Mobile Agent Based Wireless Sensor Networks

Min Chen[†], Taekyoung Kwon[*], Yong Yuan[+], and Victor C.M. Leung[†]
[†]Dept. of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada
Email: {minchen, vleung}@ece.ubc.ca
[+] Department of Electronics and Information, Huazhong University of Science and Technology
Email: yy_hust@hotmail.com
[*]School of Computer Science and Engineering, Seoul National University, Seoul, 151-744, Korea
Email: tkkwon@snu.ac.kr

***Abstract*—Recently, mobile agents have been proposed for efficient data dissemination in sensor networks. In the traditional client/server-based computing architecture, data at multiple sources are transferred to a destination; whereas in the mobile-agent based computing paradigm, a task-specific executable code traverses the relevant sources to gather data. Mobile agents can be used to greatly reduce the communication cost, especially over low bandwidth links, by moving the processing function to the data rather than bringing the data to a central processor. This paper proposes to use the mobile agent paradigm for reducing and aggregating data in a planar sensor network architecture. The proposed architecture is called mobile agent based wireless sensor network (MAWSN). Extensive simulation shows that MAWSN exhibits better performance than client/server communications in terms of energy consumption and the packet delivery ratio. However, MAWSN has a longer end-to-end latency than client/server communications in certain conditions.**

***Index Terms*—mobile agent, energy efficient, aggregate, data dissemination, wireless sensor networks**

I. INTRODUCTION

Recent years have witnessed a growing interest in deploying large numbers of micro-sensors that collaborate in a distributed manner on data gathering and processing. Sensors are expected to be inexpensive and can be deployed in a large scale in harsh environments, which implies that sensors are typically operating unattended. Energy-efficient data delivery is crucial because sensor nodes operate with limited battery power. Currently, most energy-efficient proposals [1] in wireless sensor network (WSN) are based on the client/server computing model, where each sensor node sends its sensory data to a back-end processing center or a sink node. Because the link bandwidth of a WSN is typically much lower than that of a wired network, a sensor network's data traffic may exceed the network capacity.

To solve the problem of the overwhelming data traffic, Qi. et al [3] proposed the mobile agent based distributed sensor network (MADSN) for scalable and energy-efficient data aggregation. By transmitting the software code, called a mobile agent (MA), to sensor nodes, a large amount of sensory data can be reduced or transformed into a small amount of data by eliminating the redundancy. However, the operation of an MADSN is based on the following assumptions: (1) the sensor network architecture is clustering based; (2) each source node is within one hop from a clusterhead; (3) much redundancy exists among the sensory data which can be fused into a single data packet with a fixed size. These assumptions pose much limitation on the range of applications that can be supported by an MADSN. Thus, we will consider MA in multi-hop environments with the absence of a clusterhead. In this paper, a MA is exploited in two levels to reduce the information redundancy in a planar WSN. Specifically, the MA is proposed to perform the following functions: (1) eliminating data redundancy among sensors by application context-aware local processing at the node level; (2) eliminating spatial redundancy among closely-located sensors by data aggregation at the task level; (3) reducing communication overhead by concatenating data at the combined task level. The proposed architecture is called mobile agent based wireless sensor network (MAWSN).

Extensive simulation-based comparisons between MAWSN and client/server based WSN (CSWSN) show that, depending on the parameters, MAWSN can significantly reduce the energy consumption while conditionally improving the end-to-end delay.

The rest of this paper is organized as follows. Section II presents related work. We describe the MAWSN architecture and design issues in Sections III. Simulation model and experiment results are presented in Sections IV and V, respectively. Section VI concludes the paper.

II. RELATED WORK

Recently, MAs have been proposed for efficient data dissemination in WSNs [3-8]. In a typical client/server based WSN, the occurrence of certain events will alert sensors to collect data and send them to a sink node. However, the use of MAs leads to a new computing paradigm, which is in marked contrast to the traditional client/server-based computing. The MA is a special kind of software that propagates over the network either periodically or on demand (when required by the applications). It performs data processing autonomously while migrating from node to node. Q. Wu et. al. [5]

presents a genetic algorithm based solution to compute an approximation to the optimal source-visiting sequence. The use of MAs in computer networks has certain advantages and disadvantages [10], such as code caching, safety and security, depending on the particular scenario. Regardless, they have been successful deployed in many applications ranging from e-commerce to military situation awareness [7]. As described in [3], many inherent advantages (e.g., scalability, extensibility, energy awareness, reliability) of the MA architecture make it more suitable for WNSs than the client/server architecture. In [4], MAs are found to be particularly useful for data fusion tasks in distributed WSNs.

In our previous work [2], we only presented a description of data dissemination using MA in a planar WSN, where MAs are exploited at three levels (i.e., node level, task level, and combined task level). We extend that work in this paper by proposing a scheme for MA migrating in MAWSN. Then, we verify the efficacy of MAWSN by extensive simulations.

