IEICE TRANS. COMMUN., VOL.E93–B, NO.12 DECEMBER 2010

LETTER *Special Section on Wireless Distributed Networks*

# A Buffer Management Technique for Guaranteed Desired Communication Reliability and Low-Power in Wireless Sensor Networks*

**Dae-Young KIM**†, ***Nonmember***, **Jinsung CHO**†a), ***Member***, ***and*** **Ben LEE**††, ***Nonmember***

**SUMMARY** Reliable data transmission is desirable in wireless sensor networks due to the high packet loss rate during multi-hop transmissions. To reliably transmit data for event-driven applications, packet loss recovery mechanism is needed. For loss recovery, sensor nodes need to keep packets in their buffers until transmissions successfully complete. However, since sensor nodes have limited memory, packets cannot be buffered for a long period of time. This letter proposes an efficient buffer management technique that caches data packets for appropriate amount of time to minimize the resource requirements and at the same time provide reliable data transmission among sensor nodes.

***key words:*** *buffer management, reliable transmission, loss recovery, wireless sensor networks*

## 1. Introduction

A *wireless sensor network* (WSN) is an infrastructureless, self-organizing network of sensor devices that can be deployed in a variety of scenarios, including industrial, civilian, and military applications [1], [2]. In general, WSN applications involve monitoring information obtained by sensor nodes and then transmitting it to a sink node that manages the network. Since accurate and timely information is crucial for WSN applications, its reliable transmission from sensor nodes to the sink node is essential.

In general, the sink node collects the same type of data and/or redundant data from numerous source nodes in a large area to increase the reliability of the information. If reliability is not satisfied, the sink node requires more data from the source nodes, which would then have to perform frequent data retransmissions to satisfy the desired reliability. However, critical event-driven applications, such as emergency detection, usually occur in remote areas and they do not generate numerous redundant data. Thus, the sink node would have to recover lost packets to provide reliable delivery of event-driven data.

There are two types of loss recovery mechanisms used in reliable end-to-end transmissions for event-driven applications: *end-to-end* loss recovery and *hop-by-hop* loss recovery [3], [4]. End-to-end loss recovery schemes request retransmissions of lost packets from source nodes, while hop-by-hop loss recovery schemes request retransmissions of lost packets from intermediate or relay nodes that cache copies of the data.

Event-driven data in WSNs requires different levels of *communication reliability* (*CR*) per application service. In order to satisfy the desired end-to-end *CR* of an application, applying one recovery technique or the other to all data traffic is not suitable. For this reason, the authors proposed *Active Caching* [5], which is a flexible loss recovery technique that determines the most appropriate relay nodes in a routing path to cache the data based on the desired *CR*.

However, multiple data flows cause excessive use of buffers and thus increase the memory requirements of sensor nodes. Since sensor nodes have limited resources, an efficient buffer management together with a loss recovery mechanism are required to provide reliable data transmissions. Xiao et al. [6] proposed a method based on maximum round trip time (RTT), where sensor nodes cache data packets in their buffers until no retransmission requests are received for some period of time. Another method proposed by Paek and Govidan [7] uses a feedback message with a cumulative ACK sequence, which is the last continuous sequence number for the received packets. A node that receives the feedback message can safely remove packets in its buffer based on the ACK sequence. The RTT-based method requires sensor nodes to maintain the cached packets for a long period of time, while the feedback-based method causes high energy consumption due to additional messages required for lost packets.

This letter proposes a buffer management technique that precisely calculates how long packets need to be buffered, referred to as *packet holding time*, in intermediate nodes before they can be safely removed. Therefore, data packets are buffered for an appropriate amount of time to retransmit lost packets without incurring additional energy consumption to send feedback messages.

Manuscript received April 22, 2010.
Manuscript revised July 27, 2010.

†The authors are with the Dept. of Computer Engineering, Kyung Hee University, Korea.
††The author is with the School of Electrical Engineering and Computer Science, Oregon State University, OR, USA.

*This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2009-0083992) and by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2009-(C1090-0902-0002)).

a) E-mail: chojs@khu.ac.kr

DOI: 10.1587/transcom.E93.B.3522

$RELIABLE-TRANSMIT(CR, i, P_d(i-1))$
1. $P_d(i) \leftarrow P_d(i-1) \cdot (1-p_i)$
2. if $P_d(i) \leq CR$
3. $\quad P_d(i) \leftarrow 1 \cdot (1-p_i)$
4. $\quad$ cache data packets to a node $n_i$
5. forward $P_d(i)$ to the next node

**Fig. 1** Active Caching algorithm at *i*th node, $n_i$.

## 2. Background

The proposed buffer management technique is based on *Active Caching* [5], which is a flexible method that can be applied to various environments. Thus, this section briefly discusses Active Caching. The proposed buffer management technique, however, is a general technique that can be applied to any existing loss recovery mechanisms [3], [4].

