Energy Join Quality Aware Real-time Query Scheduling Algorithm for Wireless Sensor Networks

Luong Thi Thu Phuong, Sungyoung Lee, Young-Koo Lee
Dept. of Computer Engineering, Kyung Hee University, Korea
e-mail : {luongthuphuong, sylee}@oslab.khu.ac.kr}; {yklee}@khu.ac.kr

Abstract

Nowadays, the researches study high rate and real-time query applications seem to be real-time query scheduling protocols and energy aware real time query protocols. Also the WSNs should provide the quality of data in real time query applications that is more and more popular for wireless sensor networks (WSNs). Thus we propose the quality of data function to merge into energy efficiency called energy join quality aware real-time query scheduling (EJQRTQ). Our work calculate the energy ratio that considers interference of queries, and then compute the expected quality of query and allocate slots to real-time preemptive query scheduler.

1. Introduction

Nowadays, real-time applications involve heterogeneous high data rate applications with different prioritizations that have to guarantee meeting their respective deadline. Furthermore, the real time applications are increasing demand for high performance query services not only quality of service but also quality of data. Moreover, the system needs to adapt variable workload in order to serve the diverse environment changes as well as the high priority emergent changes such as in case of stroke patients. For WSNs these is a very important requirement that supports the recent real time communications. However, another most important characteristic of WSNs has to provide the different quality query requirements while ensuring timing constraints and preemptive constraints that become a main challenge for real time applications in WSNs. So in this paper we introduce the energy join quality aware real time query scheduling framework (EJQRTQ) to address these problems for WSNs. This works on the basic real-time preemptive query scheduling [1] to analysis the energy ratio based on the quality requirement constraints. The energy ratio is the ratio of the useful energy consumption and total energy consumption with given quality of data whose requirements can be specified by users. Given the quality function and real time query scheduling policies, node estimates the energy ratio maximization for each processed query according to the quality it severed. We exploit the characteristics of query services including in-network data aggregation and periodic timing properties which are very important from a point of view of schedulability.

Related work

Recently, lots of real time query scheduling researches that most of them are considered under query service which supports in sensor models. RTQS [1] protocol is a TDMA scheduling that solves real-time query scheduling with energy efficiency but does not considers the quality of data for query as well as DCQS [2]. Alternatively, there are few studies on quality of service and quality of data in query scheduling but they don't compute for real time preemptive query scheduling and interference of queries together [3]. EJQRTQ has both two improvements: one is study of energy and quality of data (QoD) in real-time preemptive query scheduling, and other is provision of approximation approach to reduce interference energy consumption that is brought from higher priority queries to lower priority queries. Previous works [3], [4], [5] for both computation and communication are close to our work.

Our work bases on aggregated query services in TinyDB [6] that presents a query service that allows aggregation query to collect information of the entire network or a group of nodes through a routing tree. We will use aggregation characteristic to analysis the network behavior in such energy consumption.

2. System model

2.1 Network model

In this paper, we assume a WSNs model as an Interference Communication (IC) graph [1]. IC (E,V) has all nodes as vertices and has two types of directed edges: *communication* $\overrightarrow{\boldsymbol{ab}}$ *edge and interference* $\overrightarrow{\boldsymbol{cd}}$ *edge* which are already defined in [1]. With this, two transmissions are conflict-free ($\overrightarrow{\boldsymbol{ab}} \,||\, \overrightarrow{\boldsymbol{cd}}$) then they can be scheduled concurrently if firstly, *a, b, c, d* are distinct and secondly, $\overrightarrow{\boldsymbol{ab}}$ and $\overrightarrow{\boldsymbol{cd}}$ are not communication or interference edges in E. In this model, the clocks are assumed that they are synchronized. To construct the IC graph is described in [7].

2.2 Query model

In this paper, we refer a common query model in which source nodes generate periodically data reports. Let us assume that a query is characterized by the following parameters: a set of sources that respond to a query, a function for in-network aggregation [3], the start time $\boldsymbol{\Phi_l}$, the query period $\boldsymbol{P_l}$, the query deadline $\boldsymbol{D_l}$ and a static priority. As we said above that the query is characterized by temporal properties therefore in the beginning of the each period of a query there is always a query instance is released. We refer $\boldsymbol{I_{l,i}}$ is a $\boldsymbol{i^{th}}$ query instance of query *l*. Priority of query instance has the same priority of its query. Two query instances have the same priority, the higher priority are made by the earliest release time. With a query instance $\boldsymbol{I_{l,i}}$ a node *n* needs $\boldsymbol{m_{l,n}}$ slots to transmit its (aggregated) data report to its parent. Here $\boldsymbol{m_{l,n}}$ can be considered as a pay load of query instance $\boldsymbol{I_{l,i}}$.

