In-network RFID Data Filtering Scheme in RFID-WSN for RFID Applications*

Ali Kashif Bashir[1,**], Myong-Soon Park[2], Sang-Il Lee[1], Jinseop Park[1], Wongryol Lee[1], and Sayed Chattan Shah[3]

[1] National Fusion Research Center, Daejeon, Korea
{ali12,leesi,linupark,wrlee}@nfri.re.kr
[2] Department of Computer Science and Engineering, Korea University, Seoul, Korea
myongsp86@korea.ac.kr
[3] Electronics and Telecommunications Research Institute, Daejeon, Korea
shah@etri.re.kr

Abstract. In the integration of wireless sensor networks (WSN) and radio frequency identification (RFID), RFID data can use WSN protocols for multi-hop communication. Due to readers overlapped regions in dense areas and due to readers multiple read cycles, a lot of duplicate data is produced. Transmitting such duplicates towards base station waste node energies. In-network filtering of these duplicates can save transmission overhead, but on the other hand it increases computation cost. Delay is an important parameter in RFID applications that has not been considered yet by existing approaches. Both communication overhead and computation overhead can affect the delay performance in terms of queuing delay and processing delay respectively. Therefore, it is required to tune the filtering algorithm. In this paper, our in-network filtering scheme tend to find this trade-off between these two costs for better delay performance. In simulation part, we showed the effect of these costs on delay performance.

Keywords: In-network processing in RFID-WSN integrated networks, In-network filtering, Duplicate data filtering, Delay in RFID, Communication and computation cost.

1 Introduction

The next revolution in computing technology is the widespread of small wireless computing and communication devices; they will integrate seamlessly into our daily life [1]. In the near future we can expect lots of devices to grow by multiple orders of magnitude such as tags, sensors, readers, etc. By technology perspective, RFID and sensor networks are important components of this paradigm since both technologies can be used for coupling physical and virtual world usually named as pervasive computing [2].

* This work is supported by Ministry of Education Science &Technology, South Korea.
** Corresponding author.

J. Lee et al. (Eds.): ICIRA 2013, Part II, LNAI 8103, pp. 454–465, 2013.

WSNs are networks of small, cost effective devices with ability of sensing, processing, and communication. On the other hand, RFID technology provides identification to tagged objects or humans. It consists of reader, tags, and the application. Readers read tags attached to objects, store data in its memory, and applications access it. RFID technology does not support multi-hop communication; however, by integrating it with WSN, we can route RFID data from readers to base station by using sensor network protocols. For this, nodes can have both functionalities: sensing and reading as shown in Fig. 1. RFID and WSN can be integrated in several other ways discussed in literature [2], [3], [4].

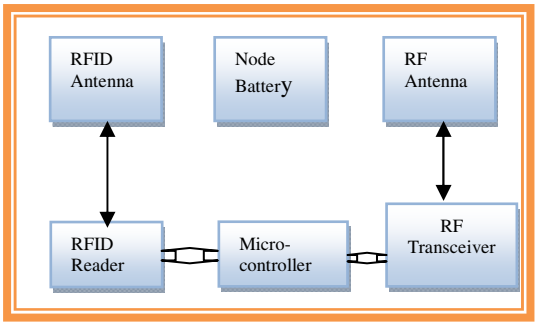

Fig. 1. Integrated WSN node and RFID Reader

RFID data is streaming in nature and usually contains an excessive amount of duplicate readings. In WSN, nodes are densely deployed usually and due to density they have overlapping regions with neighboring nodes. Tags that exist in overlapping areas are read by more than one reader and results in duplicate readings. Sending these duplicate data packets to the base station by multi-hop fashion consumes nodes energies, whereas, energy is a critical issue in WSN as nodes battery lifetime is limited.

By applying in-network processing we can filter these duplicates within the network to save extra transmissions. In-network processing reduces communication cost, but on the other hand increases computation cost. In the last few years, researchers found tradeoffs between computation and communication cost to increase energy efficiency [6], [7], [8]. WSN is tolerant to network delay since there is limited bandwidth [6, 9]. But, to provide better quality of service, improving network delay emerges as an important factor in WSNs. Delay can occur due to intensive communication on nodes or due to congestion [10].