Active Caching supports various levels of *CR* for application services using dynamic programming to determine data caching positions. The problem and subproblems of Active Caching for dynamic programming are defined as follows:

*Problem*: $P_d(H) > CR$.

*Subproblem*: $P_d(h) > CR$,

where $P_d$ represents the *packet delivery rate*, $H$ represents the number of hops between a source node and the destination node, and $h = 1, 2, \cdots, H$. The problem states that the packet delivery rate for $H$ hops between a source node and the destination node, $P_d(H)$, should be greater than *CR* of data. In order to achieve this, the packet delivery rate from a source node to each hop $h$, $P_d(h)$, should be greater than *CR*. The subproblem can be solved by caching data packets if the packet delivery rate does not satisfy *CR*. By solving the subproblems, the entire problem can be solved.

Figure 1 shows the Active Caching algorithm at *i*th node $n_i$. The node $n_i$ receives $P_d(i-1)$ piggybacked on data packets from node $n_{i-1}$. The node $n_i$ computes $P_d(i)$ by multiplying $P_d(i-1)$ by a factor $(1-p_i)$, where $p_i$ represents the packet loss rate at the *i*th hop. Then, $P_d(i)$ is compared with *CR*. If the value does not guarantee the desired reliability, packets are buffered at node $n_i$ and $P_d(i)$ is replaced with the packet delivery rate of the *i*th hop. Finally, the newly computed $P_d(i)$ is delivered to the next node.

## 3. The Proposed Method

The proposed buffer management technique estimates packet holding times based on the number of data transmissions. The estimated transmission counts also include the number of retransmissions for lost packets, which can be calculated using the analysis presented in [5]. When a source node transmits $k$ data packets through $h$ hops, the transmission counts will be $kh$ assuming there are no transmission failures. When packet losses occur, the transmission count can be estimated by adding the retransmission count for the lost packets to $kh$. The lost packets during a multi-hop communication are continually retransmitted until they all arrive at the next caching node.

During data transmission, a packet delivery rate at node $s$ for $h$ hops, $P_d(s,h)$, can be represented as

$$P_d(s,h) = \prod_{i=s}^{s+h-1} (1-p_i), \tag{1}$$

where the packet loss rate $p_i$ consists of a uniform link error $p$ and a contention error $p_c$, i.e., $p_i = p + p_c$. $p_c$ occurs when nodes attempt to transmit data simultaneously. This can be calculated by using a probability $p_a$ that nodes are in active state for communication within a transmission range, which is less than 0.1 for many IEEE 802.15.4 applications [8]. Howitt and Gutierrez [8] analyzed $p_c$ using $p_a$. Since data flows aggregate on the sink node, it is necessary to consider that the $p_a$ value for a node increases when it is located closer to the sink node.

WSNs have a transmission radius of 10 m as personal operating space and are based on multi-hop transmissions. Thus $p_a$ changes as hop-count varies. Assuming $p_a$ has an exponential distribution with the mean $\lambda$ based on the node's hop-count $h$ from the sink node, its density function is represented as

$$f(h) = \lambda e^{-\lambda h}, \quad h \geq 0, \tag{2}$$

where $\lambda$ is the maximum value of $p_a$. In addition, its distribution function is represented as

$$F(h) = \int_0^h \lambda e^{-\lambda h} dh = 1 - e^{-\lambda h}. \tag{3}$$

Each node lies between $h-1$ hop and $h$ hop distance and $p_a$ has a different value according to its location. For example, if a node is placed at the distance between 3-hop and 4-hop from the sink node, $h$ is 4 and $p_a$ of the node can be determined by the $h$. Then, $p_a$ is represented as

$$p_a(h) = e^{-\lambda(h-1)} - e^{-\lambda h}. \tag{4}$$

After $p_a$ is determined, $p_c$ for a node can be calculated using the number of neighbor nodes $N_t$. The probability of activity for each node within a transmission rage is $p_a/(N_t+1)$. Thus, $p_c$ is represented as

$$p_c = 1 - \left(1 - \frac{p_a}{N_t + 1}\right)^{N_t}. \tag{5}$$

Then, $p_i$ is

$$p_i = p + p_c. \tag{6}$$

In WSNs, each node receives a control message from the sink node. From the hop-count of the message, a caching node $s$ can estimate $p_a$ and can calculates $p_i$ ($i = s, s+1, \cdots, s+h$). $P_d(s,h)$ can be applied to Active Caching to obtain the hop-count $h$ from node $s$ to the next caching node. Given a level of *CR* for an application data, node $s$ obtains $h$ using the $HopCount(CR)$ function given in Fig. 2.