III. System Architecture And Design of MAWSN

In this section, we present the architecture and design of the MAWSN. We first give an overview of the network organization, and then describe MA assisted information redundancy reduction at three levels. Lastly, we present the operation of MAWSN in detail.

A. Overview

In the architecture illustrated in Fig. 1, a sink queries multiple targets simultaneously by means of the MA. The data in the target region is collected from the targets one by one. The operation of the basic MAWSN will be described in detail in Section III.C.

In traditional scenarios, multiple requests for different physical information arrive at different times. We believe that applications that require multiple different requested tasks to be executed concurrently will become widespread in the future. The reaction to a single task can range from the simple return of a result by collaborative processing among some sensor nodes (e.g., in the application to obtain the population of the objects, the system aggregates reports of individual objects right at the point of data source and sends the already-aggregated object counts), to a complex return of a large volume of sensed data (e.g,. a picture captured by an image sensor). Due to protocol overheads, the communication cost of sending a longer message is usually less than sending the same amount of data using many short messages. For the concurrent tasks associated with small amounts of data, we can perform them by a single packet carrying multiple requests (e.g., one request for each task) and also concatenate their results into a single packet to save communication overhead. In Fig. 1, the combined multiple tasks will be executed one by one, so that the overall processing will take a longer time. If the application's minimum quality of service requirement (e.g., latency bound) is not violated, especially in the case that the target region is far from the sink node, the energy savings of this combined execution can be significant. We believe a combined-task query model that enables applications to initiate multiple tasks is needed to support this new and growing class of applications for WSNs. Basically, the combined-task has three features: (1) all the tasks belonging to a combined-task can be processed by the common processing code part of the MA packet, thus extra communication overhead is not needed for the MA to carry additional processing code; (2) compared with the distance between the center of the combined-task region and the sink, the task regions are likely to be close to each other; (3) all the tasks are requested concurrently by the application.

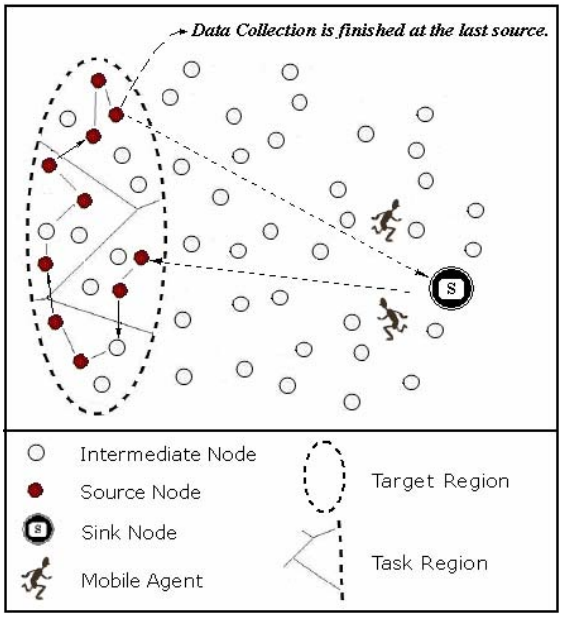

Figure 1. The Architecture of MAWSN

B. Information Redundancy and Communication Overhead Reducing

The proposed dissemination framework employs the MA's ability to carry processing codes that allow the computation and communication resources at the sensor nodes to be efficiently harnessed in an application specific fashion. The MAs should adjust their behaviors depending on quality of service needs (e.g. data delivery latency) and the network characteristics to increase network lifetime while still meeting those quality of service needs. In this section, we discuss how an MA may be dedicated to reduce the information redundancy and communication overhead in three levels so as to prolong network lifetime.

(1) Node Level: Elimination of Application Redundancy by MA Assisted Local Processing

With the development of WSN, "one deployment, multiple applications" is a trend due to the application-specific nature of sensor networks. Such a trend requires sensor nodes to have various capabilities to handle multiple applications, which is economically difficult to achieve, if not infeasible. In general, due to memory-constraints, it is impossible to store every possible application in the local memories of embedded sensors.

Thus, a way of dynamically deploying a new application is needed. In Fig. 1, the sink can assign the processing code (behavior) of an MA based on the requirement of a specific application. The processing code carried by the MA packet only requires local processing of the raw data at the source nodes as requested by the application. This capability enables a reduction in the amount of data transmission by allowing only relevant information to be extracted and transmitted.