In sensor networks, multiple packets can be aggregated into one due to the correlation of the sensed data. However, RFID data packets cannot be aggregated as every read data packet has its own identity, but due to duplications they can be filtered. In-network filtering drop RFID duplicate data packets to avoid redundant transmissions in the network. It reduces communication cost, but processing on nodes nodes result in computation overhead and in processing delays. RFID data contain real-time information and applications are interested in timely reports such as in a department store, managers would like to have the updated information about sales

and stocks. Therefore, a delay is an important metric to consider in RFID. In literature, several in-network filtering solutions [11], [12], [13], [14] are proposed, but they are focused on reducing communication overhead and do not consider delay.

In this paper, we propose IRDF :*In-network RFID Duplicate Data Filtering*, which is an extension of our previous work EIFS [14]. Topology and number of filtering points within network affect the performance of in-network filtering approaches. In IRDF, we choose clustering topology, as it helps in reducing energy consumption [15], whereas the filtering module runs only at cluster heads. Moreover, we vary the number of filtering points in the network and measure the communication cost, computational cost, and delay. On the base of these results, IRDF chooses the optimal filtering points that provide best delay performance.

The rest of this paper is organized as follows: Section 2 discusses related work. In section 3, we presented the assumptions and a preliminary algorithm. In section 4, we describe the proposed approach in detail. We analyze our idea and compare it with previous research by using simulations in section 5. Finally, in section 6, we conclude this paper.

2 Related Work

Data filtering is as an important issue in RFID applications. Several researchers provided solutions to filter RFID data to save communication cost. [5, 16] proposed their approaches to filter duplicate data using sliding-window. Sliding window keeps the history of the previous read cycles in buffer and output the data when it increases than a certain threshold. These solutions are proposed for server middleware. This middleware can be implemented in the readers, but due to the limited memory of readers this is not an appropriate solution. Moreover, the performance of this approach degrades with the smaller size of the window and filtering redundant data at base station do not decrease the transmission overhead on nodes.

In-network processing is widely researched in WSN in terms of data aggregation [10, 17]. In WSN data is highly correlated; therefore, parent nodes or cluster heads can aggregate multiple data packets into one. While RFID data is not correlated as each EPC tag represents one real world object. However, due to the enormous amount of duplication in RFID data, we need to perform in-network data filtering to avoid transmitting duplicate data within the network. Following are the duplication types that need to be filtered by in-network filtering solutions:

- *Data level:* Multiple tags with same EPC (Electronic Product Code) are attached to the same object in order to reduce missing rate and increase reliability [5].
- *Multiple Read Cycle:* Tags in the vicinity of a reader for a long time (in multiple reading cycles) are read by the reader multiple times [18]
- *Redundant Reader:* Multiple readers are installed to cover a larger area or distance, and tags in the overlapped areas are read by multiple readers [19] as shown in Fig. 2, where such as tag T3 is read by three readers R2, R3 and R4.

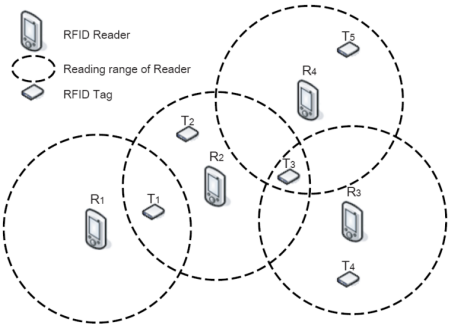

Fig. 2. Redundant Reader: Duplication due to overlapping of readers

Data level and multiple read cycle duplication can be eliminated by a simple filtering algorithm at the reader. To eliminate duplication generated due to overlapping, readers need to collaborate with each other. Such as [19] deals with the problem of redundant readers. This solution resolves the problem by temporarily deactivating the readers that have a maximum overlapped region with neighboring readers. This mechanism reduces the redundant transmission, but in large deployments finding which readers to be turned off is an NP-hard problem [21].

In-network processing also increases the computation overhead on nodes. An efficient solution should create a balance between communication and computation costs to meet the desired objectives of applications. Kadayif et al. [20] proposed a strategy to keep a balance between computation energy and communication energy in wireless sensor networks. In many streams-oriented WSN applications such as military scenarios, both link bandwidth and node energy are constraint resources. For such applications, in-network processing imposes non-negligible computational cost. In the last few years, researchers tried to tune the trade-offs among computation and communication cost to increase the network lifetime by increasing the lifetime of sensor nodes.

In RFID-WSN, routing RFID data in presence of duplicates waste nodes energies. Data should be filtered within the network ideally close to source to avoid redundant transmissions to save energy. Existing solutions [11], [12], [13], and [14] filter the data with less computation cost. These schemes mainly use two kinds of topologies: tree and clustering. Both of these technologies have their own pros and cons as explained below.

