Data Aggregation and Dilution by Modulus Addressing in Wireless Sensor Networks

Erdal Cayirci, *Member, IEEE*

***Abstract*—DADMA is a distributed data aggregation and dilution technique for sensor networks where nodes aggregate/dilute sensed data by following the rules given in an SQL statement. Our test results show that DADMA reduces the number of transmitted packets 60% on the average.**

***Index Terms*—Addressing, data aggregation, data dissemination, data fusion, sensor networks, task assignment.**

I. Introduction

SINCE the lifetime of a wireless sensor network (WSN) is generally dependent on irreplaceable power sources in tiny sensor nodes, power efficiency is one of the critical constraints for WSNs [1]. Data aggregation techniques that reduce the number of data packets conveyed through the network are therefore important while also being required for effective fusion of data collected by vast number of sensor nodes [1], [2]. The following characteristics of WSNs should be considered while designing a data aggregation scheme.

- Sensor nodes are limited in both memory and computational resources. They cannot buffer large number of data packets.
- Sensor nodes generally disseminate short data packets to report the amplitude for an attribute, e.g., temperature, pressure, humidity, etc.
- The observation areas of sensor nodes often overlap. Therefore, many sensor nodes may report the correlated data related to the same attribute. However, in many cases the replicated data is needed because sensor network concept is based on the cooperative effort of low fidelity sensor nodes [1].
- Since there may be thousands of nodes in a sensor field, associating data packets from numerous sensors to their events, and correlating the data about the same event reported at different times may be too complex a task for a single sink node or a central system.
- Due to large number of nodes and other reasons such as power efficiency, sensor nodes are not globally addressed [1].

Our novel data aggregation and dilution by modulus addressing (DADMA) protocol is designed as to fit these characteristics. Besides, DADMA can be used in conjunction with the known routing schemes such as directed diffusion [3], SPIN [2] and LEACH [2], does not need time synchronization, supports node mobility, and allows to control sensor off duty cycles by a middleware architecture.

For DADMA, a WSN is perceived as a distributed relational database that has a single view created by joining records distributed in the local sensor node tables. Sensed data is retrieved from the WSN by an SQL statement. While the data is being retrieved, it is also aggregated or diluted based on the rules defined in the SQL statement.

The remainder of this paper is organized as follows: In Section II, we describe our new system. The experimental results are given in Section III. Then, we conclude our letter.

II. Data Aggregation and Dilution

In DADMA, a sensor network is a distributed relational database composed of a single view that joins local tables located at sensor nodes. Records in local tables are the measurements made upon a query arrival and consist of two fields, namely *task* and *amplitude*. Since a sensor node may have more than one sensor attached to it, *task* field, e.g., temperature, humidity, etc., indicates the sensor that makes the measurement. Sensor nodes have limited memory capacity and they do not store the results of measurements. Therefore, *task* field is the key field in the local tables created upon a query arrival. Our perception of WSNs makes relational algebra practical to retrieve the sensed data without much memory requirement, which is different from the scheme explained in [4] where the sensed data for each task is maintained at a different column in a table. Moreover, sensor nodes decide whether they are required to be involved in a query or not by the use of DADMA, and also data is aggregated while being conveyed through the sensor network.

Sensor network database view (SNDV) can be created temporarily either at the sink or at an external proxy server. An SNDV record has three fields, i.e., *location, task* and *amplitude*. While data is being retrieved from a sensor node, the sensed data is also joined with the *location* of the sensor node. Since multiple sensor nodes may have the same type of sensors, i.e., multiple sensors can carry out the same sensing *task, location* and *task* fields become the key in an SNDV. For many WSN applications, the sensed data is needed to be associated with location data. For example, in target tracking and intrusion detection WSNs, sensed data is almost meaningless without relating it to a location. Therefore, location awareness of sensor nodes is a requirement imposed by many WSN applications. There are a number of practical location finding techniques reported for WSNs [5].

It is also possible to maintain a database in a remote proxy server where the records obtained from queries, i.e., the records at an SNDV, are stored after being joined with a *time* label. For example a daemon can generate queries at specific time intervals, and insert the records in the SNDV resulting from these

Manuscript received February 5, 2003.The associate editor coordinating the review of this paper and approving it for publication was Dr. P. Demestichas.

The author is with the Wargaming and Simulation Center in Turkish War Colleges, Yeni Levent 80625, Istanbul, Turkey (e-mail: cayirci@cs.itu.edu.tr).

Digital Object Identifier 10.1109/LCOMM.2003.815663

queries into the database after joining them with a *time* field. Note that each query results in a new SNDV where the results of the query are gathered temporarily.

