Causally Ordered Delivery Protocol for Overlapping Multicast Groups in Broker-based Sensor Networks

Chayoung Kim[1] **and Jinho Ahn**[2*]

[1] **Contents Convergence Software Research Center, Kyonggi University**
Suwon, Gyeonggi-do 443-760, Republic of Korea

[2] **Dept. of Computer Science, College of Natural Science, Kyonggi University**
Suwon, Gyeonggi-do 443-760, Republic of Korea

Abstract

In sensor networks, there is a lot of overlapping multicast groups because of many subscribers, associated with their potentially varying specific interests, querying every event to sensors/publishers. Also gossip-based communication protocols are promising as one of potential solutions providing scalability in P(Publish)/ S(Subscribe) paradigm in sensor networks. Moreover, despite the importance of both guaranteeing message delivery order and supporting overlapping multicast groups in sensor or P2P networks, there exist little research works on development of gossip-based protocols to satisfy all these requirements. In this paper, we present a causally ordered delivery guaranteeing protocol for overlapping multicast groups, based on sensor-brokers as delegates. In the protocol based on sensor-broker, sensor-broker might lead to make overlapping multicast groups organized by subscriber's interests. The message delivery order has been guaranteed consistently and all multicast messages are delivered to overlapping groups using gossip-based protocols by the sensor-broker. Therefore, these features of the protocol based on sensor-broker might be significantly scalable rather than those of the protocols by coordinated groups like traditional committee protocols.

Keywords: *Sensor Network, Group Communication, Overlapping Multicast Groups, Scalability, Reliability.*

