots, S_{m_n} \}
\]

Where stream \( S_{m_i} \) originates at node \( i \). Also each synchronous message stream \( S_{m_i} \) can be characterized as

\[
S_{m_i} = \{ T_i, I_i, D_i, P_i \}
\]
Robust and Real Time Data Delivery Mechanism (RRTD)

Where

- \( T_i \) is the maximum amount of the time required to transmit a message in the stream.
- \( I_i \) is the inter arrival period between messages in the stream.
- \( D_i \) is the relative deadline of messages in the stream, that is the maximum amount of time that can elapse between a message arrival and completion of its transmission.
- \( P_i \) is the priority of the stream.

Each synchronous message stream places a certain load on each system. We define the effective utilization, \( EU_i \), of the stream \( S_m \), as follows:

\[
EU_i = \frac{T_i}{\min(I_i, D_i)}
\]

The total utilization of the synchronous message set \( S_m \) is the fraction of time used to transmit the synchronous messages and is denoted as \( TU(S_m) \)

\[
TU(S_m) = \sum_{i=1}^{s} EU_i
\]

Each station can transmit its synchronous messages as much as the synchronous bandwidth allocated to it namely \( T_M_i \). To ensure stable operation of the timed-token protocol, the total bandwidth allocated to synchronous messages must be less than the available network bandwidth. This protocol constraint is

\[
T_M_i \leq TTRT - \tau
\]

### 3.3.2 Packet Priority Management

RRTD orders packet delivery sequence according to their priorities. The packet with the highest priority will be delivered first. Priorities in RRTD are illustrated in Table 3.1
TABLE 3.1: Priority in RRTD

<table>
<thead>
<tr>
<th>Priority level</th>
<th>Priority</th>
<th>Packet Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>10</td>
<td>Real-time</td>
</tr>
<tr>
<td>P2</td>
<td>20</td>
<td>RRTD Control (Reservation)</td>
</tr>
<tr>
<td>P2</td>
<td>21</td>
<td>RRTD Control (Multi-path)</td>
</tr>
<tr>
<td>P3</td>
<td>22</td>
<td>DSDV Routing</td>
</tr>
</tbody>
</table>

Here, RRTD control message (reservation) is sent by sender nodes (the source or intermediate nodes) to tell receiver nodes (the destination or intermediate nodes) the bandwidth to be reserved. RRTD control message (multi-path) is used by the source node to find multiple paths to the destination node. The number of paths is application-specific. Note that this number is an upper bound, because there might be less than the required number of paths between the source and destination nodes.

3.3.3 Delivery Bandwidth Computation

In order to know how much bandwidth is available for a node to use, we must take into consideration all the transmissions that directly affect its opportunities to transmit. To avoid the “hidden terminal” problem, before data transmission, the source node sends “Request to Send” (or RTS), and the destination node replies “Clear to Send” (or CTS). Every other node receiving RTS/CTS should remain in silence during the transmission period. With RTS/CTS, a node is not allowed to transmit whenever [17]:

1) It is receiving data;
2) One of its neighbors is receiving data (due to the reception of CTS);
3) One of its neighbors is transmitting data to the node that is neither the neighbor nor the node itself (due to the reception of a RTS).

The available bandwidth for a node x to transmit $B_{avl,x}$ is calculated as follows:

$$B_{avl,x} = B_{eff,x} - \left( B_{rec,x} + \sum_{y \in N_x} B_{rec,y} + \sum_{y \in N_x, k \in N_x^{+}} B_{yk} \right)$$

3.4
where $B_{rec}^x$ and $B_{rec}^y$ are the receiving bandwidths used by node $x$ and node $y$, $B_{yk}$ is the traffic from node $y$ to $k$, and $N_x \cup N_x^+$ is the set of neighbors of node $x$ excluding/including itself. In real systems, poor link quality and the interference between nodes makes only a portion of the total bandwidth is usable. Therefore, here we use the total effective bandwidth $B_{eft}$ for available bandwidth computation.

We consider data delivery in a 4-hop path where node A may be the sender node (S), node B, C, D may be the intermediate nodes and node E may be destination (R) node. Table 3.2 shows bandwidth consumption of nodes in data delivery path.