A. Querying in a Sensor Network by DADMA

A statement that has the structure given below starts a query. Note that the standard SQL notation is used in this statement except for the last field starting with "*based on*" keyword.

```
Select [task, time, location, [distinct |
  all], amplitude,
  [[avg | min | max | count | sum] (ampli-
  tude)]]
from [any , every , aggregate m, dilute
  m]
where [power available [<|>] PA | location
  [in | not in] RECT |
  t_min < time < t_max | task = t | amplitude
  [< |==| >] a]
group by task
based on [time limit = l_t | packet limit =
  l_p | resolution = r |
  region = xy]
```

A user can retrieve a subset of data fields available in an SNDV, and can aggregate *amplitude* data either by grouping data based on task and/or by using *aggregate m* function given in (1). Some of the sensor nodes can also be excluded from a query by *dilute* m function in (2)

$$f(x) = x \operatorname{div} m \quad (1)$$

$$f(x) = (x/r) \bmod (m/r) \quad (2)$$

where x is the grid location of a node relative to one of the axes, r is the resolution in meters, and m is the dilution or aggregation factor.

When *dilute* m command is given by the user, every sensor node first uses (2) to find out its location indices in horizontal and vertical axes and then compare these indices with the region values x and y sent in the "*based on*" field of the query. If they match, the sensor node replies the query. For example, the location indices of a sensor node at location $\{46, 74\}$ are $\{3, 1\}$ for $m = 8$ and $r = 2$. Therefore, if the region value in the query is $\{3, 1\}$, this sensor should respond. Hence, only the sensor nodes in r × r meter squares located in every m meters react to the query, and the others stay idle. This is a practical technique especially when sensor nodes are randomly deployed according to uniform distribution.

For the same example the indices found out by using (1) are $\{5, 9\}$. When aggregate m command is received, the values measured by a sensor node are aggregated with the values measured by the other nodes having the same indices. Hence, we can address the sensor nodes at certain geographic locations, and aggregate data based on the location of nodes.

B. Implementation of DADMA

Sensor nodes run the following algorithm upon a call arrival:

```
If (newtaskreceived (querry))
{
  t_i=currenttime; received=0;
  broadcast(querry); //repeat the received
  querry
  currentlist=createtasklist();
  if (taskavailable(querry.task)) {
    run (querry.task, amplitude);
    addtasklist (querry.task, amplitude,
  currentlist);
    send (currentlist);
    initializetasklist (currentlist); }
  while (!received&&currenttime- t_i < l_tmin)
    if (newdatareceived (tasklist)) {
      received++;
      addtasklist (tasklist, currentlist);
  }
  if (!received&&currenttime - t_i ≥ l_tmin)
      send("push"); //this is a leaf node
  else
  {
    while (currenttime-t_i < l_t)
    {
      if (!received)
        while (!received&&currenttime-t_i < l_t)
          if (newdatareceived(tasklist)) {
            received++;
            addtasklist (tasklist, cur-
  rentlist); }
      while (received&&currenttime-t_i < l_t
  &&received<l_p)
        if (newdatareceived(tasklist))
          if (tasklist.task=="push")
            received--;
          else {
            received++;
            addtasklist (tasklist, cur-
  rentlist); }
      if (notempty(currentlist) {
        send(currentlist);
        initializetasklist(currentlist); }
    }//end while
    send ("push");
  }//end else
}//end if
```

When a sensor node receives a query, it first repeats the query if it is involved in task dissemination process. Then, it determines if it can carry out the task, e.g., measuring temperature, humidity, pressure, etc., and be involved in the query according to the aggregation and dilution rules. If the node is supposed to reply to the query, it makes the measurements, i.e., create a virtual sensor node table, then reports the records in the table to its gradient. Note that this process uses the services provided by network layer and is independent from the routing algorithm. Then the node waits at least a system defined time period $l_{t\,\min}$. If it does not receive any data from any other node during $l_{t\,\min}$, this indicates that it is a leaf node. Therefore, it sends a "push" packet to its gradient to notify the gradient not to wait for any other data packet from it to aggregate. The incoming data packets are aggregated based on aggregation rules until l_p packets are received. This process continues until the time period l_t expires or "push" packets are received from all children. Note that l_p and l_t are given in the "based on" field of a query. The node terminates the task by sending a "push" packet at the end of l_t.

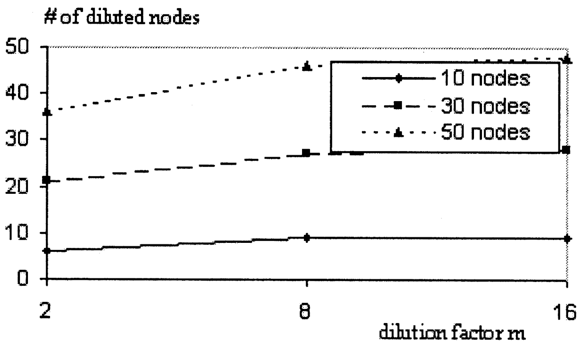

Fig. 1. Number of diluted nodes for varying dilution factors when dilution resolution is 2 m.

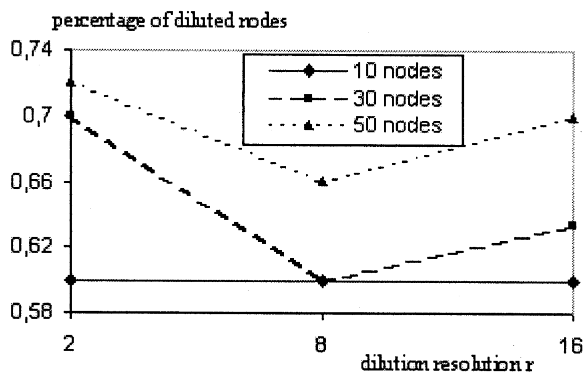

Fig. 2. Percentage of diluted nodes for varying dilution resolution when dilution factor is 2.