Once $h$ is obtained, the number of successfully transmitted packets among $k$ packets, $\Gamma_s(k, s, h)$, can be obtained using

$$\Gamma_s(k, s, h) = \sum_{m=1}^{k} m \cdot P_s(k, m, s, h), \quad (7)$$

where $P_s(k, m, s, h)$ represents the probability of successful transmissions for $h$ hops from node $s$ given as

$$P_s(k, m, s, h) = \binom{k}{m} \cdot P_d(s, h)^m \cdot (1 - P_d(s, h))^{k-m}. \quad (8)$$

Then, the number of retransmitted packets $\gamma$ and the number of retransmission request messages $\nu$ can be obtained using the $GetReTxCounts(K, s, h)$ function in Fig. 3, where $K$ is the total number of data packets.

After $\gamma$ and $\nu$ are obtained, the caching node $s$ can estimate the packet holding time, $T_b$, given as

$$T_b = \frac{(K \times h + \gamma + \nu \times h) \times S}{R}, \quad (9)$$

where $S$ is the size of a single packet and $R$ is the transmission rate. $GetReTxCounts(K, s, h)$ considers all possible retransmissions needed to deliver all the packets. Thus, $T_b$ represents the worst-case transmission time for data packets, and the cached data packets will not be removed before the retransmissions for lost packets are completed.

The proposed buffer management technique can also be applied to both end-to-end and hop-by-hop loss recovery schemes. For an end-to-end loss recovery scheme, $T_b$ at the source node is calculated using the hop-count from the source node to the destination node, i.e., $h$. Similarly, for a hop-by-hop loss recovery scheme, $T_b$ at each intermediate node is calculated based on $h = 1$.

```
HopCount(CR)
1. h ← 1
2. loop: P_d(s,h) > CR
3.       h ← h + 1
4. end loop
```

**Fig. 2** Function to obtain the hop-count $h$ for the next caching node.

```
GetReTxCounts(K, s, h)
1. k ← K, packets ← 0
2. γ ← 0, ν ← 0, success ← 0
3. loop: packets < K
4.       success ← [Γ_s(k, s, h)]†
5.       packets ← packets + success
6.       If packets < K
7.           ν ← ν + 1
8.           k ← k − success
9.           γ ← k × h + γ
10. end loop
```

**Fig. 3** Function to obtain the number of retransmitted packets and retransmission request messages.

## 4. Evaluation

This section discusses the simulation environment and evaluates the performance of the proposed buffer management technique. The simulated WSN consists of 300 homogeneous sensor nodes and operates under IEEE 802.15.4. Active Caching is used to provide reliable end-to-end transmissions with $CR$ values of 0.7 to 1. Six randomly chosen source nodes generate event data with the Poisson distribution, and each event consists of 20 packets with the packet size of 30 bytes. In addition, the wireless channel has a transmission rate of 250 kbps with a uniform link error of $p = 0.01$ and contention error $p_c$ based on $\lambda = 0.05$ for event-driven applications (see Sect. 3). The simulator was implemented with SMPL library [9], which is a C-language based event-driven simulator.

The proposed method is compared with the RTT-based method presented in [6] and the feedback-based method presented in [7] in terms of average buffer requirements and energy consumption.

Figure 4 compares the packet holding time of the proposed buffer management with RTT- and feedback-based methods for different $CR$ values. Both the proposed technique and the feedback-based method have significantly shorter packet holding time than the RTT-based method and are even shorter than one RTT. The feedback-based method requires slight longer packet holding time than the proposed method because it includes additional messages. In contrast, caching nodes in the RTT-based method perform retransmissions for lost packets within the first RTT and then they wait for second RTT to remove the cached packets from their buffers.

When a node receives data packets with a high $CR$, they are maintained in its buffer for a short period of time. Since a higher $CR$ induces smaller hop counts between two caching nodes, the time for retransmission requests is shorter and lost packets can be quickly be recovered. On the other hand, when a node receives data packets with a low $CR$, they are buffered for a long period of time. However, since hop counts between two caching nodes are larger, the packet de-

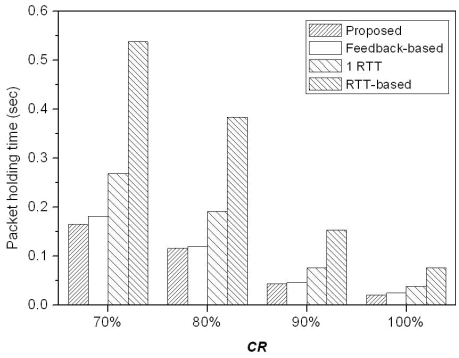


**Fig. 4** Packet holding times.

†$[x]$ is $n$, in case of $n - 0.5 \leq x < n + 0.5$.