**TABLE 3.2: Delivery Table**

<table>
<thead>
<tr>
<th>Hop-Count</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hop 1</td>
<td>S</td>
<td>R</td>
<td>CTS</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Hop 2</td>
<td>RTS</td>
<td>S</td>
<td>R</td>
<td>CTS</td>
<td>-</td>
</tr>
<tr>
<td>Hop 3</td>
<td>-</td>
<td>RTS</td>
<td>S</td>
<td>R</td>
<td>CTS</td>
</tr>
<tr>
<td>Hop 4</td>
<td>-</td>
<td>-</td>
<td>RTS</td>
<td>S</td>
<td>R</td>
</tr>
</tbody>
</table>

Algorithm 3.1 is used to compute the required bandwidth. The required bandwidth for real-time data delivery is denoted by $B_{req}$. In delivery processes, nodes not only need to reserve $B_{req}$ bandwidth, but also need to consider the extra bandwidth which they use to remain in silence due to the reception of RTS/CTS.

```plaintext
if x = source / destination then
  if destination / source in neighbors then
    $B_{req}^x \leq B_{avl}^x$;
  else
    $B_{req}^x \leq B_{avl}^x / 2$;
  else if $x \in N_{source} \cup N_{destination}$ then
    $B_{req}^x \leq B_{avl}^x / 3$;
  else
    $B_{req}^x \leq B_{avl}^x / 4$;

Algorithm 3.1 Required Bandwidth
```
### 3.3.4 The Proposed Control Plane

The control plane is implemented in a distinct management station in the network and also resides in the logical link control sub layer. This assumption seems reasonable when we look at the system from the upper layers. The requirements for upper layers real-time communication are provided by the algorithms running on the logical link control sub layer and the management station. Source station $a$ that is not currently in the ring tries to establish a connection to destination station $d$ and transmit a synchronous stream $S_{ma}$ where $S_{ma} \notin Sm$. $TM$ is the set of synchronous bandwidths of stations that is calculated by the EMCA algorithm. Algorithm 3.2 describes dynamic ring management. Whenever a station wants to establish a connection and needs to join the ring, it sends a connection request to the station that runs the request status procedure to determine whether the connection should be accepted and whether a station should be removed from the ring. In our centralized approach, the management station has the knowledge of the global state of the system which includes all connections currently active in the system with their corresponding parameters and the class of traffic carried over these connections. First of all, the management station checks whether the new connection request can be accepted without risking other stations’ real-time guarantees. At this stage the EMCA SBA scheme is used for determining the synchronous bandwidths for the stations in the network. If it is possible to allocate the necessary bandwidth to the requesting station, the new $T_{mi}$ and TTRT values are broadcasted to the ring members and the station that sent the connection request joins the ring. TTRT should be updated according to the deadline parameter in the connection request. According to the timing properties of the timed token protocol, the maximum amount of time that can pass between two visits of the token are $2\cdot TTRT$. And hence if $D < 2\cdot TTRT$ the deadline can be missed. So $D$ must be greater than or equal to $2\cdot TTRT$ to meet the deadline. As a result, management station should update TTRT if the deadline in the connection request is smaller than $2\cdot TTRT$. After the station joins the ring, the predecessor of this station updates its connectivity tables so that it can pass the token to that station in the next cycle.
If the management station cannot accept the connection request due to the fact that accepting it will disrupt other stations’ real-time guarantees, it executes the priority management procedure. This procedure is used to remove the best possible station from the ring such that the new connection could be accepted. Here, the traffic class parameter in the connection request comes into play. The management station checks the stations other than the destination of the connection; since, to set up the connection, the destination station must stay in the ring. And then the management station tries to find the station with the least possible traffic class such that removing this station from the ring will make the system handle the new connection without disrupting the real-time guarantees of other stations. The connection of the evicted
station is disconnected. When a disconnection request is received from any other station, the management station could give higher priority to evicted stations when setting up a new connection for the sake of fairness. After the eviction of a station the management station calculates the new $T_{Mi}$ and TTRT values and broadcasts them together with the MAC address of the evicted station to all the ring members so that they can update their connectivity tables and parameters appropriately. If there is not a proper station to be removed from the ring to guarantee the QoS requirements of the requesting station, the connection request is rejected.

