# ETRI-QM: Reward Oriented Query Model for Wireless Sensor Networks

Yang Jie, Shu Lei, Wu Xiaoling, Jinsung Cho, Sungyoung Lee, Sangman Han

Department of Computer Engineering
Kyung Hee University, Korea
{yangjie, sl8132, xiaoling, sylee, i30000}@oslab.khu.ac.kr
{chojs}@khu.ac.kr

**Abstract.** Since sensors have a limited power supply, energy-efficient processing of queries over the network is an important issue. As data filtering is an important approach to reduce energy consumption, interest is used to be a constraint to filter uninterested data when users query data from sensor networks. Within these interested data, some of them are more important because they may have more valuable information than that of the others. We use 'Reward' to denote the importance level of data. Among the interested data, we hope to query the most important data first. In this paper, we propose a novel query model ETRI-QM and a new algorithm ETRI-PF (packet filter) dynamically combines the four constraints: Energy, Time, Reward and Interest. Based on our simulation results, we find out that our ETRI-QM together with ETRI-PF algorithm can improve the quality of the information queried and also reduce the energy consumption. [1]

## 1 Introduction

Wireless sensor networks are envisioned to consist of large numbers of devices, each capable of some limited computation, communication and sensing, operating in an unattended mode. One unifying view is to treat them as distributed databases. The simplest mechanism to obtain information from this kind of database is to use queries for data within the network. However, most of these devices are battery operated, which highly constrains their life-span, and it is often not possible to replace the power source of thousands of sensors. So how to query with the limited energy resources on the nodes is a key challenge in these unattended networks.

Researchers have noted the benefits of a query processor-like interface to sensor networks and the need for sensitivity to limited power and computational resources [1, 3, 6, 7, 9,]. Prior systems, however, tend to view query processing in sensor networks simply as a power-constrained version of traditional query processing: given some set of data, they strive to process that data as energy-efficiently as possible. Typical strategies include minimizing expensive communication by applying aggregation and filtering operations inside the sensor network.

[1] Dr. Sungyoung Lee is the corresponding author.

In our paper, we present a novel query model (ETRI-QM) to query the data with more important information among the interested data. By using this query model, we can dynamically combine these four constraints (Energy, Time, Reward, and Interest) to provide diverse query versions for different applications. Within our query model, each packet has four parameters: (1) energy consumption of the packet; (2) processing time of the packet; (3) important level of the packet; and (4) interest level of the packet. By using this ETRI-QM, we can achieve the following contributions: (1) Using *interest constraint* as the threshold to filter the uninterested incoming packets to reduce the energy consumption; (2) Using *reward constraint* to choose the high quality information and minimize the queried packet number to minimize the energy consumption but still satisfy the minimum information requirement.

The remainder of the paper is structured as follows. In the next section, we describe the related work. We introduce our query/event service APIs to illustrate the design of ETRI-QM in Section 3. The main principle of our ETRI-QM is in Section 4. In the simulation, we examine the performance of our query model and compare with the other different query plans (Section 5). Finally the paper is concluded in Section 6.

## 2 Related Work

In [8], the authors present a sensor information networking architecture called SINA, which facilitates querying, monitoring, and tasking of sensor networks. To support querying within sensor networks, they design a data structure kept inside the sensor nodes based on the spreadsheet paradigm. In the spreadsheet paradigm, each sensor node maintains a logical datasheet containing a set of cells. By defining the semantic of a cell to specifying scope of the query, the information can be organized and accessed according to specific application needs, and also the number of the packets need to be sent can be reduced, thus the energy consumption will be reduced. However, there exist a tradeoff between the energy cost to run SINA on each sensor node and the energy reduced by using SINA.

In [2], our work also has some similarities to techniques proposed, the authors introduced a new real-time communication architecture (RAP) and also a new packet scheduling policy called velocity monotonic scheduling (VMS). VMS assigns the priority of a packet based on its requested velocity. This work differs from our work in two aspects: one is that the cost-model is different in the two scenarios–in RAP is primarily reducing the end-to-end deadline miss radio while we are minimizing energy consumption and maximizing the querying quality; the second one is that RAP intends to maximize the number of packets meeting their end-to-end deadlines without considering their value (reward, importance level), and in our model, we take reward an important constraint to deal with the queries.