Let r, ($0 < r < 1$), be the data reduction ratio contributed by the MA assisted local processing, S_i be the size of raw data at source i, and R_i be the size of reduced data. Then,

$$R_i = r \cdot S_i \ . \qquad (1)$$

(2) Task Level: Spatial Redundancy Elimination by MA Assisted Data Aggregation

The degree of sensed data correlation among sensors is closely related to the distance between sensors, so that it is very likely for closely-located sensors to generate redundant sensed data. The MA aggregates individual sensed data when it visits each target source. Though this kind of aggregation technique is typically used in clustering or aggregation tree based data dissemination protocols, the MA assisted aggregation does not need any overhead to construct these special structures.

We calculate the size of sensed data accumulated by the MA using the similar method as [6]. According to [6], a sequence of sensed data can be combined with a fusion factor ρ . Let N_i be the amount of sensed data accumulated after the MA collects the result of source i. Recall that R_i is the size of locally processed sensory data which will be accumulated by the MA at source i. Note that data aggregation only begins from the second source, then we have,

$$\begin{aligned} N_1 &= R_1, \\ N_i &= R_1 + \sum_{k=2}^{i} \rho \cdot R_k, \quad (i \geq 2) \end{aligned} \qquad (2)$$

(3) Combined Task Level: Communication Overhead Saved by MA Assisted Concatenation of Data from Multiple Tasks

Instead of MA assisted aggregation of data from individual sensor nodes right at data sources, we propose a packet unification technique that concatenates the data from several short packets into one longer packet in order to reduce the communication overhead at the combined-task level. Due to data concatenation, the duty cycle and communication overhead of intermediate sensor nodes can be reduced so as to increase network lifetime. However, such energy savings can be achieved usually at the cost of the prolonged data latency.

Let ID_j be the identifier of the last source at which the MA has finished task j. Let C_j be the amount of concatenated data that result after the MA leaves source j. Let $N(j)$ be the total amount of data the MA has collected for task j. Then C_j is equal to:

$$C_j = \sum_{m=1}^{j} N(m) \qquad (3)$$

C. The Operation of MAWSN

Before describing the operation of MAWSN, we first present our assumptions made for the MAWSN design and its application context.

(1) The sink knows the set of source nodes which will be visited by the MA.
(2) The itinerary of the MA is already setup before the sink dispatches it.
(3) A source or sink's flooding control message will cause other sensor nodes to set up a gradient to it. Assuming symmetrical channels, the sensor nodes can route packet to the source or sink by means of gradient. The detailed routing mechanism is beyond the scope of this paper, interest readers please refer to one phase-pull Directed Diffusion [11].

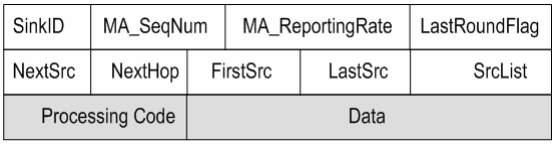

Figure 2. MA Packet Structure

The information contained in an MA packet is shown in Fig. 2. The pair of *SinkID* and *MA_SeqNum* is used to identify an MA packet. Whenever a sink dispatches a new MA packet, it will increment the *MA_SeqNum*. The list of sources (*SourceList*) specifies which sources will be visited by the MA. In *SourceList*, there are two special sources, namely, the first source (*FirstSource*) and the last source (*LastSource*) that the MA will visit. The pair of *FirstSource* and *LastSource* indicates the beginning and end points of the MA's data gathering. *NextSource* specifies the next destination source node to be visited. *NextHop* indicates the immediate next hop node which is an intermediate sensor node or a target source node. If *NextHop* is equal to *NextSource*, it means that the next hop node is the current destination source.

We define *round* as the period from the instance an MA collects a data packet in *FristSource* to the instance it collects a data packet in *LastSource*. *LastRoundFlag* indicates the current round is the last round of the whole task. The flag is set by *FirstSource*. After the data collection of the last round is done, the task will be finished.

The payload of an MA packet includes two kinds of data. One is *ProcessingCode* which is used to process sensed data; the other is *Data* which carries the aggregated data. If the MA arrives at *FirstSource*, it will be stored temporarily. To initiates a new round of data collection repeatedly, *FirstSource* will create a new MA by copying from the stored one with the frequency of *MA_ReportingRate*. *MA_ReportingRate* is equal to the desired data rate in a CSWSN.

```
/* Check whether an MA arrives
            at a specific source or not */
If (ThisNode=FirstSource)
    MA migrates toward FirstSource;
elseif (ThisNode=NextSource)
       &&(NextSource!=LastSource)
    MA collects sensed data;
    Among the sources in SourceList,
          select the one with maximum
                gradient as NextSource;
    Set NextSource in the MA packet;
    MA migrates towards NextSource.
elseif (ThisNode=LastSource)
    MA collects sensed data;
    MA migrates back to sink;
elseif ( NextSource=FirstSource)
    MA migrates between source nodes.
endif
```

Figure 3. Mobile Agent Migrating Algorithm

Fig. 3 explains how an MA migrates in a round. When the MA arrives at a sensor node, it looks at the identifier of the current node (*ThisNode* in Fig. 3) to decide whether or not it has arrived at the destination source or not. If not, the MA continues migrating toward the specific source. Otherwise, it operates as follows: (1) collect the locally processed sensed data; (2) delete the identifier of current target source from *SourceList*; (3) choose *NextSource* as the next destination. If the current node is *LastSource*, the MA will return to the sink.