Currently, query service constructs the routing tree that has root located at base station. In-network query processing systems, we use only one kind of aggregation query.

2.3 Power model

Power modes in a node are categorized as follows:
Active: node evaluates the query: $\boldsymbol{P_{active}}$.
Sleep: node radio is turned off that consumes $\boldsymbol{P_{sleep}}$.
Transmit/receive: power consumes to transmit or receive one packet: $\boldsymbol{P_{trans/packet}}$ and $\boldsymbol{P_{recv/packet}}$.

2.4 EJQRTQ framework overview

Before going to more detail the EJQRTQ algorithm, we show the EJQRTQ framework overview in Fig 1. This algorithm is performed on each local node and it is adopted the existing real-time query scheduling protocol [1]. EIQRTQ calculates the expected quality for each query based on computing energy ratio maximization, and then determining the number of transmitting and receiving slots need for scheduling. We will explain obviously in latter parts.

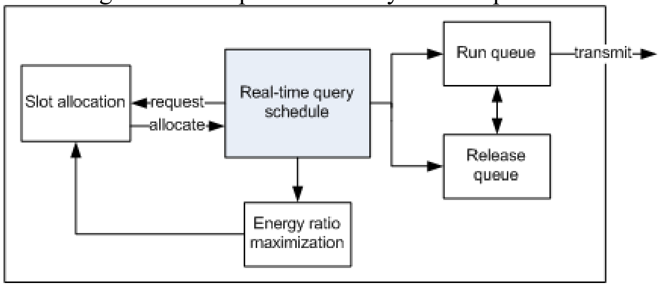

Figure 1. EIQRTQ framework overview

3. Problem statement

3.1 Quality of data (QoD)

QoD is defined as follow:

$$QoD = \frac{QB}{QN} - \frac{1}{L}\rho\sum_{i=1}^{L}\left(\frac{QB}{QN} - \frac{QB(i)}{QN(i)}\right) \qquad (1)$$

In equation (1), QB and QN are number of queries received at base station (root node of WSNs), and number of queries generated in whole WSN. In real-time query scheduling [1] constructs the planner that plans all steps to assign entire data report transmissions into each node. L is the maximum length of the plan of query that injected to network [1]. $QB(i)$ and $QN(i)$ are number of transmitted queries in step i and number of all responding queries that should be generated in step i. The ratio $\frac{QB}{QN}$ individual can not describe the quality of a network well. The network wants to return responding queries from nodes that are closer to the root node as much as possible to save energy. In this case, we add a weighted difference between $\frac{QB}{QN}$ and $\frac{QB(i)}{QN(i)}$ of each step i as a punishment. The ρ $(0 < \rho \leq 1)$ is defined by user. ρ is larger if user expects received source data to be more evenly distributed among steps.

3.2 Energy ratio problem

As we said above, the energy ratio is the ratio of the useful energy consumption and total energy consumption. The total energy consumption is the sum of the useful energy consumption and the interference energy consumption.

In the real-time query scheduling that supports dynamic preemptive scheduling can make the higher priority query interfere to lower priority query. Under that, because of interference, a higher priority query can steal from a lower priority query at most $\left\lceil\frac{R_l}{P_h}\right\rceil \times \min(2\Delta, L)$ slots are deduced in [1]. In there, R_l is the worst-case delay of the query l by higher priority query h. The inverse of query period $\frac{1}{P_h}$ is the rate of query h and the Δ is minimum step distance between two queries such that two steps of these two queries may be executed concurrently without conflict [1]. Thus, the worst-case interference of a query l by all of its higher priority queries is: $\sum_{h\in hp(l)}\left\lceil\frac{R_l}{P_h}\right\rceil \times \min(2\Delta, L)$ slots where $hp(l)$ is the set of queries with higher priority than or equal to l's priority. So energy consumption of query l has to consider the interference that is called interference energy consumption $E_{interfer}$. We try to reduce the interference energy consumption by punishing it. Hence, we define a punishment to $E_{interfer}$ by linear function of QoD of queries that interfere to current processed query. Additionally, since punishing the interference energy consumption that means we can reduce the worst-case delay of current processed query to increase the query successfulness probability that can meet its deadline as well as increasing the reliability of network. The problem for EJQRTQ is to find the maximum energy ratio of the queries processed. After that, we have the energy ratio formula as follow:

$$R = \frac{E_{useful}}{E_{useful}+E_{interfer}-(\alpha QoD+\beta)} \qquad (2)$$

3.3 Energy consumption

The useful energy consumes on a node for a current processed query l within T length of time can be evaluated as $E_{useful}(l) = P.T$ [5]. Now the issue is to consider energy in the different operations of query instance execution, the energy consumption should be computed as: $E_{useful}(l) = Epoch \times \sum T_j \times P_j$ where T_j is the execution time of each type of j-th operation per epoch and P_j is the corresponding power. $Epoch$ is total number of epochs that need to process the query. With T_j, we consider four execution states of operator include: (1) $T_{evaluate}$ is the query evaluation time. (2) T_{recv} is time to receive all the packets from its children. (3) T_{trans} to transmit all the packets by node n to its parent. (4) T_{sleep} is time that node n is in sleep mode.

The power consumption composes of $P_{evaluate}$, P_{trans}, P_{recv}, and P_{sleep} according to the each time T_j on a node n for a single query l. Firstly, the evaluation power consumption $P_{evaluate}$ that to evaluate the query is computed in active power mode P_{active}. Secondly:

$$P_{trans} = \frac{1}{P_l}.P_{trans/packet}.M_{l,n} \quad (3)$$

And

$$P_{recv} = \frac{1}{P_l}.P_{recv/packet}.\sum_{c\in child(n)} M_{l,c} \quad (4)$$

where $\frac{1}{P_l}$ is the rate of query l. $M_{l,c}$ is the maximum number of received packets on node n by all its child c to satisfy the workload demand of query l and $M_{l,n}$ is the maximum number of query packets transmitted by node n to its parent.

With above analysis, we have the equation of useful energy consumption as:

$$E_{useful}(l) = Epoch \times (T_{evaluate} \times P_{active} + T_{recv} \times P_{recv} + T_{trans} \times P_{trans} + T_{sleep} \times P_{sleep}) \quad (5)$$

We need to calculate each execution time of j-th operator per epoch. The time $T_{evaluate}$, T_{rec}, T_{trans} and T_{sleep} are deduced in [3] as follows:

$$T_{evaluate} = QoD \times \sum_{j=1\dots n} T(j) - T_0 \quad (6)$$

$$T_{recv} = QoD \times N_{child} \times t_s \quad (7)$$

$$T_{comm} = (N_{child} + 1)t_s \quad (8)$$

$$T_{trans} = T_{comm} - T_{rec} = (N_{child} - QoD \times N_{child} + 1) \times t_s \quad (9)$$

$$T_{sleep} = T - (T_{evaluate} + T_{rec} + T_{trans}) = T - (T_{evaluate} + T_{comm}) \quad (9)$$

where QoD is computed in equal (1) that is quality of data can be given on each node, $T(j)$is the execution time of the j-th operator and T_0 is the overlapping time between the query execution and the communication on a node, N_{child} is an average number of children nodes, t_s is a slot time duration, T_{comm} is the communication time. The operator execution time can be measured offline

Given the above types of execution time, we get the useful energy consumption can be expressed as:

$$E_{useful}(l) = Epoch \times (a_1 \times QoD + a_2) \quad (10)$$

With:

$$a_1 = (P_{active} - P_{sleep}) \times \sum_{j=1\dots n} T(j) - P_{trans} \times t_s \times N_{child} + P_{recv} \times t_s \times N_{child} \quad (11)$$

$$a_2 = (P_{active} + P_{sleep}) \times T_0 + P_{trans} \times t_s \times N_{child} + P_{recv} \times t_s \times N_{child} + P_{sleep} \times T - P_{sleep} \times t_s \times (N_{child} + 1) \quad (13)$$

As we mentioned above, the $E_{interfer}$ is expressed in the following equation:

$$E_{interfer} = \sum_{h\in hp(l)} \left\lceil \frac{R_l}{P_h} \right\rceil . \min(2\Delta, L) \times P_{active} \times t_s \times Epoch \quad (14)$$

where $R_l = \Delta + \sum_{h\in hp(l)} \left\lceil \frac{R_l}{P_h} \right\rceil . \min(2\Delta, L) \quad (15)$

After finding the min($2\Delta, L$) worst-case interference of set of higher priority queries $hp(l)$ on query l, R_l can be computed by solving (15) using a fixed algorithm similar to the one used in the response time analysis [8]. With above equation, we can refer to a reduced equation as follow:

$$E_{interfer} = a_3 \times Epoch \quad (16)$$

With $a_3 = \sum_{h\in hp(l)} \left\lceil \frac{R_l}{P_h} \right\rceil . \min(2\Delta, L) \times P_{active} \times t_s \quad (17)$