1. Introduction

A wireless sensor network(WSN) has important applications such as remote environmental monitoring, target tracking, natural disaster relief, biomedical health monitoring, hazardous environment exploration and seismic sensing and virtual worlds such as massive multiplayer games [18, 25]. The design of WSN depends significantly on the application, and it must consider factors such as the environment, the application's design objects and system constraints [1, 5, 25]. The environment plays a key role in determining the size of the network and the network topology. For indoor environment, fewer sensor nodes are required to form a network in a limited space whereas outdoor environments may require more sensor nodes to cover a larger area [1, 5, 25]. There are two types of WSNs: structured and unstructured. An unstructured WSN is one that constrains a dense collection of sensor nodes, deployed in an ad hoc manner into the field. In an unstructured WSN, network maintenance such as managing connectivity and detecting failures is difficult since there are so many nodes [25]. In a structured WSN, all or some of the sensor nodes are deployed in a pre-planned manner. The advantage of a structured network is that fewer sensor nodes can be deployed with lower network maintenance and management cost [25]. For reliable communication for this sensor networks, services such as congestion control, acknowledgements, and packet-loss recovery are necessary to guarantee reliable packet delivery [25]. Especially, these above applications need a variety of collaboration features, such as chat windows, white boards, p2p video and other media streams, and coordination mechanisms [11]. So, the use of p2p overlapping groups is expected to generate new models of interactive communication and cooperation to support these applications [25] in sensor networks. A new data dissemination paradigm for such sensor networks is different from mobile ad-hoc networks in a method of designing data propagation and aggregation generated by various and lots of sensor nodes [1, 5, 21]. There are several researches based on the P (publish)/S (subscribe) paradigm [8] in the area of sensor network communications to address the problem of querying sensors from mobile nodes in order to minimize the number of sent result packets [14, 20]. In P/S paradigm systems, a query node periodically runs an algorithm to identify the sensors it wishes to track and "subscribe" to these sensors of their interest, and the sensors periodically "publish" [14, 20]. So, the intermediate sensors in the networks, along the reverse path of interest propagation might aggregate the query results by combining reports from several sensors [14]. An important feature of that interest and data propagation and aggregation are determined by localized sensors interactions [14]. And performing local computations to reduce the amount of data before transmission can obtain orders of magnitude energy savings [14, 20]. Recently, gossip-based protocols seem more appealing in many P/S systems because they are more scalable than traditional reliable broadcast [3] and network-level protocol deriving from IP Multicast for many of the various applications requiring reliable dissemination of events [7, 15]. In gossip-based protocols, when a process sends a multicast message, it randomly selects a small subset of members, called gossip targets. The number of gossip targets is called fan-out, which relates reliability of gossip-based protocols. Usually the time necessary to reach all processes in a group is logN, where N is the size of the group, the maximum number of gossip rounds. This approach often relies on the assumption that every process knows every other process [4]. Gossip-based protocols have turned out to be adequate for large scale settings by achieving a "high degree of reliability" and strong message delivery ordering guarantees offered by deterministic approaches [6, 7, 10]. The seminal probabilistic broadcast (pbcast) algorithm of Birman et. al. [4] is originally described as a broadcast presented in the system based on global view and Eugster’s algorithm(lpbcast) [7] is implemented for P/S systems as a broadcast. These previously developed gossip-based protocols implicitly assume that all processes in a group are interested in all events [4]. Such a flooding technique is not adequate when many events are only of interest for lower than half the processes in the overall groups [6]. PMCAST [6] deals with the case of multicasting events only to subsets of the processes in a large group by relying on a specific orchestration of process as a superimposition of spanning trees. The delegates of this protocol [6] yet are themselves not interested in the same topics as subscribers have. But, when multicasting an event, PMCAST [6] follows the underlying tree, by gossiping depth-wise, starting at the root and the interested subscribers of overlapping multicast groups receive their event messages [6]. An atomic broadcast on gossip-based protocols is implemented in Birman et. al. [4] and Eugster et. al. [6] for P/S systems. But, these protocols are performed by hierarchical membership protocols [6] for each delegate group or the totally ordered delivery properties are maintained by global member views [4]. These features are likely to be highly overloaded on each member and not scalable. Also, there is no causal order guaranteeing multicast protocol supporting overlapping multicast groups, useful for many distributed applications with a variety of collaboration features, such as chat windows, white boards, p2p video and other media streams, based on publisher(sensor-broker) in the previously developed protocols. A causal ordering protocol ensures that if two messages are causally related and have the same destination, they are delivered to the application in their sending order [3]. Consider a distributed application that uses a sensor, a controller and a monitor for machine monitoring. The controller aggregates sensor notifications and controls the machine. In some cases, the controller decides to stop the machine due to a notification from the sensor, in which case the sensor also sends a reading to the monitor. In this case, causal order would ensure that the monitor would receive the sensor reading before the stop notification. The wrong order would falsely indicate a malfunction in the controller, that is, the delayed sensor reading could indicate that the machine was still operating [22]. In [16], Kim et al. suggested an efficient and scalable causal order guaranteeing multicast protocol to use only local views supporting overlapping multi-groups. In the proposed protocol, there is no sensor-broker. So, overlapping multi-groups are defined by only subscribers’ interests. The messages of join/leave are disseminated by gossip communication based on its local views. Messages including causal context graphs [19] based on group identification are delivered to the application layer without any sensor-brokers [6, 9]. In this paper, we present a causal order guaranteeing multicast protocol based on sensor-brokers as delegates that aggregate the information of results in sensor networks, periodically gossip about the messages of them and guarantee causally ordered delivery of the messages in the face of transient member population. This protocol is appropriate for sensor networks in a pre-planned manner. Fewer sensor nodes can be deployed since nodes are placed at specific locations to provide small coverage. In this structured network, there are lower network maintenance and management costs because fewer sensor nodes cannot be changed frequently.

2. Background

2.1 System Model

In the distributed system, a group consists of a set of processes. Processes join and leave the system dynamically and have ordered distinct identifiers. The process maintains a local membership list called a "local view". It can send unicast messages to another process through the communication network. A finite set of processes communicate only by exchanging messages over a fully connected, point-to-point network. Processes communicate using the primitives *send(m)* and *receive(m)*. Communication links are fair-lossy, but correct processes can construct reliable communication links on top of fair-lossy links by periodically retransmitting messages. Each member performs operations according to a local clock. Clock rates at all members are the same. Runs of the system proceed in a sequence of rounds. Members may undergo two types of failures, both probabilistic in nature. The first is process failure. There is an independent, per-process probability of at most Υ that a process has a crash failure during the finite duration of a protocol. Such processes are called faulty. Processes that survive despite the failures are correct. The second type of failures is message omission failure. There is an independent, per-message probability of at most δ that a message between non-faulty processes experiences a send omission failure. The union of all message omission failure events and process failure events are mutually independent. For simplicity, we do not include process recovery in the model. Also, we expect that both Υ and δ are small probabilities. There are no malicious faults, spurious messages, or corruption of message i.e. we do not consider Byzantine failures.