IV. Simulation Methodology

Table 1. Basic Parameters configuration

Network Scale & Source Specification	
Network Size	300 m × 300m
Maximum Transmission Range	15m
Topology Configuration Mode	Randomized
Total Sensor Node Number	600
Number of Target Sources (n_{source})	Default: 4
Data Rate at MAC layer	2Mbps
Sensed Traffic Specification	
Size of Sensed Data (S_{data})	Default: 4Kbytes
Sensed Data Interval	1s
Task Duration (T_{task})	Default: 100 minutes
MA Specification	
Fusion Factor (ρ)	1
Reduction Ratio (r)	Default: 10%
Local Processing Time at Target Source (T_{proc})	Default: 50ms
Size of Processing Code (S_{proc})	Default: 0.4Kbytes

In order to demonstrate the performance of MAWSN, we choose a client/server based scheme (e.g. DD [11]) to compare with MAWSN. We use OPNET [12, 13] for discrete event simulation. The sensor nodes are battery-operated except the sink. The sink is assumed to have an infinite energy supply. We assume both the sink and sensor nodes are stationary. Each source is expected to generate sensed data packets with 1 minute interval on average. We use the energy model in [14]. The parameter values used in the simulations are presented in Table 1. The basic settings are common to all the experiments.

There are four performance metrics evaluated:

(1) *Energy Consumption* – We use e_{cs} and e_{ma} to denote the energy consumption in the CSWSN and MAWSN, respectively. For both schemes, this includes the total energy consumption due to transmitting, receiving, and overhearing during each simulation. In the MAWSN, the energy consumption for local processing at the target sensor node is additionally included. We disregard the energy consumption of processing in the CSWSN since most processing is done in the sink that has a sufficient energy supply.

(2) *Average End-to-end Packet delay* – We use t_{cs} and t_{ma} to denote the average end-to-end delay in the CSWSN and MAWSN, respectively. t_{cs} and t_{ma} include all possible delays during data dissemination, caused by queuing, retransmission due to multiple access collisions, and packet transmission time. In the MAWSN, t_{ma} is the average time interval between the time an MA is created and the time the MA returns to the sink. When the MA visits each source, t_{ma} is incremented by the time value of T_{proc} in Table 1. T_{proc} includes the delay of the MA accessing a target source node and the latency of local processing.

Let t_1 be the time interval between the time the sink dispatches an MA and the time the MA arrives at *FirstSource*, and let t_i be the time interval between the time the *FirstSource* dispatches an MA and the time the MA returns to the sink in round *i*. Let *R* be the number of rounds, where:

$$R = \frac{T_{task}}{SensedDataInterval} \qquad (4)$$

Then, t_{ma} is give by:

$$t_{ma} = \frac{t_1 + \sum_{i=1}^{R} t_i}{R} \qquad (5)$$

If T_{task} is large enough, t_{ma} is approximately equal to the average value of t_i.

(3) *energy*delay* - In sensor networks, it is important to consider both energy and delay. The combined energy*delay metric [15] can reflect both the energy usage and the end-to-end delay. We adopt the following definitions:

$$energy * delay_{cs} = e_{cs} \times t_{cs}, \qquad (6)$$

$$energy * delay_{ma} = e_{ma} \times t_{ma}. \qquad (7)$$

(4) *Packet delivery ratio* - It is the ratio of the number of data packets delivered to the sink to the number of packets generated by the source nodes.

V. PERFORMANCE EVALUATIONS

In this section, we compare the CSWSN and MAWSN with respect to the above performance metrics by simulation, and determine the conditions under which the MAWSN is more efficient than the CSWSN. Though these conditions are affected by many parameters, only a set of important parameters are chosen, such as the number of target sources (n_{source}), reduction ratio (r), size of processing code (S_{proc}), and size of sensed data at each sensor (S_{data}).

Although fusion factor (ρ) is also an important parameter, as described in Section 2.2, we will take a conservative approach to verify the efficacy of our scheme even in an adverse condition. Thus we set ρ to 1 in Table 1. For combined-task scenarios, it can be deemed that the data aggregation of each task does not work, or that each task only query data from a single source. When an MAWSN is applied to a wide-range of applications, this consideration is necessary. In the following sub-sections, four groups of simulations are evaluated. Only one parameter (i.e., n_{source}, r, S_{proc}, or S_{data}) is changed in each group while the other parameters are fixed.