Samuel et al discussed the design of an acquisitional query processor (ACQP) for data collection in sensor network in [10]. They provide a query processor-like interface to sensor networks and use acquisitional techniques to reduce power consumption. Their query languages for ACQP focus on issues related to when and how often samples are acquired. To choose a query plan that will yield the lowest overall power

consumption, the query is divided into three steps: creation of query, dissemination of query and execution of query. Optimizations are made at each step.

Our ETRI-QM combines four constraints (**energy, time, interest** and **reward)** to maximize the querying quality with minimum energy consumption. In papers [4, 5]. Cosmin Rusu, *et al.* first time consider **Energy**, **Time**, and **Reward** these three constraints simultaneously while Reward denotes the important level of tasks. They believe that among a set of tasks of real time applications, some of them are more valuable than the others. So instead of processing several unimportant tasks just consuming less energy, it is more meaningful to process one valuable task consuming more energy. In our query model, we use **reward** to denote the importance level of data, so that we can transmit the data with more valuable information first. By considering the four constraints simultaneously, we make out our target that is to query the most valuable (**reward**) packets from the **interested** area to be transmitted while meeting **time** and **energy** constraints.

## 3 ETRI-QM

Applications may submit queries or register for events through a set of query/event service APIs. The APIs provides a high-level abstraction to applications by hiding the specific location and status of each individual node. These APIs allow applications to specify the timing constraints as well as other constraints of queries.

ETRI-QM provides the following query/event service APIs.

**Query {attribute_list, interested_area, system_value, timing_constraints, querier_loc}**

Issue a query for a list of attributes in an interested area with the maximum system value (reward). Attributes refer to the data collected by different types of sensors, such as temperature sensors, humidity sensors, wind sensors, rain sensors etc. Interested area specifies the scope of the query, the area from which data is needed by the users. System value is defined as the sum of selected packets' reward. Timing_constraints can be period, deadline and so on. If a period is specified for a command, query results will be sent from the interested area to the issuer of query periodically. The querier_loc is the location of the base station that sends out the query.

Imagine a heterogeneous network consisting of many different types of sensors: temperature sensors, humidity sensors, wind sensors, rain sensors etc. monitoring the chemical found in the vicinity of a volcano. Suppose the volcano has just broken out, and we want to know which five chemicals found have the highest particle concentration. Obviously, sensors near the volcano will have more valuable data, which means that the importance levels of these data are much higher than those of the data collected by the further sensors. Thus, here we can consider reward to be the distance between sensors and the volcano.

Consider another example: lots of sensors are deployed in some area with different densities. For the EventFound case, take noise into account, the data collected by sensors having higher densities will be more reliable. So, here the reward is changed to be the density of the sensors around the interested area in the network.

There is one more example to make you clearly understand the concept of "reward". In the case of real-time communication for wireless sensor network, meeting the end-to-end deadline seems to be the most importance issue, so that we can consider arriving time to be the reward value. Reward is defined to be the importance level of the data collected by sensors. In the case of different wireless sensor networks, it can be specified to various formats.

A query is send to every node in the interested area specified in the API, and the results will first be sent back to the cluster head, then the cluster head will use the algorithm we will introduce in next section to decide the packets to be sent back to the base station of which the location is also provided by the API.

## 4 ETRI-PF

After receiving the query message, the sensor nodes will start to collect related data and then send the packets to the cluster head. The cluster can be formed using LEACH or other techniques. In terms of the cluster head, many unprocessed packets are still physically existing in different sensor nodes and waiting for the processing of cluster head. Therefore, in sensor network, except the cluster head, all the other sensor nodes which are going to send packets to the cluster head can logically be considered as a buffer, since all of these packets are waiting for the processing of cluster head. We regard this buffer as the **First Tier Buffer (FTB)**. Actually the FTB is a logical concept for cluster head. The **Second Tier Buffer** (**STB**) is the buffer that physically exists inside cluster head. Since many sensor nodes will send packets to cluster head, obviously, cluster head needs buffer to store these received packets. Therefore, we propose the Two Tiers Buffer model for wireless sensor network as the figure 1 shows.