In proposed protocols, a group of processes is defined through two primitives PMCAST and PDELIVER, which use gossip protocols to provide probabilistic reliability in networks. Processes communicate with these two pairs of primitives, PMCAST and PDELIVER, which model unreliable communication associated with probability α of successful message transmission. We refer to probability α as the expected reliability degree. These primitives are as follows: (**Integrity)** For any message *m*, every correct process PDELIVER *m* at most once, and only if *m* was previously PMCAST by *sender(m)*. (**Validity**) If a correct process *p* PMCASTs a message *m* then *p* eventually PDELIVERs *m*. (**Probabilistic Agreement**) Let *p* and *q* be two correct processes. If *p* PDELIVERs a message *m*, then with probability α, *q* PDELIVERs *m*. In other terms, the only probabilistic property is Agreement. This probabilistic notion of agreement also captures a weakly consistent membership of local view, typical for large scale settings.

2.2 Related Work

In P. Eugster et. al [6], the protocol deals with the case of multicasting events only to subsets of the processes in a large group by relying on a specific orchestration of process as a superimposition of spanning trees. But, PMCAST [6] is not also a genuine multicast[13] because of considering delegates. Birman et al. [4] proposed a gossip-style protocol called bimodal multicast thanks to its two phases: a "classic" best-effort multicast such as IP multicast is used for the first rough dissemination of messages. The second phase assures reliability with a certain probability by using a gossip-based retransmission. But gossip-based broadcast protocols based on Lpbcast [7] proposes gossip-based broadcast membership mechanisms based on a partial view without a global view. Each process has a randomly chosen local view of the system. Lpbcast [7] is a completely as a decentralized membership protocol because of no dedicated messages for membership management based on gossips. In Eugster's algorithm [9], atomic probabilistic broadcast (apbcast) implemented for publish/subscribe[8] programming is a hybrid approach. Its deterministic ordering of messages ensures the consistency of the delivery order of broadcast messages and its probabilistic propagation of broadcast messages and order information provides a high level of reliability in the face of an increasing number of process failures because of more heroic efforts by making use of the membership of delegates. However, building such the membership of delegates requires the global knowledge of membership, and it may be very difficult to maintain such the structure in the present of joins/leaves of processes. Probabilistic Atomic Broadcast (pabcast) [9] is fully probabilistic by mixing message atomic ordering and propagation, basing these on gossips without a membership of delegates. But, a promising approach for increasing scalability is to weaken the deterministic ordering guarantees to make the properties of dependencies between broadcast messages probabilistic. Also, it does not give the guarantees achieved for the consistency of the delivery order of overlapping groups.

As a fundamental problem in distributed computing, much effort has been invested in solving atomic broadcast [3]. Early work such as [3] mostly focuses on stronger notions of Agreement and also membership than the proposed protocols discussed in this paper. In [19], an inter-process communication mechanism, called Psync, explicitly encodes partial ordering with each message. Psync is based on a conversation abstraction that provides a shared message space through which a collection of processes exchange messages. The general form of this message space is defined by a directed acyclic graph that preserves that partial order of the exchanged messages. And there are researches based on the P (publish)/S (subscribe) paradigm [8] in the area of sensor network communications to approach the problem of querying sensors from mobile nodes [14, 20]. Directed Diffusion [14] can be seen as publish-subscribe mechanism, which is implemented using the tree-based architecture rooted at the publisher. SENSTRACT [20] is mapping from queries to topics and the corresponding underlying sensor network structure. SENSTRACT [20] is a tree-based P/S system structured by service providers as roots, representing one of the data-centric routing protocols for data dissemination of sensor networks.

Recently, there is a gossip-based technique that GO [24] is a new platform support for gossip applications targeted to large-scale deployments. Adaptive Peer-Sampling [23] focuses on Newscast, a robust implementation of peer sampling service. It is an adaptive self-control mechanism for its parameters, namely-without involving failure detectors-nodes passively monitor local protocol events using them as feedback for a local control loop for self-tuning the protocol parameters. The research work [2] presents P3Q, a fully, decentralized gossip-based protocol to personalize query processing in social tagging systems. This P3Q does not rely on any central server: users periodically maintain their networks of social acquaintances by gossiping among each other and computing the proximity between tagging profiles.