A. Effect of the Number of Target Sources

In these experiments, we change n_{source} from 1 to 14 and keep all the other parameters in Table 1 unchanged. In Fig. 4, the energy consumptions in the CSWSN and MAWSN grow as n_{source} increases, but the MAWSN always consumes less energy. The larger n_{source} is, the more source-sink pairs exist. Thus, in Fig. 5, the average t_{cs} increases relatively slowly with respect to n_{source}, since multiple paths are used to deliver data packets in the CSWSN. However, despite the advantage of energy saving in the MAWSN, t_{ma} includes additional local processing time at each target source. When n_{source} is larger than 10, CSWSN outperforms MAWSN in terms of the average end-to-end delay in Fig. 5. From Figs. 7 and 8, it can be observed that the energy efficiency and end-to-end delay objectives conflict with each other in both CSWSN and MAWSN schemes. However, Fig. 6 shows that energy*delay of MAWSN is always lower.

For large n_{source}, we believe that multiple MAs should be dispatched in each round to keep t_{ma} within an acceptable value by compromising energy consumption.

In Fig. 4, t_{cs} levels out when n_{source} is larger than 10. This is caused by the loss of data packets. The larger n_{source} is, the more nodes are involved in delivering data packets in the CSWSN. Thus, congestion is more likely to happen. In contrast, only a single data flow is sent per round in the MAWSN. Fig. 7 shows that the MAWSN has a high packet delivery ratio even when n_{source} is large.

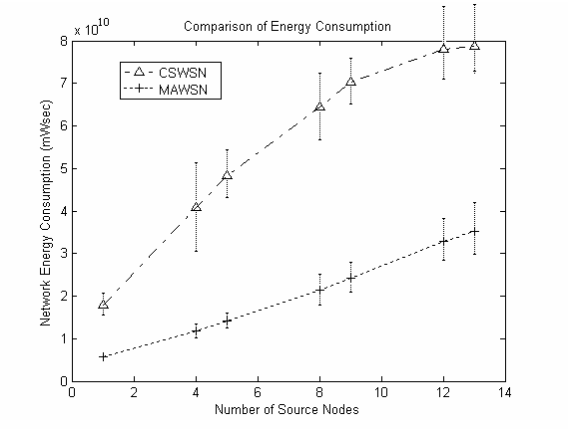

Figure 4. Comparison of Total Energy Consumption

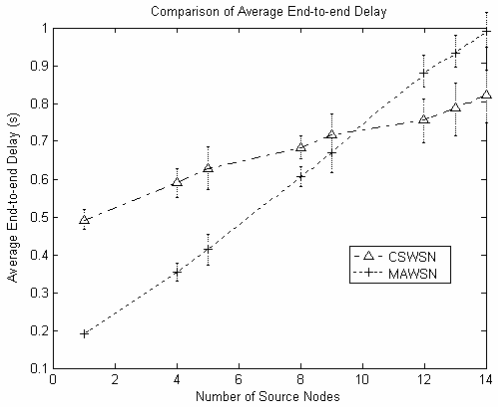

Figure 5. Comparison of Average End-to-end Delay

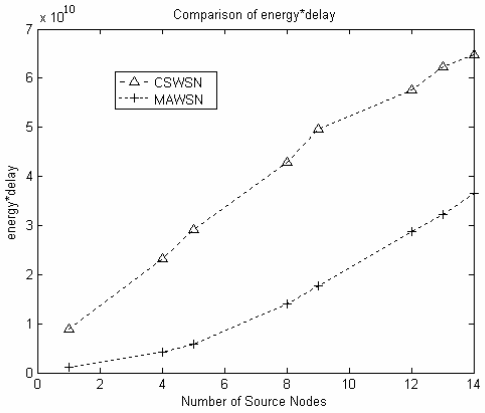

Figure 6. Comparison of Energy*Delay

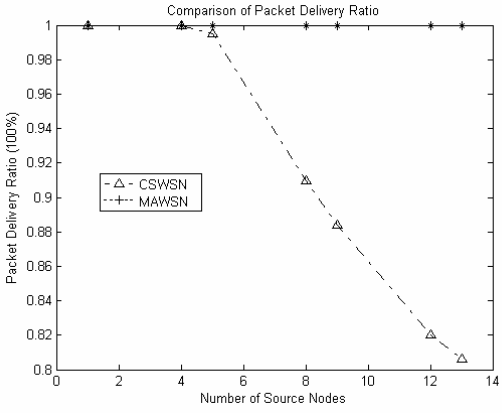

Figure 7. Comparison of Packet Delivery Ratio

B. Effect of Reduction Ratio

In these experiments, we change the reduction ratio *r* from 0.1 to 0.4, and keep the other parameters in Table 1 unchanged. Since changing *r* has an effect on MAWSN only, the performance of CSWSN is constant.