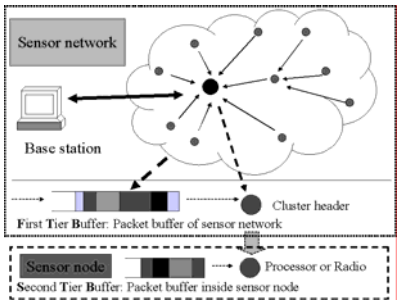


**Figure 1.** Two tiers buffer

The main contribution of this paper is the ETRI-PF algorithm used in FTB to filter and accept packets. In FTB, what we want is to **Maximize reward value to meet the Reward constraint (in terms of system_value in the query/event service APIs)**. The key idea of this algorithm is that instead of processing two or more unimportant packets which just consume a small amount of energy we would like to process one important packet which may consume relatively larger amount of energy.

Reward value is used to denote the important level of packet. A packet with a larger reward value means that this packet is more important. Therefore, the sensor nodes always accept packets which have the highest reward value. Thus, we can guarantee that the most important packets can be processed first.

After deciding which packets are to be accepted, the algorithm will also arrange the packets according to their value. Packets with the largest value will be sent to STB first, meanwhile, FTB will sum the reward value of all the packets having been sent to STB. If the summation is up to the system_value defined in the query/event service APIs, no more packets will be sent to STB. That is to say, all the packets having been sent to STB is enough to solve the query.

Based on this Two Tiers Buffer model and the algorithms above, we introduce the details of our ETRI packet scheduling principles. The principles of ETRI-PF are as follows:

(1) Whenever a new packet is accepted, its energy consumption should not exceed the remaining energy;
(2) Whenever a packet is processed, it must meet its deadline;
(3) Every packet can under Energy, Timing, Reward, and Interest constraints simultaneously;
(4) It is not necessary to always under these four constraints at the same time;

We can dynamically compose these constraints to filter and schedule packet for heterogeneous sensor nodes and divers working purposes.

### 4.1 Problem Formulation

We define the interested areas as $A \subseteq \{A_1, A_2 \ldots A_M\}$. From each interested area $A_x$ the cluster head can accept a subset of packets $P_x \subseteq \{ P_{x,1}, P_{x,2,}\ldots, P_{x,N} \}$.The processing time of the packet $P_{x,y}$ is denoted by $T_{x,y}$. Associated with each packet $P_{x,y}$ there is an Interest value $I_{x,y}$ and a Reward value $R_{x,y}$. Interest value is used to distinguish the interested packets from different areas. Reward value is used to denote the important level of this packet. The larger reward value means the higher important level. These four constraints of algorithm are defined as follows:

- The *energy constraint* imposed by the total energy $E_{max}$ available in the cluster head. The total energy consumed by the accepted packet should not exceed the available energy $E_{max}$. In other words, whenever the cluster head accept one packet, the energy consumption $E_{x,y}$ of this packet should not be larger than the remaining energy *RE*.
- The *time constraint* imposed by the global deadline *D*. The common deadline of this user's data query is *D*. Each packet that is accepted and processed must finish before *D*.
- The *interest constraint* imposed by the interest value threshold *IT*. Each packet that is accepted and processed must satisfy the interest value threshold $IT_{min} \leq I_{x,y} \leq IT_{max}$.
- The *reward constraint* imposed by the *value ratio* $V_{x,y}$ ($V_{x,y} = R_{x,y} / E_{x,y}$) between reward value $R_{x,y}$ and energy consumption of packet $E_{x,y}$. The larger $V_{x,y,}$ the packet has, the more valuable the packet is.