3.4 Energy ratio

With all the $E_{useful}(l)$ equation, $E_{interfer}$ equation and quality function, the energy ratio R can be calculated by the following equation:

$$R = \frac{Epoch \times (a_1 \times QoD + a_2)}{Epoch \times (a_1 \times QoD + a_2) + Epoch \times a_3 - (\alpha QoD + \beta)} \quad (18)$$

Thus, the energy ratio problem turns into searching for a pair of $(QoD, Epoch)$ such that R is maximized. In essence, this problem is calculating extreme value of two bivariate functions. And we compute the partial derivatives to solve this problem. The method that we use is close to the *Profit* maximization problem in [3] but they did not treat the interference energy consumption thoroughly as we do.

The partial derivatives of energy efficiency in the direction of QoD and $Epoch$ as shown in the following equations:

$$\frac{\partial R}{\partial QoD} = \frac{a_1 a_3 Epoch^2 - (a_1\beta + a_2\alpha)Epoch}{[(Epoch \times (a_1 \times QoD + a_2) + Epoch \times a_3 - (\alpha QoD + \beta))]^2} \quad (19)$$

$$\frac{\partial R}{\partial Epoch} = \frac{-a_1\alpha QoD^2 - a_1\beta QoD + a_2\alpha QoD + a_2\beta}{[(Epoch \times (a_1 \times QoD + a_2) + Epoch \times a_3 - (\alpha QoD + \beta))]^2} \quad (20)$$

Hence, it is clear that $\frac{\partial R}{\partial QoD}$ =0 and $\frac{\partial R}{\partial Epoch}$ =0 are the necessary conditions for $(QoD, Epoch)$ to be extreme value point of R. Then, we calculate the second derivatives of energy ratio: $\frac{\partial^2 R}{\partial QoD^2} = X_1$, $\frac{\partial^2 R}{\partial QoD \partial Epoch} = X_2$, and $\frac{\partial^2 R}{\partial Epoch^2} = X_3$. EJQRTQ can find out the pair values $(QoD_m, Epoch_m)$, that enables the energy ratio $R_m(l)$ of each query l to be a maximum one by using the following mathematical method: (1st) compute the pair $(QoD_m, Epoch_m)$ that satisfy the two conditions $\frac{\partial R}{\partial QoD}$ =0 and $\frac{\partial R}{\partial Epoch}$ =0. (2nd) if QoD_m, $Epoch_m$ satisfy the threshold quality of query that is predefined before by user, then calculate the second derivatives at these values, if not jump to (4). (3rd) Calculate the checking condition $X = X_1X_3 - {X_2}^2$ whether $(QoD_m, Epoch_m)$ is the maximum point, if $X < 0$, then $(QoD_m, Epoch_m)$ else jump (4). (4th) if we cannot find the maximum point of energy ration function or may be it exceeds the quality threshold, then solution for pair $(QoD_m, Epoch_m)$ is set them as lowest or highest possible quality. (5th) after all, EJQRTQ computes the target quality for the each query.

3.5 The proposed EJQRTQ algorithm :

Input: Set of different priority queries l_k (*k=1...K)* with respective quality functions. The real-time preemptive query scheduling is done previously.

Output: Target quality for each query, predictable slot allocation

```
1: For k=1 to K do
   {Compute $\frac{\partial R[k]}{\partial QoD[k]}$=0 ; $\frac{\partial R[k]}{\partial Epoch[k]}$=0;
   $\frac{\partial^2 R}{\partial QoD^2} = X_1$; $\frac{\partial^2 R}{\partial QoD\partial Epoch} = X_2$; $\frac{\partial^2 R}{\partial Epoch^2} = X_3$;}
2: If ( $X = X_1X_3 - {X_2}^2$ <0 ) and ( $QoD_m[k], Epoch_m[k]$ )< threshold Then
3:    $slot_{recv}[k] = Q_{recv}[k] \times QoD_m[k]$;
4:    $slot_{trans}[k] = Q_{trans}[k] \times QoD_m[k]$;
5: Else {Maximum ratio point is not found or exceeds the quality threshold;
        Set $(QoD_m, Epoch_m)$ = lowest or highest possible quality}
End
6: Send the $slot_{recv}[k]$ and $slot_{trans}[k]$ of query $k$ to underlying real time query scheduling protocol to adjust the scheduler.
```