Tree based approaches: In the tree approaches [11], [13], node filters its own and children's data. Amount of data being filtered at one node is less. To filter the maximum amount of duplicate data, filtering algorithm need to run at every node for every incoming data packet which results in high computation cost and delay. INPFM [11] was the first in-network filtering approach in RFID-WSN. Dong-Hyun et al. [13] proposed in-network filtering scheme in object tracking applications. He assumed that data should meet within network, which is a strong assumption.

Cluster based approaches: To improve performance of filtering, CLIF [12] and EIFS [14] used clustering topology. They divided the redundancy into: intra-cluster

redundancy and inter-cluster redundancy. Intra-cluster redundancy is being filtered at local CH and inter-cluster redundancy at intermediate cluster heads. These schemes save computation overhead; on the other hand filters fewer amounts of data in comparison with [11]. EIFS [14] adopts a neighbor discovery algorithm for fast detection of duplicates. To filter inter-cluster data at neighboring cluster heads, they use feedback message. CHs update their routing tables to drop duplications on neighboring cluster heads.

The objective of these approaches was to filter the maximum amount of data to reduce communication overhead. On the other hand, huge traffic results in congestion and queuing delay. Therefore, it is important to choose an appropriate amount of data filtering at one node or number of nodes to filter data without increasing delay. None of the existing in-network filtering approaches have considered delay as a performance metric.

3 Preliminaries

In our scheme, duplication is divided into two: intra-cluster and inter-cluster duplication. Nodes that have overlapping with nodes of same cluster are called as intra-cluster nodes, whereas, nodes that have overlapping with neighboring nodes of another cluster are called inter-cluster nodes.

3.1 Assumptions

Following are the assumption of our model.

- Nodes are homogeneous in nature and static after deployment.
- Transmission range is double than the reading range and nodes can communicate with CH directly.
- Every sensor node contains a reader module.
- We assume a simple communication mechanism with a medium access control (MAC) protocol that ensures no collision and interference [10].
- Cluster heads will send data to the base station via intermediate cluster heads (multi-hop).
- Filtering process will run only at cluster heads.
- Clusters are static in nature, but the cluster head task can be rotated among member nodes.
- Data loss and possible contention are not considered.

3.2 Redundancy Definition

Check for duplication if following three conditions are true.

- The contents of tags (EPCs) are the same: In this research, we will only compare EPC serial numbers using BFF algorithm [22].

- The node IDs or cluster ID is the same in the intra-cluster phase. In an inter-cluster, if the data is from my neighboring cluster.
- The difference between the reading times of two data is less than a predefined time interval.

Cluster head will store a copy of tag data in a tag list for a certain time period for comparison with upcoming data to drop the duplications. In our research, we compare 32 bits of EPC Serial numbers. For that we used backward first search algorithm [22]. In our previous work EIFS; we compared 36 bits of EPC serial numbers. Therefore, to improve the performance of detection in IRDF, we compare 32 bits of Serial Number. The time interval can be defined as reporting time for cluster heads to send data.

3.3 ND Array

In our approach after cluster formation to distinguish among nodes, each node exchanges an *ND message* with neighboring nodes. The *ND message* contains *node ID* and *cluster ID*. A node that receives ND messages from its neighbors keeps the *cluster ID* in an *ND array*. From the *ND array* of a node, we can know whether it has the *ID* of any neighboring clusters or not. If the *IDs* of two or more than two clusters exist in a node *ND array*, it will be considered as an *inter - cluster node*. *ND message* helps in decreasing computation cost.

When an intra-cluster node, interrogates a tag x, after tag response, node generates an RFID data packet with the value of f (number of remaining filtering operations) as 1, shown in table 2. The node sends this data packet to its cluster head. If any other neighboring node also reports to the cluster head with tag x, the cluster head will filter it to avoid duplication. In case of inter-cluster nodes, the value of f is f_e. However, in this scheme we vary the f_e to monitor the network performance. Table 1 shows the structure of the RFID data packet in inter-cluster node. Every node sends their data to its cluster heads and they decide the type of the sender from the f field. If the value of f is 2 or more, the sender is considered an inter-cluster node. If the value of f is 1, the sender is intra-cluster node.