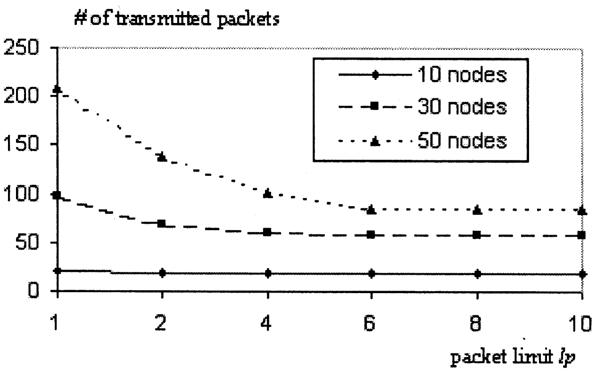

Fig. 3. Number of transmitted packets for varying number of packet limits (aggregation factor $= 8$, aggregation resolution $= 2$).

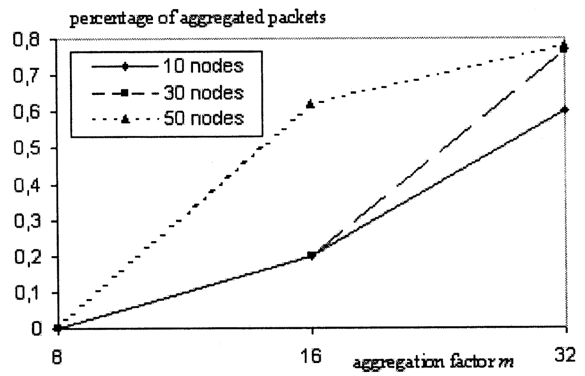

Fig. 4. Percentage of aggregated packets for varying aggregation factors (aggregation resolution $= 2$, packet size limit $= 2$).

III. Experimental Results

In this section, the gains of DADMA in the number of transmitted data packets, and data packets discarded by aggregation or dilution are evaluated. The affects of changing aggregation/dilution factor m, packet limit l_p, and resolution r are examined. We simulate 10–50 sensor nodes and a sink randomly deployed over an area 74 m $\times$ 74 m in size. In simulations, we assume that every node has data to respond the queries to ensure the fairness and to test our system against the worst-case scenario.

The network layer protocol used in our simulations is directed diffusion [3]. To the best of our knowledge, there is not any alternative technique that we can fairly compare our scheme. Therefore, we compare two cases, namely directed diffusion with DADMA and without DADMA to examine the performance gains. Since physical layer parameters such as path loss exponent and hop distance do not affect our results, they are not factoring parameters in our experiments.

In Figs. 1 and 2, we evaluate the number of nodes not involved in a query for varying node densities, dilution factor m and resolution r parameters. The percentage of diluted nodes increases as the dilution factor m and node density increase. The percentage of diluted nodes is between 60% and 73%for dilution factor m is 2.

In Fig. 3, the number of transmitted data packets are shown for varying node densities and packet limit l_p. When l_p is 1, the system cannot use DADMA. Therefore, the first plots in the curves are for the directed diffusion without DADMA case. When node density is 1 node per 100 m^2, and packet limit l_p is 6, DADMA reduces 60% of transmissions comparing to directed diffusion without DADMA case. Note that the higher the node density is, the higher the reduction in the number of transmissions becomes.

In Fig. 4, the percentage of the data packets aggregated with other data packets while being conveyed through the sensor network is shown. When aggregation factor is 32 and packet limit m is 6, almost 80% of the sensed data packets are aggregated with the other data packets based on the given aggregation rules on their way to the sink.

IV. Conclusions

DADMA is a middleware for data aggregation and dilution. The aggregation/dilution rules are given by an SQL statement, and a simple algorithm run on sensor nodes follow these rules while conveying the data packets coming from the other nodes. Our experiments show that DADMA reduces 60% of data transmissions on the average, and manage to aggregate up to 80% of data packets while being conveyed through the sensor network.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks,” *IEEE Commun. Mag.*, pp. 102–114, Aug. 2002.

[2] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, “Adaptive protocols for information dissemination in wireless sensor networks,” in *ACM MobiCom’99*, 1999, pp. 174–185.

[3] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed diffusion: A scalable and robust communication paradigm for sensor networks,” in *ACM MobiCom’00*, 2000, pp. 56–67.

[4] S. Madden, R. S. Szewczyk, M. J. Franklin, and D. Culler, “Supporting aggregate querries over ad-hoc wireless sensor networks,” in *4th IEEE Workshop on Mobile Computing Systems and Applications*, 2002.

[5] A. Nasipuri and K. Li, “A directionality based location discovery scheme for wireless sensor networks,” presented at the 1st ACM International Workshop on Wireless Sensor Networks and Applications, 2002.