3. The Proposed Protocol

3.1 Basic Idea

In sensor-broker based protocol, some sensors that are designed as brokers might lead to make overlay networks and query nodes subscribe to all the topics that match their interest. The mapping of subscribers and brokers is entirely driven by the query application. Recently, much research has been devoted to designing broker selection methods that best suits application needs [15, 20]. Each sensor can provide information periodically with some of its brokers by peer-sampling services [15], assumed to be implemented in the systems and the brokers might aggregate the query results by combining reports from several sensors. The aim of peer-sampling services is to provide every node with peers to exchange information with[15]. This assumption has led to rigorously establish many desirable features of gossip-based broadcast protocols like scalability, reliability, and efficiency [15] and a wide range of higher funtions, which include information dissemination, aggregation, and network management [15]. We also consider the reliability of the service by examining its self-healing capacity and robustness to failure. If all brokers representing a sensor grid may be stale, all members of brokers are not changed because of periodical peer-sampling services [15], i.e., the failure of brokers having their interests is tolerant by self-healing functions. But, if all subscribers in a grid lose their interests in information published on a particular topic, the brokers representing the grid send leave messages to the corresponding overlay multicast groups and then all of their group members are updated by leaving brokers. Query nodes subscribe to the corresponding sensor-broker networks of their interest and receive the results of the queries. The sensor-broker periodically gossip about the messages of the results [20] and guarantee causally ordered delivery of the messages with aggregating the information of the results in overlapping networks. In this protocol, because every broker knows every other brokers, a VT(vector time) for each broker, pi is a vector of length n, where n = the number of broker members. And this protocol leads us to extend single VT to multiple VTs because a broker belongs to several interest overlapping groups [3]. Every sensor-broker maintains a VT for each group and attaches all the VTs to every message that they multicast. Also, epoch protocol [1] is assumed to be implemented for the member leaving overlapping multicast groups and each group membership list updated by the member. The digest information of vector and membership list is sent or received periodically using gossip-based protocols [4]. Therefore, this protocol might make up transient faults of sensor-broker with the peer-sampling services [15] and deal with brokers' leave by membership management. That is, the processing of temporal faults is different from that of brokers' leave. So, these features of this protocol might result in its very low membership management cost compared with the cost incurred by maintaining member list for traditional committee in the previous protocols [4].

3.2 Algorithm Description

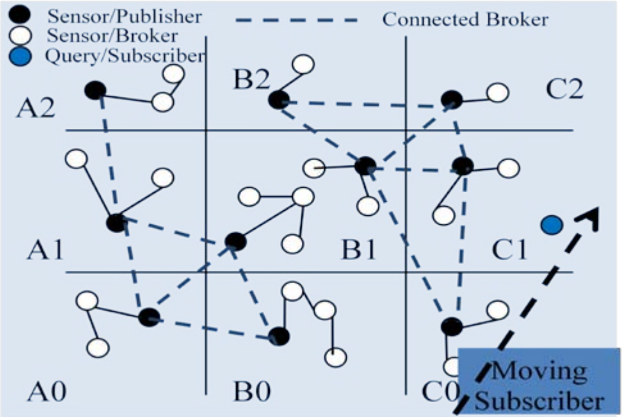


Fig. 1 Publishers(Sensors) vs. Subscribers

In figure 1, there is a two-dimensional area of interest(AoI), which the sensor-broker publishes messages to a particular topic, while query nodes subscribe to all the topics that match their interests. In this protocol based on sensor-broker, we present a causal order guaranteeing multicast protocol supporting overlapping subscriber groups and useful for these applications with a variety of collaboration features, such as chat windows, white boards, p2p video and other media streams, and coordination mechanisms requiring causally ordered delivery of messages.

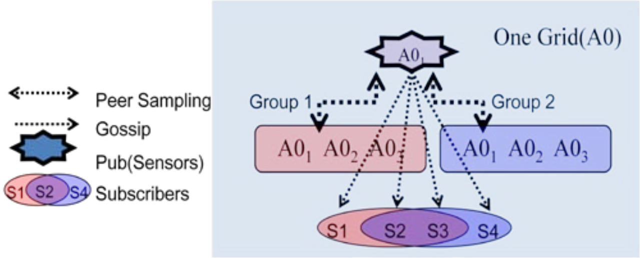
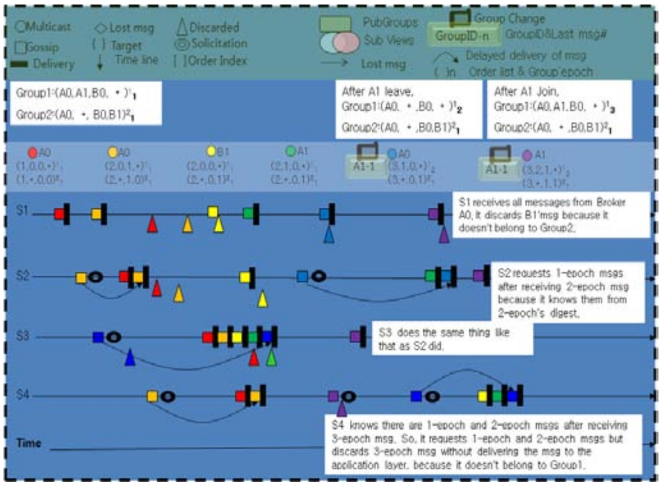