Table 1. The structure of the node generated data packet

	Tag *ID*	Node *ID*	Time Stamp	*f*
Intra-cluster Node	*x*	*N*	*T*	*1*
Inter-cluster Node	*x*	*N*	*T*	f_e

4 Proposed Algorithms

IRDF: In-network RFID data filtering scheme is an extension of our previous scheme named *EIFS: energy efficient in-network RFID data filtering scheme* [14] in terms of delay. Like EIFS, IRDF also adopts cluster topology. Process of intra-cluster duplication is similar like EIFS, however, in inter-cluster EIFS filter data at every

node and inform source cluster heads with feedback messages. In IRDF, we do not use any feedback mechanism and change routing paths. Feedback messages introduce heavy communication overhead that increase latency. Rather we find the optimal number of filtering points that provide best delay performance.

When a cluster head receives an RFID data packet, it decides the type of sender by the *f* field. If the value of *f* is 1, the sender is an intra-cluster node and local cluster head execute the duplicate detection algorithm to check for the duplication. After removing the duplication, it sets the *f* field as 0 and forwards the data to base station. Such packets will not be filtered at any intermediate cluster head which saves computation costs. At intermediate CH when the value of *f* of arrival packet is 0, it means the data is already filtered. This mechanism significantly reduces the number of comparisons. Detailed algorithm is given in fig. 3.

Intermediate CH's will check the value of *f,* if value of *f* is more than 0, they will perform duplicate detection mechanism for inter-cluster duplicates. The value of *f* decreases by 1 with each hop from source cluster head to intermediate cluster heads. Detailed inter-cluster duplicate data filtering algorithm is presented in fig. 4.

In EIFS, we filter data at every intermediate CH. Intermediate heads detects the duplication and inform source cluster heads that cause inter-cluster duplication with feedback messages. In the next rounds, they can change the routing path of that specific tags to drop duplications on neighboring cluster heads. IRDF filters inter-cluster duplicate data at neighboring cluster heads of a specific hop count. EIFS performance also degrades when tag mobility is high in the network.

```
Function Intra_cluster_duplicate_data filtering ()
Loop until I am cluster head
  If incoming data packet comes then
   If data.number_of_remaining filtering is 1
    // Intra-cluster duplication//
      Decrease data.number_of_remaining_filtering by 1.
      If it is not duplicated data then
          Update the tag_list.
          Send the data to the sink.
      Else
          Drop the data.
      End if
   End if
  Else if data.number_of_remaining_filtering is ∞ or 0 then
       // Inter-cluster duplication required to filtering//
       Call inter_cluster_duplicated_data filtering.
  Else if data.number_of_remaining_filtering is 0 then.
      Send the data to the next hop node.
   End if
  End if
  End loop
```

Fig. 3. Intra-cluster filtering algorithm

```
Function Inter_cluster_duplicate_data_filtering
  Seek the data.tag_id from the tag_list.
 If found then
  Decrease the value of f by 1
 If the data is duplicated then
Else
  Second the data to the next hop.
End if
Else
    Inset the data into tag_list.
     End if
Else
  Send the data to the next hop
    End if
End if
```

Fig. 4. Inter-cluster filtering algorithm

5 Simulation Results

In our previous work [14] we presented and compared our algorithms in detail with [11] and [12] in terms of communication cost and computation cost. However in this work, we will vary the number of detection points and measure the performance of the algorithm in terms of communication cost, computational cost, and delay. Computation cost and communication cost have a trade-off and delay is chosen as a decisive performance metric. We developed our simulator using C++. The detailed simulation environment is given in table 2.

Table 2. Simulation environment

Parameters	Value
Field Area	100 x 100 m^2
Number of nodes	361
Number of clusters	19
Members in a cluster	19 (including cluster head)
Reading Range	5 *m*
Transmission Range	10 *m*
Distance between nodes	7 *m*
Reading interval	2 *sec*
Duplication ratio	20 %
Number of tags	100 to 500
Limit of History Data	300

α is introduced as a variable representing number of intermediate cluster heads (detection points/filtering points) from source cluster heads to perform in-network filtering. We vary value of α and measure the computation cost, communication cost, and delay. Fig. 5 shows the computation cost of our algorithm in terms of number of computations. When we filters data at more intermediate CH, we require more comparisons to detect duplicates. Logic is very simple, if we filters data at every intermediate node for every arriving packet, it requires more comparisons to detect and hence increase computation cost.