3.4 RRTD BANDWIDTH RESERVATION

In Equation 3.4, we observe that a node’s available bandwidth is affected by its receiving bandwidth ($B_{rec}^x$), neighbor nodes’ receiving bandwidth ($B_{rec}^y$) and sending bandwidth ($B_{yk}^{bjk}$). In order to reserve enough bandwidth, a node (the source, destination or intermediate) should collaborate with its neighbors. Except current available bandwidth $B_{avl}^x$, the extra required bandwidth for real-time data delivery, $B_{ex}^x$, is given by:

$$B_{ex}^x = B_{rq} - B_{avl}^x$$  \hspace{1cm} 3.5$$

If $B_{ex}^x \leq 0$, then there is no need to reserve extra bandwidth. But the node should exchange messages with its neighbors, telling them to control their bandwidth usage. For instance, node y can only use $B_{ex}^{xy}$ more bandwidth for its own transmission ($B_{ex}^{xy} \leq B_{ex}^x$). If $B_{ex}^x > 0$, then RRTD mechanism needs to reserve $B_{ex}^x$ more bandwidth for transmitting the data. For both neighbors and the node itself, if a data receiving process does not begin, they can postpone this process by delaying to send CTS messages. In this way, they can reduce $B_{rec}^x$ or $B_{rec}^y$ in order to obtain $B_{ex}^x$ more bandwidth. This is as well done through collaboration between the node and its neighbors.

There are two ways to reduce the sending rate. One sending rate (SR) is sharply reduced from SR0 to the required value RS (required sending rate) at $T_r$, which is the starting time of the real-time data delivery. However, there is a time delay for
available bandwidth $B_{avl}$ to increase from current $B_{avl}/0$ to the required bandwidth $B_{rq}$. Because real-time delivery begins at $T_r$, the delay time causes the delivery to violate its time constraint. RRTD adopts the second sending rate control where the sending rate reduction begins earlier than $T_r$. At time $T1<T_r$, it reaches the required $RS$, and continues to decrease till delivery begins. After that, the sending rate increases to $RS$. Although the available bandwidth increases with a time delay, at $T_r$, the total available bandwidth exceeds the required value $B_{rq}$ and comes back to $B_{rq}$ at time $T2$. In this way, RRTD satisfies each real-time data delivery’s time constraint.

### 3.4.1 Packet Blocking Control

RRTD provides a packet blocking control mechanism to deal with conditions where SR0 is very large and the sending rate does not decrease very fast. When starting to receive real-time data, a receiver node (an intermediate node or the destination node) computes the error ESR between the real-time data receiving rate $RD$ and the required bandwidth reservation $B_{rq}$, by extracting necessary information from the MAC layer [25]:

$$E_{SR} = \frac{RD - B_{rq}}{B_{rq}}$$  \hspace{1cm} (3.6)

If $E_{SR} < 0$ then the receiver randomly drops $E_{SR}$ percentage.
3.4.2 Multi-path Data Delivery

Message losses and node failures are frequent in some sensor networks. To achieve reliable real-time data delivery, RRTD adopts multi-path delivery mechanism as shown in figure 3.3(a) (the number of paths is application-specific). Figure 3.3(b) shows if the required bandwidth exceeds a node’s total effective bandwidth $B_{\text{eff}}$, multi-path delivery can help to distribute the delivery work load to different paths.

![Multi-path for Reliable Delivery](image)

Figure 3.3(a): Multi-path for Reliable Delivery

![Multi-path for Large Data Delivery](image)

Figure 3.3(b): Multi-path for Large Data Delivery
3.5 SIMULATION STUDIES

We conducted a set of simulations of the RRTD mechanism using J-sim [70] and its performance was compared with several other mechanisms, such as AODV [68], DSR [67], and GF [66] which are available in the literature. These mechanisms also provide real-time services in adhoc networks.

3.5.1 Simulation Environment

We consider a single broadcast region with an available link capacity of 2 Mb/sec under the IEEE 802.11 protocol with an effective data rate of approximately 1.43 Mb/sec. Each node generates variable-rate traffic (randomly uniform distribution) according to the exponential on-off traffic. Real-time data chunk sizes are randomly generated subject to uniform distribution with the minimum value of 50 Bytes.

Non-real-time data is transmitted with a random, uniformly distributed rate with the minimum value of 100 Bytes and the maximum of 400 Bytes. Within the transmission time, each delivery task’s sending rate remains constant. Each real-time task’s execution time ($T_e$) and slack time ($T_s$) are also in a randomly uniform distribution with the minimum value of 1000 ms (real-time tasks are started). Inter-packet time between two packets is exponentially distributed. We employ random topologies with the node mobility uniformly distributed ranging from 0.1m/s to 3 m/s. If there is no possible path between source and destination, the packet is deleted and counted as an unsuccessful delivery. Other parameters, e.g., 802.11 physical layer parameters, were set to default values as recommended in J-Sim [70]. Simulation parameters are summarized in Table 3.3.