Fig. 8 shows that MAWSN always outperforms CSWSN in terms of energy consumption as *r* varies. Fig. 9 shows the results of end-to-end delay. When *r* is greater than 0.28, MAWSN tends to have longer end-to-end delays since the larger *r* is, the larger is the size of the aggregated data. In Fig. 10, energy*delay of MAWSN is lower when *r* is smaller than 0.35.

The results show that MAWSN is sensitive to *r*, which is an important factor to decide whether or not MAWSN outperforms CSWSN. The value of *r* is dependent on the type of application and the size of the processing code. For the same application, intuitively *r* gets smaller as the size of the processing code gets larger.

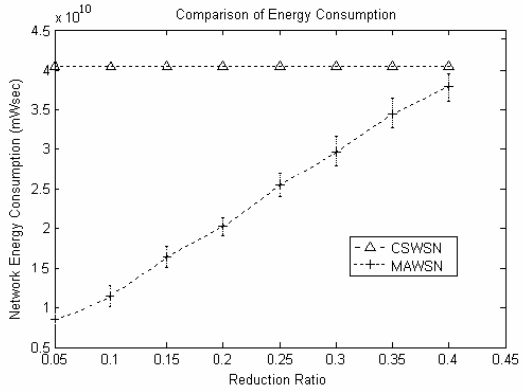

Figure 8. Comparison of Energy Consumption

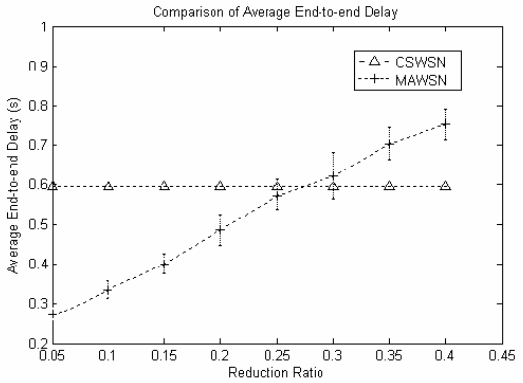

Figure 9. Comparison of Average End-to-end Delay

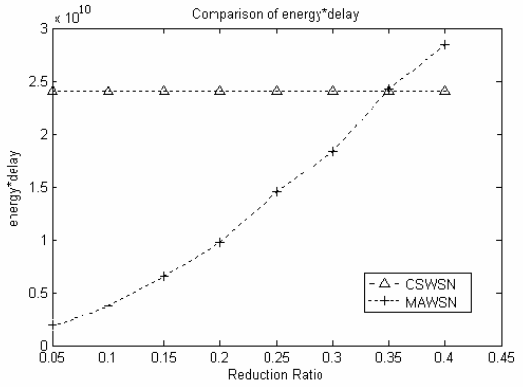

Figure 10. Comparison of Energy*Delay

C. Effect of Size of Processing Code

In these experiments, we change the size of the processing code (S_{proc}) from 0.1KB to 2KB. Since S_{proc} has nothing to do with CSWSN, its performance is constant. Over the range of processing code size considered, Fig. 11 shows that the energy consumption of MAWSN is always lower than that of CSWSN. The larger S_{proc} is, the more transmission overhead is needed by MAWSN. Fig. 12 shows the results of end-to-end delay. When S_{proc} is greater than 1.5KB, t_{ma} tends to be larger than t_{cs}.

According to the results of both Sections 6.2 and 6.3, it is observed that MAWSN is more sensitive to r than S_{proc}. Considering the tradeoff between r and S_{proc}, we prefer to use a larger S_{proc} to attain a lower r in return, hence improving the overall system performance.

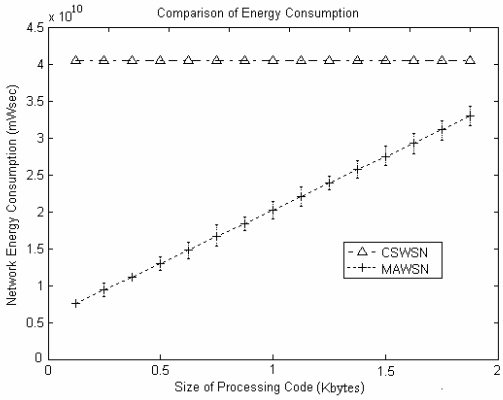

Figure 11. Comparison of Energy Consumption.