Fig. 5. Computational cost

In fig. 6, we measured the communication cost in terms of number of relays required to send data from source to base station. It is visible that if value of α increases, number of relays decreases. Decreasing number of relays clearly means a reduction in the amount of packet transfer. In other words, if we filter data at more nodes, we can able to filter more duplicate data and can save more transmission overhead. In literature, approaches that filter data at more intermediate points have better performance in terms of communication cost [11], [13]. It is because, when α is 1, nodes have to transmit more packets to forward all read data to the base station in

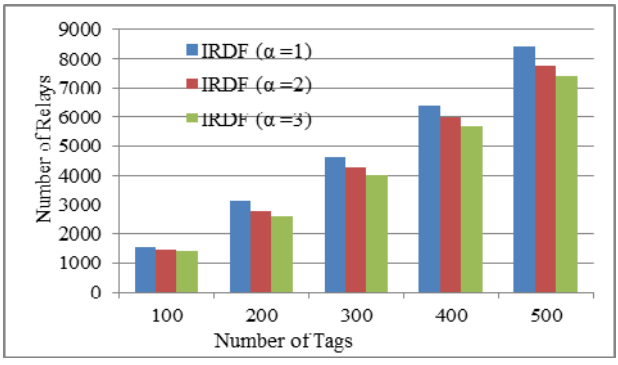

Fig. 6. Communication cost

the presence of duplicates. When α increases, the performance of in-network filtering algorithms improves in terms of communication cost.

From figure 5 and 6, it is clear that computation cost increase with increasing number of filtering points whereas computation cost decreases. Therefore, this tradeoff needs to be considered to use network resources efficiently. However, in IRDF we considered delay to be a deciding factor. In simulation, we assume the network delay the time it takes to reach the destination from source. The result of the delay parameter is quite different than computation and communication costs. When value α is 1 and 3, delay is high. When α is 1, communication cost is high as nodes have to forward a lot of packets which results in queuing delay at nodes. On the other hand, when α is 3, computation cost gets higher and that results in processing delay at nodes. Packets have to wait to be processed before being forward. However, in case of α as 2, delay is mediate as shown in fig. 7.

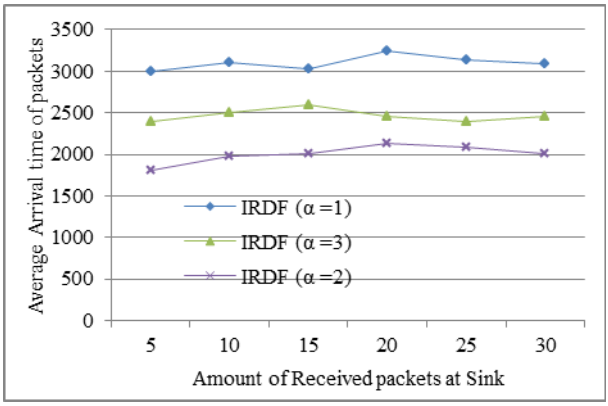

Fig. 7. Delay in terms of arrival time of packets at base station

6 Conclusions

In the integration of RFID-WSN, RFID data contain duplications which need to be filtered to avoid redundant transmissions hence to save node energies. Existing in-network filtering approaches tend to filter maximum duplicate data and results in increased computation cost and processing delay. Moreover they have not considered delay as a performance metric, whereas, a delay is an important factor in RFID applications. In this paper, our approach divides duplication into intra-cluster and inter-cluster. Intra-cluster duplication is being filtered at local CH and inter-cluster duplications at neighboring CHs. In simulation section, we monitored the trade-off between computation and communication costs. Moreover, considering delay as important parameter, we selected it to be deciding factor for selecting the appropriate number of filtering points within the network.