The ultimate goal of ETRI-PF is to query a set of packets $P = P_1 \cup P_2 \cup \ldots \cup P_M$ among interested packets to maximize the *system value* which is defined as the sum of selected packets' *value ratio* $V_{x,y}$ to meet the system_value defined in the query/event service APIs. Therefore, the problem is to

$$\text{Maximize} \sum_{x \in A, y \in P} V_{x,y} \quad \leq \text{system\_value} \tag{1}$$

$$\text{Subject to} \sum_{x \in A, y \in P} E_{x,y} \leq E_{\max} \tag{2}$$

$$\sum_{y \in P} T_{x,y} \leq D \tag{3}$$

$$IT_{\min} \leq I_{x,y} \leq IT_{\max} \tag{4}$$

$$x \in \mathrm{A} \tag{5}$$

$$\mathrm{A} \subseteq \{A_1, A_2 \ldots A_M\} \tag{6}$$

$$y \in P_x \tag{7}$$

$$P_x \subseteq \{1, 2 \ldots \mathrm{N}\} \tag{8}$$

Since $P = P_1 \cup P_2 \cup \ldots \cup P_M$, we can have the following equation as:

$$\sum_{x \in A, y \in P} V_{x,y} = \sum_{A_1, y \in P_1} V_{A_1, y} + \sum_{A_2, y \in P_2} V_{A_2, y} + \ldots + \sum_{A_M, y \in P_M} V_{A_M, y} \tag{9}$$

From equation (9), we can find that the real problem of ETRI-PF is to find out the minimum subset of $P_x \subseteq \{1, 2 \ldots \mathrm{N}\}$ to maximize the *system value* to system_value from each interested area $A_x$. Thus, the problem is changed to

$$\text{Maximize} \sum_{A_x, y \in P_x} V_{x,y} \quad \leq \text{system\_value} \tag{10}$$

$$\text{Subject to} \quad E_{x,y} \leq RE \tag{11}$$

$$\sum_{y \in P} T_{x,y} \leq D \tag{12}$$

$$IT_{\min} \leq I_{x,y} \leq IT_{\max} \tag{13}$$

$$x \in \mathrm{A} \tag{14}$$

$$\mathrm{A} \subseteq \{A_1, A_2 \ldots A_M\} \tag{15}$$

$$y \in P_x \tag{16}$$

$$P_x \subseteq \{1, 2 \ldots \mathrm{N}\} \tag{17}$$

Inequality (11) guarantees that the *time constraint* is satisfied. Inequality (12) guarantees that only the interested packets are accepted, and inequality (13) guarantees that the energy budget is not exceeded. In order to solve the problem that is presented by (10)-(17), we give the following steps for our ETRI-PF algorithm.

### 4.2 Steps of ETRI-PF

Before sending the real data of a packet to cluster head, sensor node can send its packet's parameters to the cluster head by including them in a small packet, which just consumes very limited energy. We give a name to this kind of small packet as *Parameter Packet (PP)*. There is a physical buffer that exists inside cluster head to store these *PPs*. After receiving these *parameter packets*, cluster head can decide which packet to be accepted and which packet should be discarded based on these sent parameters. In terms of this Two Tiers Buffer model, basically, we can define our ETRI-PF algorithm into the following steps:

**Step 1: Initialization.** After receiving PP $\subseteq \{PP_1, PP_2, \ldots, PP_N\}$, we assume that tables exist inside the cluster head for storing parameters of every packet $i$ ($i \in$ PP): *energy consumption* $E_{x,y}$, *processing time* $T_{x,y}$, *reward value* $R_{x,y}$, and *interest value* $I_{x,y}$. For each $PP_i$, there are energy consumption for checking $CE_i$ and a period of time for checking $CT_i$. We also use two structure arrays, *considered(i)* and *selected(i)* of size N, to store the information for all received *PPs*. Initially, we start with an empty schedule (*selected(i).status = false*) and no *PP* is considered (*considered(i).status = false*). The set of selected *PPs* (initially empty) is defined as S = {*(i) | selected(i).status = true*}. After selecting the *PPs*, cluster head accepts packets that are corresponded to these selected *PPs*. Therefore, packet's parameters can be expressed as *considered(i).*$E_{x,y}$, *considered(i).*$T_{x,y}$, *considered(i).*$R_{x,y}$, *considered(i).*$I_{x,y}$, *selected(i).*$E_{x,y}$, *selected(i).*$T_{x,y}$, *selected(i).*$R_{x,y}$, and *selected(i).*$I_{x,y}$. We define five variables: 1) *checking energy* ($\sum_{i \in PP} CE_i$) is used to store the total energy consumption for checked *PPs*; 2) *checking time* ($\sum_{i \in PP} CT_i$) is used to store the total processing time for checked *PPs*; 3) *processing energy* ($\sum_{i \in PP}$ *selected(i).*$E_{x,y}$) is used to store the total energy consumption for processed packets; and 4) *processing time* ($\sum_{i \in PP}$ *selected(i).*$T_{x,y}$) is used to store the total processing time for processed packets. 5) *system value summation* ($\sum_{i \in PP}$ *selected(i).*$R_{x,y}$) is used to store the total value for packets to be processed in STB. These five variables are all initialized to zero.

**Step 2: In FTB, we filter and accept packets based on the ETRI constraints**. A packet that can be accepted should satisfy all the following criteria:

- This packet's *PP* is not considered before (*considered (i).status = false*).
- The current schedule is feasible (*checking time* + *processing time*) ≤ *D.*
- By accepting this packet to current schedule, the energy budget is not exceeded (*checking energy* + *processing energy* + *considered(i).*$E_{x,y} \le E_{max}$).
- This packet is intentionally queried by end user ($IT_{min} \le$ *considered(i).*$I_{x,y} \le IT_{max}$).
- Among all the *PPs* that satisfy the above criteria, select the one that has the largest *considered(i).*$V_{x,y}$ = *considered(i).*$R_{x,y}$ / *considered(i).*$E_{x,y}$.
- By accepting this packet to current schedule, the summation of the system value is just up to the system_value defined in the query/event service APIs. ($\sum_{i \in PP}$ *selected(i).*$V_{x,y} \le$ *system_value*)

After choosing the *PP*, cluster head can send Acknowledge back to accept new packet. In addition, for those packets which end user is not interested in, their corresponded sensor nodes will discard them. In this case, we refuse and discard the un-

necessary data; consequently, we can reduce the energy consumption by reducing the data transmitting and receiving.

**Step 3: In STB**, **we transmit accepted packets to base station by using Velocity Monotonic Scheduling:**

As the algorithm that has been presented in [2], which assigns the priority of a packet based on its requested velocity. VMS minimizes the deadline miss ratios of sensor networks by giving higher priority to packets with higher requested velocities, which also reflects the local urgency. VMS embodies with both the timing constraint and location constraint.

**Another aspect: Replace or drop a packet in the STB**. A new packet is always accepted if possible. When receiving new *PP* from sensor node, if the STB is full, we can replace or drop a packet based on the following criteria:

- This packet's *PP* is selected (*selected(i).status = true*).
- Among all selected packet's *PPs*, find out the one that has the smallest *selected(i).*$V_{x,y}$ = *selected(i).*$R_{x,y}$ / *selected(i).*$E_{x,y}$.
- If this found one is not the new packet that is going to be accepted, we use this new packet to replace this found one, otherwise, we drop this new packet.

The flowchart and source code of ETRI-PF principles are showed in figure 2 and 3.