| TABLE 3.3: Simulation Parameters of RRTD |
| Routing Protocols | DSR, AODV, GF, RRTD |
| MAC Layer | 802.11 |
| Radio Layer | RADIO-ACCNOISE |
| Propagation model | TWO-RAY |
| Bandwidth | 200Kb/s |
| Payload size | 32 Byte |
| TERRAIN | (200m, 200m) |
| Node number | 100 |
| Node placement | Uniform |
3.5.2 Simulation Results

3.5.2.1 End to End Miss Ratio

The miss ratio is the most important metric in soft real-time systems. We set the desired delivery speed SSP to 1km/s, which leads to an end-to-end deadline of 200 milliseconds. In the simulation, some packets are lost due to congestion or forced-drops. We also consider these situations as a deadline miss. The results shown in Figure 3.4 are the summary of 16 randomized runs. AODV and DSR didn’t perform well in the face of congestion because both algorithms flood the network in order to discover a new path when congestion leads to link failure. This flooding just serves to increase the congestion. GF only switches the route when there are link failures caused by heavy congestion. The routing decision is based solely on distance and does not consider delay. Only RRTD tries to maintain a desired delivery speed through MAC and network layer adaptations, and therefore has a much less miss ratio than other algorithms. Due to its transient behavior, RRTD still has about a 20% miss ratio when the network is heavily congested.

![Figure 3.4: Miss Ratio](image)

3.5.2.3 Congestion Avoidance

In a sensor network, where node density is high and bandwidth is scarce, traffic hot spots are easily created. In turn, such hot spots may interfere with real-time guarantees of critical traffic in the network. To achieve reliable real-time data
delivery, RRTD adopts multi-path delivery mechanism (the number of paths is application-specific). To test the congestion avoidance capabilities, we use a base station scenario, where 6 nodes, randomly chosen from the left side of the terrain, send periodic data to the base station at the middle of the right side of the terrain. The average hop count between the node and base station is about 8~9 hops. Each node generates 1 CBR flow with a rate of 1 packet/second. To create congestion, at time 80 seconds, we create a flow between two randomly chosen nodes in the middle of the terrain. This flow then disappears at time 150 seconds into the run. In order to evaluate the congestion avoidance capability under different congestion levels, we increase the rate of this flow step by step from 0 to 100 packets/second over several simulations. Figure 3.5 plots the end-to-end (E2E) delay for the four different routing algorithms. At each point, we calculate the average of the E2E delays of all the packets from the 96 flows (16 runs with 6 flows each). The 90% confidence interval is within 2~15% of the mean, which is not plotted for the sake of legibility. Under the no or light congested situations, Figure 3.5 shows that all geographic based routing algorithms have short average end-to-end delay as compared to AODV and DSR. There are several factors accounting for this outcome. First, the route acquisition phase in AODV and DSR leads to significant delays for the first few packets, while geographic based routing doesn’t suffer from this. We argue that without an initial delay cost, geographic based routing is more suitable for real-time applications like target tracking where the base station sends the actuation commands to the sensor group, which is dynamically changing as the target moves. In such a scenario, DSR and AODV need to perform route acquisition repeatedly in order to track the target. Second, the route discovered through flooding and path reversal has relatively more hops than greedy geographic forwarding. The reason for even higher delay in AODV than DSR is that DSR implementation intensively uses a route cache to reduce route discovery and maintenance cost. Under the heavy congested situations (Figure 3.5), each routing algorithm responds differently. RRTD performs best. For example, RRTD reduces the average end-to-end delay by 30%~40% in the face of heavy congestion in comparison to the other algorithms considered. The key reasons for RRTD’s better performance are that DSR, AODV and GF only respond to severe congestion, which leads to link failures (i.e., when multiple retransmissions fail at the MAC layer). DSR, AODV and GF routing decisions are not based on the link delays, and therefore may cause congestion at a particular receiver even though it has long
Delays. DSR and AODV flood the network to rediscover a new route when the network is already congested.

The mean value of the service time, expressed in number of slots and averaged over all locations in the nodes, is

$$E[M] = \frac{2}{R^2} \int \frac{r \, dr}{Q(r)}$$  \hspace{1cm} (3.7)

Since the probability of successful reception $Q(r)$ decreases with increasing propagation distance $r$, the number of (re-)transmission attempts, $M$, statistically increases with increasing $r$. $R$ is the cell radius. We approximate hexagonal cells of unity size by circular cells of radius $R = \frac{3}{4}\sqrt{(2\pi)} \approx 0.91$.