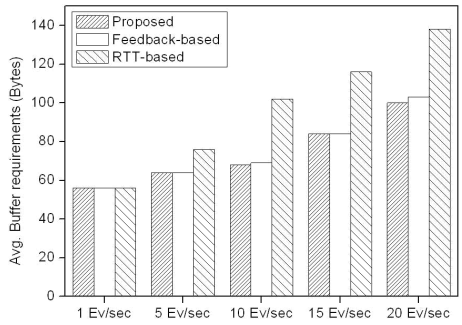


**Fig. 5** Average buffer requirements for the six data flows: *CR*s for the flows are randomly selected from a range of 70% to 100%.

**Table 1** Energy consumption for end-to-end transmissions.

| *CR* | Proposed | Feedback-based | RTT-based |
|---|---|---|---|
| 100 % | 98.7 mJ | 101.4 mJ | 98.7 mJ |
| 90 % | 102.4 mJ | 107.2 mJ | 102.4 mJ |
| 80 % | 113.5 mJ | 118.3 mJ | 113.5 mJ |
| 70 % | 115.6 mJ | 124.1 mJ | 115.6 mJ |

livery rate drops and frequent transmission failures occur. In addition, the time for a retransmission request is also longer, and thus, the packet holding time is longer for a lower *CR*.

Figure 5 shows that the average buffer requirements increase as the event generation rate increases. For the feedback-based method, the caching node that receives feedback messages can quickly remove packets from its buffer. In contrast, a caching node in the RTT-based method maintains data packets for a longer period of time. The proposed buffer management technique appropriately sets the packet holding time, which results in similar buffer requirements as the feedback-based method but without generating any additional messages. As shown in Fig. 4, since the feedback-based method shows slightly longer packet holding time than the proposed method, it can have larger buffer requirements than the proposed buffer management. This is significant especially under heavy load due to increased communication delay.

Table 1 shows the energy consumption for end-to-end transmissions. A simple energy consumption model based on LEACH [10] was employed for the simulation study. The energy consumption model assumes free space wireless channel and 10 m distance between two communicating nodes. The energy consumption is represented as $E = (0.11 \times 10^{-6}) \times l \times \tau$ *Joule* for $l$-bit data and $\tau$ total transmission count from a source node to a destination node. Although the feedback-based method has low buffer requirements, it has the highest energy consumption due to additional feedback messages. In contrast, the proposed method provides efficient buffer requirement as well as low energy consumption.

## 5. Conclusion

An efficient loss recovery mechanism based on retransmission of lost packets is needed for reliable transmissions in WSNs. Retransmissions require sensor nodes to maintain data packets in their buffers. Since tiny sensor nodes have limited resources and demand very low-power consumption, an efficient buffer management is necessary for not only high utilization but also low-power. The proposed buffer management technique provides efficient buffer utilization by estimating transmission times of data packets. Moreover, there is no energy overhead required to issue additional messages to remove data packets from the buffers.

### References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," IEEE Commun. Mag., vol.40, no.8, pp.102–114, Aug. 2002.
[2] D. Culler, D. Estrin, and M. Srivastava, "Guest editors' introduction: Overview of sensor networks," Computer, vol.37, no.8, pp.41–49, Aug. 2004.
[3] C.Y. Wan, A.T. Campbell, and L. Krishnamurthy, "PSFQ: A reliable transport protocol for wireless sensor networks," Proc. ACM International Workshop on Wireless Sensor Networks and Applications, pp.1–11, Sept. 2002.
[4] F. Stann and J. Heidemann, "RMST: Reliable data transport in sensor networks," Proc. IEEE International Workshop on Sensor Network Protocols and Applications, pp.102–112, May 2003.
[5] D.-Y. Kim and J. Cho, "Active Caching: A transmission method to guarantee desired communication reliability in wireless sensor networks," IEEE Commun. Lett., vol.13, no.6, pp.378–380, June 2009.
[6] Z. Xiao, K.P. Birman, and R.V. Renesse, "Optimizing buffer management for reliable multicast," Proc. IEEE International Conference on Dependable Systems and Networks, pp.187–196, June 2002.
[7] J. Paek and R. Govindan, "RCRT: Rate-controlled reliable transport for wireless sensor networks," Proc. ACM International Conference on Embedded Networked Sensor Systems, pp.305–319, Nov. 2007.
[8] I. Howitt and J.A. Gutierrez, "IEEE 802.15.4 low rate-wireless personal area network coexistence issues," Proc. IEEE Wireless Communications and Networking Conference, pp.1481–1486, March 2003.
[9] M.H. MacDougall, Simulating Computer Systems, Techniques and Tool, The MIT Press, 1987.
[10] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," Proc. 33rd Hawaii International Conference on System Sciences, Jan. 2000.