In this EJQRTQ algorithm, line (1-2) uses the first and second derivatives of energy ratio to get the expected quality $(QoD_m[k], Epoch_m[k])$ for *k-th* query according to the conditions that are presented previously. Line (3-4) compute the number of receiving slots needed $slot_{recv}[k]$ and number of transmitting slots needed $slot_{trans}[k]$ for query k by using the $QoD_m[k]$.The $Q_{recv}[k]$, $Q_{trans}[k]$ are number of receiving and transmitting slots if $QoD_m[k] = 1$. Line 5 returns the lowest or highest possible quality if previous conditions can't be satisfied. Line 6 adjusting the scheduler with the known receiving and transmitting slots that the node needs for each query to get the maximum efficient slots allocation in real-time query scheduling. Furthermore, we can sort the order of queries according to their expected quality value that we found to provide for the real-time query scheduling algorithm make the decisions.

4. Experiment

We evaluate the finding ability the maximum point of energy ratio with different network parameters. By the way, we can get the target quality for each query and sort the order of them to adjust scheduling efficiently.

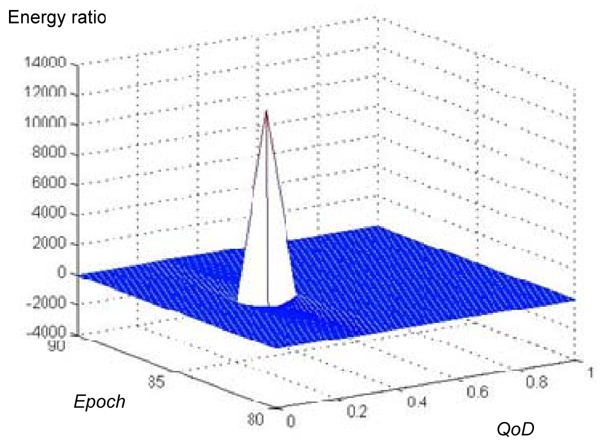

Figure 2. Energy ratio and $(QoD_m[k], Epoch_m[k])$

We simulate with assumption is the network has 100 sensor nodes, with set of three queries that have rate at 4.1Hz, 4.2Hz and 4.3 Hz, respectively. Based on the real time preemptive query scheduling in [1] with total time is 200s. the quality function is as follow: $\alpha = 167.67$, $\beta = -66.67$. We prove that exist the maximum point of energy ratio according to equation we proposed. In this experiment, we can realize the expected quality for a processed query at energy ratio maximum is near 0.4 and $Epoch_m[k]$ approximates 85s.

Due to the lack of real systems as well as real sensor motes, we have not implemented this algorithm to show the advantage results. We will take the experiments on the future work based on this strong analysis.

5. Conclusion

In this paper, we presented the energy join quality aware real time query scheduling algorithm in a local node that can get the expected quality of query as well as efficient slots allocation (ensuring reserved bandwidth that satisfy the one of resource constraints in this paper) while ensuring run on real-time query scheduling fundamental. This algorithm can perform on each local node instead of on base station to reduce the communication overhead. Then we can save more costly. The EJQRTQ algorithm based on the existing real time query scheduling for WSNs with key improvement is a *expected quality* that enables the users to specify their quality requirements by using quality functions and also exploit the interference energy to punish it.

References

[1] O. Chipara, C. Lu, and G. -C. Roman, "Real-time query scheduling for Wireless sensor networks" In RTSS '07.

[2] O. Chipara, C. Lu, and J. A. Stankovich, "Dynamic conflict free query scheduling for wireless sensor networks", In ICNP, 2006.

[3] H. Wu, Q. Luo, J. Li, and A. Labrinidis, "Quality aware query scheduling in wireless sensor networks", In DMSN '09 Proceedings of the Sixth International Workshop on Data Management for Sensor Networks 2009.

[4] Q. Cao, T. F. Abdelzaher, T. He, and J. A. Stankovic, "Towards optimal sleep scheduling in sensor networks for rera-event detection", In IPSN, 2005.

[5] H. Qu and H. Labrinidis, "Preference-aware query and update scheduling in web-databases", In ICDE, 2007, pp. 356-365.

[6] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong., "TAG: a tiny aggregation service for ad hoc sensor networks", In OSDI,2002.

[7] G. Zhou, T. He, J. A. Stankovic, and T. F. Abdelzaher, "RID: radio interference detection in wireless sensor networks", In INFOCOM, 2005.

[8] A. N. Audsley, A. Burns, M. Richardson, and K. Tindell, "Applying new scheduling theory to static priority preemptive scheduling", *Software Engineering Journal,* 1993 01088859789.