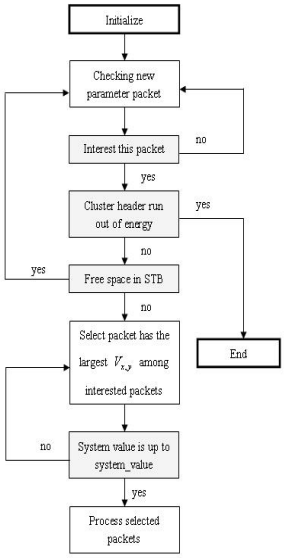


**Figure 2.** Flowchart of ETRI-PF

```
Step1  initialize: selected[i].status = false; considered[i].status = false;
       ∀i ∈{1,2,…,N}; checking_energy = 0; checking_time = 0;
       processing_energy = 0; processing_time = 0;
Step2  if(considered[i].interest_value >= interest_threshold) then
           considered[i].status = true;
           if(cluster header buff has free space) then
               receive_pp();
           else
               if(current_pp.ratio > min(pp.ratio in buff)) then
                   receive current_pp; discard min(pp.ratio in buff);
               else
                   discard current_pp;
               end if
           end if
       end if
       calculate_checking_time();
       calculate_checking_energy();
       for i to buff_size
           if(checking_time + processing_time > deadline) then
               stop_working();
           else
               if(total_ratio < system_value) then
                   if(checking_energy + processing_energy <= total_energy)
                       selected[i].status = true;
                   else
                       change selected[i] with
                       max(selected[i].processing_energy in select);
                   end if
               end if
           end if
       next
Step3  for i to buff_size
           if(selected[i].status = true) then
               receive_packet(); process_packet();
               calculate_processing_time(); calculate_processing_energy();
           end if
       Next
Step4  End
```

**Figure 3.** Pseudo code of ETRI-PF

## 5 Simulation Results

A sensor network can be modeled as a graph, where each vertex represents a sensor node and each edge represents the edge between two nodes when they are within each

other's communication range. This network tracks the values of certain variables like temperature, humidity, etc. Application users submit their requests as queries and the sensor network transmits the requested data to the application.

For the simulation work, we randomly deploy eleven different sensor nodes. And we randomly initialize these sensor nodes with: the total energy of sensor nodes (scope: from 111 to 888), the buffer size of sensor nodes (scope: from 6 to 9). Ten of these eleven sensor nodes are chosen to be the packet generators which randomly create these ten different packets and send to the remaining one. The remaining one works as the cluster head. For this cluster head, we design five parameters: the *total energy = 666*, the *buffer size = 6*, the *deadline = 5*, the *system_value = 10* and the *interest threshold = 5*. The meaning of threshold is that we just accept the packets when their interest value are larger than 5. Packets from those areas are what the end users are interested in.

In addition, we design ten different packets that are randomly initialized with the following four parameters: energy consumption (scope: from 3 to 10), processing time (scope: from 3 to 10), reward value (scope: from 3 to 10) and interest value (scope: from 3 to 10).

These ten sensor nodes are organized into three groups based on their created packets' interest values. The packets that have the interest values belong to {8, 9, 10} are considered as group A, the packets that have the interest values belong to {6, 7} are considered as group B, and the packet that have interest values belong to {3, 4, 5} are considered as group C. Suppose the cluster head just accepts the packets from area A and B, moreover, within these interested packets it accepts the packet that has the largest $V_{x,y} = R_{x,y} / E_{x,y}$ first. And we also design that this cluster head works in the STB by using the *Velocity Monotonic Scheduling.*

In terms of energy consumption, we mainly consider the following two parts that have strong relationship with our proposed ETRI-PF, which are *processing energy* $\{E_{(Returning\ ACK)} + E_{(Receiving\ packet)} + E_{(Processing)} + E_{(Broadcasting\ event)} + E_{(Listening)} + E_{(Accepting\ ACK)} + E_{(Sending\ packet)}\}$ and *checking energy* $\{E_{(Accepting\ event)} + E_{(Deciding)}\}$. The checking energy is designed to be 0.3, which is 10% of the minimum packet consumption 3; also the checking time is designed to be 0.3, which is 10% of the minimum processing time 3. Besides ETRI-PF, we provide two different existing packet scheduling algorithms to run on the cluster head for comparison as follows:

1) Compared Algorithm one (CA 1):
   a) In FTB: No *interest constraint* and no *reward constraint*
   b) In STB: Minimizing the packet deadline miss ratio (*Velocity Monotonic Scheduling*)

The cluster head doesn't set any threshold to reduce the incoming packets, but just simply receives packets and relays them. Once it gets a packet, it will process this packet based on the *Velocity* determined by *time constraint* and *location constraint*.