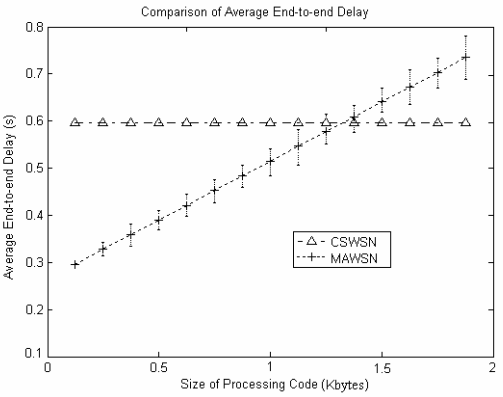

Figure 12. Comparison of Average End-to-end Delay

D. Effect of Size of Sensed Data

In these experiments, we change the size of the sensed data at each sensor (S_{data}) from 0.5KB to 4KB in 0.5KB interval. Fig. 13 shows that the energy consumption of CSWSN is always larger than that of MAWSN when S_{data} is varied. Fig. 14 shows that when S_{data} is greater than 1.5KB, CSWSN has a longer end-to-end delay. According to (1), the size of reduced data is equal to $r \cdot S_{data}$. Then, the transmission overhead of $(1-r) \cdot S_{data}$ can be saved per source node in the MAWSN. Thus, the

larger S_{data} is, the more efficient is the MAWSN. The above simulation results show that the MAWSN does not always perform better than the CSWSN. We observe that the end-to-end delay performance varies under different conditions, but in most cases, e_{ma} is lower than e_{cs}. Thus, for the scenarios where energy consumption is of primary concern, the MAWSN exhibits substantially lower energy consumption and hence potentially longer network lifetime than the CSWSN. Note that we conservatively set ρ to 1 in the experiments. We expect that as ρ decreases, the advantages of MAWSN will become more significant.

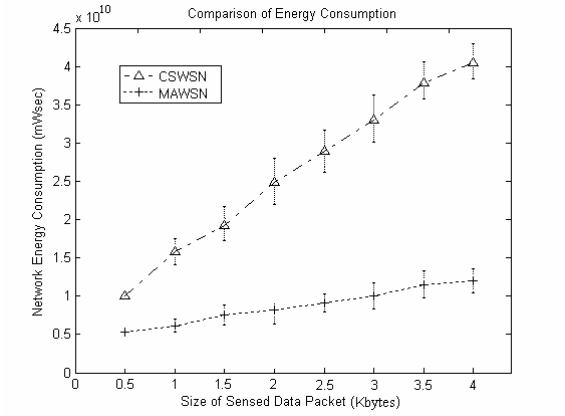

Figure 13. Comparison of Energy Consumption

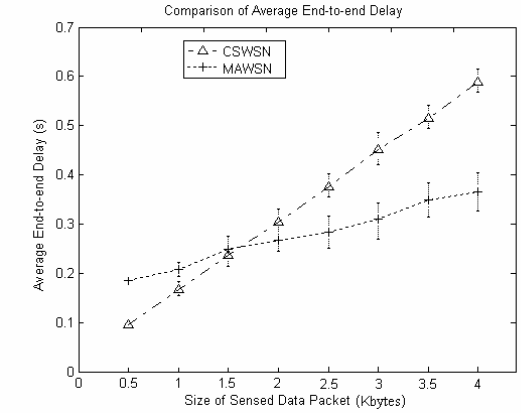

Figure 14. Comparison of Average End-to-end Delay

VI. Conclusions

In an environment where source nodes are close to each other, and considerable redundancy exists in the sensed data, the source nodes generate a large amount of traffic on the wireless channel, which not only wastes the scarce wireless bandwidth, but also consumes a lot of battery energy. Instead of each source node sending sensed data to the sink node, as typically occurs in client/server-based computing, this paper proposes a mobile agent paradigm for data processing/aggregating/ concatenating in a planar sensor architecture. The proposed architecture is called MAWSN (mobile agent based wireless sensor network). For a given set of parameters, we derive the conditions under which MAWSN exhibits better performances than the client/server-based paradigm in terms of packet delivery ratio, energy consumption, and the end-to-end delay.

Acknowledgement

This work was supported in part by the Canadian Natural Sciences and Engineering Research Council under grant STPGP 322208-05, and by the Basic Research Program of the Korea Science and Engineering Foundation under grant No. (R01-2004-000- 10372-0).

References

[1] K. Akkaya and M. Younis, “A Survey of Routing Protocols in Wireless Sensor Networks,” *in the Elsevier Ad Hoc Network Journal*, Vol. 3/3 pp. 325-349, 2005.

[2] Min Chen, Taekyoung Kwon, and Yanghee Choi, “Data Dissemination based on Mobile Agent in Wireless Sensor Networks,” *in Proc. IEEE LCN 2005*, pp.527-529, Nov. 2005.