The second moment is:

$$E[M^2] = \frac{1}{\pi R^2} \int \frac{1 - Q(r)}{Q^2(r)} \frac{z}{r \, dr}$$  \hspace{1cm} (3.8)

The expected queuing delay is found from the Pollacek-Khintchine expression for $M/G/1/\infty$ queues are D.

$$D = \frac{LC}{\eta, B_N} \left[ E[M] + \frac{E[M^2]}{E[M]} \frac{P_{\text{in}}}{2(1 - P_{\text{on}})} \right]$$  \hspace{1cm} (3.9)

Figure 3.5: Delay
3.5.2.3 Energy Consumption

Under energy constraints, it is vital for sensor nodes to minimize energy consumption in radio communication to extend the lifetime of the sensor networks. From the results shown in figure 3.6, we argue that geographic based routing tends to reduce the number of hops in the route, thus reducing the energy consumed for transmission. AODV performs the worst as a consequence of sending out many control packets during congestion. DSR has larger average hop counts and more control packets the another geographic base routing algorithms. RRTD only takes delay into account, which leads to longer routes. Figure 3.6 show’s that RRTD has nearly the same power consumption as GF because under such situations, RRTD tends to choose the shortest route and does not sent out any on demand beacons.

The instantaneous power consumption is the product of the input voltage and current:

\[ P(t) = V_{in} \frac{v_f(t)}{R} \]  

3.10

The input voltage is therefore approximated by a constant \( V_{in} \). The input current, \( i_{in}(t) \), was determined by measuring the \( v_f(t) \) voltage across the test resistance, R. Total energy consumed over an interval \([t_0, t_1]\) is the integral of power consumption over time:

\[ E_{t_0 \cdot t_1} = \frac{V_{in}}{R} \int_{t_0}^{t_1} v_f(t) \, dt \]  

3.11
3.5.2.4 Control Packet Comparison

Except AODV, all the other routing algorithms studied use a relatively low number of control packets. Most control packets in DSR and AODV are used in route acquisition. Because AODV initiates route discovery (flooding) whenever a link breaks due to congestion, it requires a large number of control packets. For example, in Figure 3.7, AODV sends out 12000 packets under the most congested situation, while SPEED only uses 1200 packets. DSR uses a route cache extensively, so it can do route discovery and maintenance with a much lower cost than AODV.

![Figure 3.7: Control Packets](image)

3.5.2.5 Void Avoidance

This experiment tries to evaluate the end-to-end delivery ratio of all the routing algorithms under different node densities. To eliminate packet loss due to the congestion, we only use four flows with a rate of 0.5 packets/second; these flows go from the left side of the terrain to the base station at the right side of the terrain. To change the density of the network, instead of increasing the number of nodes in the terrain, we keep the number of nodes constant at 100, and increase the side length of the square terrain in steps of 50 meters. It is no surprise that DSR performs best in the delivery ratio since it is a flooding based route discovery algorithm. Theoretically DSR should have 100% delivery ratio (Figure 3.8) as long as the network is not partitioned. All other geographic based algorithms have 100% delivery ratio when the network has high density (>12 nodes / per radio range). However, when the network density is reduced below 9 nodes/ per radio circle, GF, RRTD degrade performance rapidly. Only RRTD can manage to deliver 95%
of its packets to the destination. However RRTD drops 5% of its packets, because those packets need backtracking in order reach the destination. If backtracking, those packets would have a negative delivery speed, which is not allowed by RRTD for the sake of maintaining the real-time properties.

![Figure 3.8: Void avoidance](image)

### 3.6 CONCLUSIONS

In this chapter, we proposed a Reliable and Real Time Data Delivery in Wireless Sensor Networks (RRTD) mechanism to provide reliable real time data delivery. This mechanism uses centralized control plane incorporating the timed token protocol in the MAC layer for wireless token ring architectures, for providing hard real time guarantee, and in advance bandwidth reservation method to provide soft real time guarantee. The primary goal of the control plane is to manage the dynamic wireless ring network and provide sufficient bandwidth to higher priority traffic in order to satisfy their hard-real time constraints. For soft real time communication, RRTD executes bandwidth reservation before the real data delivery begins. In this way, when the data is ready for delivery, it can be immediately transmitted with desired sending rate. The simulation results justify that this mechanism ensures higher priority traffic more bandwidth than lower priority traffic and guarantees that deadline constraints are satisfied. As a result of the simulations, it is also seen that the new protocol decreases the miss ratio, delay and congestion and also minimizes the energy consumption. All of these results show that by adapting the proposed mechanism in the real time routing layer of wireless sensor network robust, reliable and real time communication is achieved.