2) Compared Algorithm two (CA 2):
   a) In FTB: Consider *interest constraint*, but no *reward constraint*
   b) In STB: Minimizing the packet deadline miss radio (*Velocity Monotonic Scheduling*)

The cluster head always accepts the packet that has the interest value larger than the interest threshold. Once it gets a packet, it will process this packet based on the *Velocity* determined by *time constraint* and *location constraint*.

We used the following metrics to capture the performance of our routing approach and to compare it with other algorithms:

1) *total processing energy* of cluster head, 2) *energy utilization* of cluster head (*energy utilization = processing energy / (checking energy + processing energy)*), 3) *discarded packets ratio* in sensor nodes (*discarded packets number* / *total created packets number* by sensor nodes), 4) *total time consumption* of cluster head (*checking time + processing time*), 5) *average interest value* per packet, 6) *average reward value* per packet. The simulation results and comparisons are showed as the following figures.

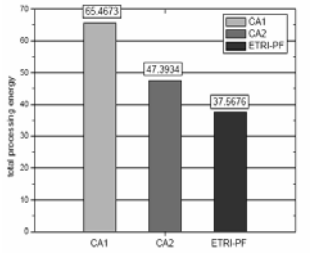


**Figure 4.** Total processing energy

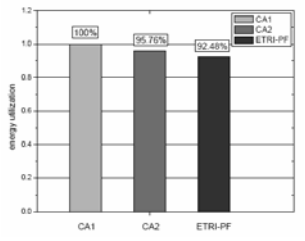


**Figure 5.** Energy utilization

From figure 4, we can find that algorithm CA1 costs a lot of *processing energy* and our ETRI-PF algorithm costs only about half of that. The reason is that the cluster header just simply receives the packets and relays them without reducing any incoming packets, neither interest nor reward constraint is considered in algorithm CA1. Take a look at figure 5, we find that the *energy utilization* (= *processing energy / (checking energy + processing energy)*) of our ETRI-PF algorithm is a little bit lower than the other two algorithms. Remember that we used both interest and reward constraints, which would definitely cost some checking energy, however, we still reduce the energy consumption of whole sensor networks. The saved energy comes from the normal sensor nodes but not from the cluster head.

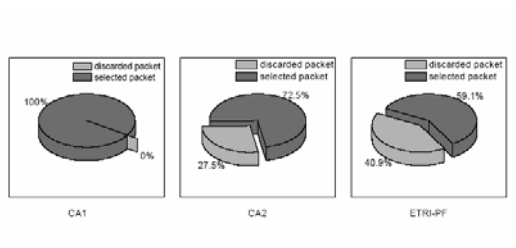


**Figure 6.** Discarded packet ratio

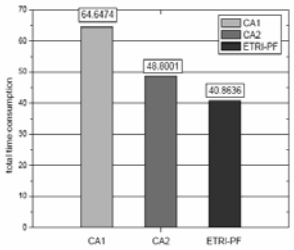


**Figure 7.** Total time consumption

Same conclusion can also be drawn form figure 6, by analyzing the *discarding ratio* (*discarding ratio = discarded packets / total created packets*), we can see that the

discarding ratio of our ETRI-PF is much higher than others. The lower *discarding ratio* the sensor nodes have, the more uninterested packets the sensor nodes send. Thus, the more unnecessary energy is consumed. In conclusion, by using the ETRI-PF, the sensor nodes can reduce the unnecessary transmission of uninterested data to reduce the energy consumption.

Consequently we get figure 7 showing the *total time consumption*, even though we need more checking time, we reduce the total time consumption by processing only part of the packets. For this part, the packets have larger reward than that of the rest packets.