References

1. Sarma, S.E.: Towards the five-cent tag. Tech. Rep. MIT-AUTOID-WH-006, MIT Auto-ID Center (2001), http://www.autoidcenter.org/research/MIT-AUTOID-WH-006.pdf (accessed on January 2009)
2. Lopez, T.S., Kim, D., Canepa, G.H., Koumadi, K.: Integrating Wireless Sensors and RFID Tags into Energy-Efficient and Dynamic Context Networks. The Computer Journal 52, 240–267 (2008)
3. Ho, L., Moh, M., Walker, Z., Hamada, T., Su, C.F.: A Prototype on RFID and Sensor Networks for Elder Healthcare: Progress Report. In: Proceedings of the ACM SIGCOMM Workshop on Experimental Approaches to Wireless Network Design and Analysis, Philadelphia, PA, USA, pp. 70–75 (August 2005)
4. Smith, J.R., Fishkin, K.P., Jiang, B., Mamishev, A., Philipose, M., Rea, A.D., Roy, S., Rajan, K.S.: RFID-based techniques for human-activity detection. Commun. ACM 48, 39–44 (2005)
5. Jeffery, S.R., Garofalakis, M., Franklin, M.J.: Adaptive Cleaning for RFID Data Streams. In: Proceedings of the 32nd International Conference on Very Large Data Bases VLDB, Seoul, Korea, September 12-15, pp. 163–174 (2006)
6. Yang, Y., Krishnamachari, B., Prasana, V.K.: Data Gathering with Tunable Compression in Sensor Networks. IEEE Transactions on Parallel and Distributed Systems 19, 276–287 (2008)
7. Qinghai, W.: Traffic Analysis & Modeling in Wireless Sensor Networks and Their Applications on Network Optimization and Anomaly Detection. Network Protocols and Algorithms 2, 74–96 (2010)
8. Seung, J.B., Gustavo, D.V., Xun, S.: Minimizing Energy Consumption in Large-Scale Sensor Networks through Distributed Data Compression and Hierarchical Aggregation. IEEE Journal on Selected Areas in Communications 22, 1130–1140 (2004)
9. Jin, Y., Wei, D., Gluhak, A., Moessner, K.: Latency and Energy-Consumption Optimized Task Allocation in Wireless Sensor Networks. IEEE Wireless Communications and Networking Conference (WCNC), 1–6 (April 2010)
10. Krishnamachari, L., Estrin, D., Wicker, S.: The Impact of Data Aggregation In Wireless Sensor Networks. In: Distributed Computing Systems Workshops, pp. 575–578 (2002)
11. Choi, W., Park, M.S.: In-network Phased Filtering Mechanism for a Large-Scale RFID Inventory Application. In: Proceedings of the 4th International Conference on IT & Applications (ICITA), Harbin, China, pp. 401–405 (January 2007)
12. Kim, D.-S., Bashir, A.K., Ming, X., Kim, J.-H., Park, M.-S.: Energy Efficient In-Network Phase RFID Data Filtering Scheme. In: Sandnes, F.E., Zhang, Y., Rong, C., Yang, L.T., Ma, J. (eds.) UIC 2008. LNCS, vol. 5061, pp. 311–322. Springer, Heidelberg (2008)
13. Lee, D.H., Lee, E.-M., Bashir, A.K., Park, M.-S.: Efficient in-network redundancy filtering in RFID system integrated with wireless sensor networks. In: Networked Computing 6th International Conference, pp. 1–6 (May 2010)
14. Bashir, A.K., Lee, S.-J., Hussain, S.H., Park, M.-S.: Energy Efficient In-network RFID Data Filtering Scheme in Wireless Sensor Networks. In: Sensors MDPI 2011, vol. 11, pp. 7004–7021 (2011)
15. Kawadia, V., Kumar, P.R.: Power Control and Clustering in Ad Hoc Networks. In: Proceedings of IEEE INFOCOM, vol. 1, pp. 459–469 (April 2003)
16. Wang, F., Liu, P.: Temporal Management of RFID Data. In: Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, pp. 1128–1139 (August 2005)

17. Yujie, Z., Ramanuja, V., Seung-Jong, P., Raghupathy, S.: A Scalable correlation aware aggregation strategy for wireless sensor networks. Inform. Fusion 9, 354–369 (2008)
18. Carbunar, B., Ramanathan, M.K., Koyuturk, M., Hoffmann, C., Grama, A.: Redundant Reader Elimination in RFID Systems. In: Proceedings of the Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, IEEE SECON 2005, Santa Clara, CA, USA, September 26-29, pp. 176–184 (2005)
19. Bai, Y., Wang, F., Peiya, L.: Efficiently Filtering RFID Data Streams. In: Proceedings of the First International VLDB Workshop on Clean Databases, Seoul, Korea (September 2006)
20. Kadayif, I., Kandemir, M.: Tuning In-Sensor Data Filtering to Reduce Energy Consumption in Wireless Sensor Networks. In: Proceedings of Design, Automation and Test in Europe Conference and Exhibition, Paris, France, pp. 1530–1539 (February 2004)
21. Chawathe, S.S., Krishnamurthy, V., Ramachandran, S., Sarma, S.: Managing RFID Data. In: Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, ON, Canada, pp. 1189–1195 (August 2004)
22. Boyer, R.S., Moore, J.S.: A Fast Searching Algorithm. In: CACM, pp. 762–772 (1997)
23. Bashir, A.K., Chauhdary, S.H., Shah, S.C., Myong-Soon, P.: Mobile RFID and its Design Security Issues. IEEE Potentials 30, 34–38 (2011)