[3] Hairong Qi, Yingyue Xu, Xiaoling Wang, “Mobile-Agent-Based Collaborative Signal and Information Processing in Sensor Networks,” *in Proceeding of the IEEE*, Vol. 91, NO. 8, pp.1172-1183,Aug. 2003

[4] Hairong Qi, Iyengar, S., Chakrabarty, K., “Multiresolution data integration using mobile agents in distributed sensor networks," *IEEE Transactions on Systems, Man and Cybernetics*, Vol.31 , No.3 , pp. 383-391, Aug. 2001

[5] Wu, Q., Rao, N.S.V., Barhen, J., etc, “On computing mobile agent routes for data fusion in distributed sensor networks,” *IEEE Transactions on Knowledge and Data Engineering*, Vol.16 , NO. 6 , pp. 740-753, June 2004

[6] Yu-Chee Tseng, Sheng-Po Kuo, Hung-Wei Lee and Chi-Fu Huang, “Location Tracking in a Wireless Sensor Network by Mobile Agents and Its Data Fusion Strategies,” *The Computer Journal*,Vol.47, No.4, July 2004: pp. 448-460

[7] K.N. Ross and R.D. Chaney, "Mobile Agents in Adaptive Hierarchical Bayesian Networks for Global Awareness," *Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics*, pp. 2207-2212, 1998

[8] Arkady Zaslavsky, “Mobile Agents: Can They Assist with Context Awareness?,” *IEEE MDM*, Jan. 2004 ,Berkeley, California

[9] Chee-Yee Chong, Kumar, S.P., “Sensor networks: evolution, opportunities, and challenges," *Proceeding of the IEEE*, Vol.91, NO. 8, pp. 1247- 1256, Aug. 2003

[10] G. Vigna, Mobile Agents: Ten Reasons For Failure *Proceedings of MDM 2004* 298-299 Berkeley, CA January 2004

[11] Fabio Silva, John Heidemann, Ramesh Govindan, and Deborah Estrin. “Directed Diffusion,” *Technical Report ISI-TR-2004-586, USC/Information Sciences Institute*, January, 2004.

[12] http://www.opnet.com

[13] Min Chen, "OPNET Network Simulation", *Press of Tsinghua University*, China, April, 2004, ISBN 7-302-08232-4, 352 pages.

[14] L.M. Feeney and M. Nilsson, “Investigating the energy consumption of a wireless network interface in an ad hoc networking environment,” *in Proc. of IEEE INFOCOM'01*, pp.1548-1557, April 2001.

[15] S. Lindsey, C. Raghavendra, and K.M. Sivalingam, “Data gathering algorithms in sensor networks using energy metrics,” *IEEE Trans.Parallel Distrib. Systems*, vol.13, no.9, pp.924-935, 2002.

Min Chen was born in Lin Chuan, China, on Dec. 1980. He received the BS, MS and Ph.D degrees from Dept. of Electronic Engineering, South China University of Technology, in 1999, 2001 and 2004 respectively. He is a postdoctoral fellow in the Communications Group, Dept. of Electrical and Computer Engineering, University of British Columbia. He was a post-doctoral researcher in the Multimedia & Mobile Communications Lab., School of Computer Science and Engineering, Seoul National University in 2004 and 2005. His current research interests include wireless sensor network, wireless ad hoc network, and video transmission over wireless networks.

Taekyoung Kwon is an assistant professor in the Multimedia & Mobile Communications Lab., School of Computer Science and Engineering, Seoul National University He received his Ph.D., M.S., and B.S. degrees in computer engineering from Seoul National University in 2000, 1995, and 1993, respectively. He was a visiting student at IBM T. J. Watson Research Center in 1998 and a visiting scholar at the University of North Texas in 1999. His recent research areas include radio resource management, wireless technology convergence, mobility management, and sensor network

Yong Yuan received the B.E. and M.E. degrees from the Department of Electronics and Information in Yunnan University, Kunming, P.R.China, in 1999 and 2002, respectively. Since 2002, he has been studying at the Department of Electronics and Information in Huazhong University of Science and Technology, P.R.China, as a Ph.D. candidate. His current research interests include wireless sensor network, wireless ad hoc network, wireless communication and signal processing.

Victor C. M. Leung received the B.A.Sc. and Ph.D degrees in electrical engineering from the University of British Columbia (UBC) in 1977 and 1981, respectively, where he was awarded the APEBC Gold Medal and the NSERC Postgraduate Scholarship. After working in MPR Teltech Ltd. and the Chinese University of Hong Kong, he returned to UBC as a faculty member in 1989, where he is currently a Professor, Associate Head of Graduate Affairs, and holder of the TELUS Mobility Research Chair in. the Department of Electrical and Computer Engineering. His research interests are in the areas of architectural and protocol design and performance analysis for computer and telecommunication networks, with applications in satellite, mobile, personal communications, and high-speed networks. Dr. Leung is a Fellow of IEEE, a voting member of ACM, and an Editor of the IEEE Transactions on Wireless Communications and Vehicular Technology.