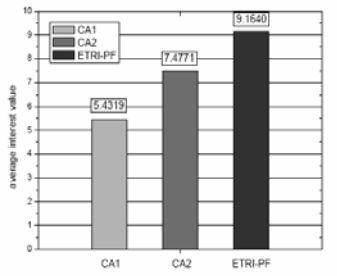


**Figure 8.** Average interest value

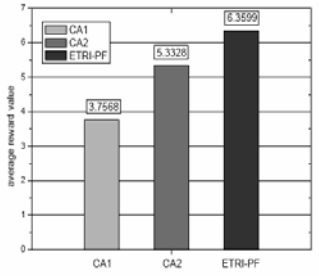


**Figure 9.** Average reward value

As we presented in foregoing paragraph, we design the *interest threshold* to accept packets that have larger interest values, therefore, the desired *average interest value* should be larger than that of other algorithms. Figure 8 shows that the *average interest value* of ETRI-PF is much larger than others, which means the ETRI-PF can exactly process the interested packets well. Figure 9 shows the comparison among three algorithms' *average reward values*. In the algorithm CA 1, because we do not intentionally maximize the *value ratio* ($V_{x,y} = R_{x,y} / E_{x,y}$), as a result, the *average reward value* of CA 1 is relatively smaller than others. Compared with CA 2, even though we add the *interest constraint* to CA 2, still no reward constraint is considered, thus the *average reward value*s of our ETRI-PF is the largest one. Once again, we demonstrate that our ETRI-QM can deal with the queries more efficiently and get more important information to solve the queries.

## 6 Conclusion and Future Work

Wireless sensor networks consist of nodes with the ability to measure, store, and process data, as well as to communicate wirelessly with nodes located in their wireless range. Users can issue queries over the network. Since the sensors have typically only a limited power supply, energy-efficient processing of the queries over the network is an important issue. In this paper, we proposed a novel query model ETRI-QM dynamically combining the four constraints: Energy, Time, Reward and Interest. By considering these four constraints simultaneously, we can maximize the system

value among the interested packet while satisfying the time and energy constraints by using our ETRI-PF algorithm. In this algorithm, we choose to process packets which have the highest reward value. A packet with a larger reward value means that this packet is more important. Based on our simulation results, we find out that our ETRI-QM and ETRI-PF algorithm can improve the quality of the information queried and also reduce the energy consumption.

However, as we mention the ETRI-QM principle that sensor nodes can know the reward value and interest value of packets well. In the simulation we randomly design the interest value and reward value for 10 different packets. But we do not mention the method that how to design the reward value and interest value for different packets based on each packet's content. Therefore, as a challenge issue to be solved in the future, we are going to explore the appropriate measure methods to evaluate the interest level and important level of different packets.

## Acknowledgement

This work was supported by grant No. R01-2005-000-10267-0 from Korea Science and Engineering Foundation in Ministry of Science and Technology.

## References

1. C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *MobiCOM*, Boston, MA, August 2000.
2. C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He, RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks, In IEEE RTAS 2002.
3. S. Madden and M. J. Franklin. Fjording the stream: An architechture for queries over streaming sensor data. In *ICDE*, 2002.
4. C. Rusu, R. Melhem, D. Mosse, "Maximizing Rewards for Real-Time Applications with Energy Constraints", ACM TECS, vol 2, no 4, 2003.
5. C. Rusu, R. Melhem, D. Mosse, "Multi-version Scheduling in Rechargeable Energy-aware Real-time Systems", to appear in Journal of Embedded Computing, 2004.
6. S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks. In *OSDI (to appear)*, 2002.
7. P.Bonnet, J.Gehrke, and P.Seshadri. Towards sensor database systems. In *Conference on Mobile Data Management*, January 2001.
8. Chien-Chung Shen, Chavalit Srisathapornphat, and Chaiporn Jaikaeo. Sensor information networking architecture and applications. IEEE Personal Communications, 8(4):52-59, August 2001.
9. O. Wolfson, A. P. Sistla, B. Xu, J. Zhou, and S. Chamberlain. DOMINO: Databases fOr MovINg Objects tracking. In *ACM SIGMOD*, Philadelphia, PA, June 1999.
10. Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. The Design of an Acquisitional Query Processor for Sensor Networks. SIGMOD, June 2003, San Diego, CA