Fig. 2 Sensor-Broker $A0_1$ covering a Grid

In figure 2, we can see that a sensor-broker $A0_1$ in a grid A0 of a sensor network like one of figure 1 publishes desired messages of query results to all query nodes, {s1, s2, s3} and {s2, s3, s4} subscribing to their topics, Group1 and Group2 respectively using gossip-style disseminations. There are overlapped members={s2, s3} subscribing to the topics of Group1 and Group2.



S1 receives all messages from Broker A0, It discards B1'msg because it doesn't belong to Group2.

S2 requests 1-epoch msgs after receiving 2-epoch msg because it knows them from 2-epoch's digest.

S3 does the same thing like that as S2 did.

S4 knows there are 1-epoch and 2-epoch msgs after receiving 3-epoch msg. So, it requests 1-epoch and 2-epoch msgs but discards 3-epoch msg without delivering the msg to the application layer, because it doesn't belong to Group1.

Fig. 3 Example of message deliveries from sensor-broker to subscribers

Figure 3 shows that sensor-broker gossip about multicast messages piggybacked with all VT clocks for all of interesting groups to guarantee causally ordered delivery of messages in a sensor network like one of figure 2. This example in figure 3 illustrates Group1={A0,A1,B0}, Group2={A0,B0,B1}, Subscribers={S1,S2,S3,S4} and maximum number of Gossip Rounds = 2. The overlapping subscriber members={S2,S3} receive all messages from Group1 and Group2, S1 receives messages only from Group1 and S4 receives messages only from Group2. In the figure 3, there are VT clocks $((0,0,0,*)^1{}_1, (0,*,0,0)^2{}_1)$ for each group, Group1 and Group2, to depict each $VT^g{}_e$ as a vector of length *n*, with a subscript epoch variable, $_e$ for covering the cases of a process leave and join and a special entry * for each process that is not a member of $Group^g{}_e$. For each message generated by a member, each $VT^g{}_e(p_i)[i]$ is incremented by 1. So, if a member A0 generates a multicast message, then $VT^1{}_1$ and $VT^2{}_1$ is $((1,0,0,*)^1{}_1$ and $(1,*,0,0)^2{}_1)$ respectively. And when a member A1 leaves and joins Group1 again, $VT^1{}_2$ for Group1 is from $(*,*,0,*)^1{}_2$ to $(*,0,0,*)^1{}_3$ because subscript epoch is changed from 2 to 3. Figures 2 and 3 show an example of the protocol based on sensor-broker with causal ordering VT clocks in what order is A0->A0->B1->A1->A0->A1. In this case that subscribers know what messages should be delivered according to causal ordering VT clocks piggybacked by multicast messages and delay some messages after comparing their causal ordering VT clocks and validating their receipt of predecessor. Also, there are undesired messages are sent to a subscriber, forcing it to discard them. Subscriber S1 receives all messages from Broker A0. It discards B1's message without delivering it to the application layer because S1 doesn't belong to Group2. Subscriber S2 requests some of 1-epoch messages to the latest gossip-sender after receiving 2-epoch messages because it knows that some of 1-epoch messages are not received from piggybacked 2-epoch's digest. Subscriber S3 does the exactly same thing as S2 did because it knows that all of 1-epoch messages are not received. Subscriber S4 knows that 1-epoch and 2-epoch messages are not received after receiving 3-epoch message. So, it requests 1-epoch and 2-epoch messages but discards 3-epoch message because S4 does not belong to Group1and 3-epoch message is generated by A1, the member of Group1. Subscriber S4 can check 3-epoch's digest for validating causal ordering VT clocks and discard the 3-epoch message without delivering it to the application layer. And S4 solicits the retransmission of 1-epoch messages and 2-epoch messages to the latest gossip-sender.

The data structures and procedures for sensors and subscribers in our protocol are formally given in figures 4 and 5.

```
Procedure RECEIVE_JOIN   //Join MSG is received
Merge member=q with p.Local_View
Adjust p.Local_View by Fixed_Size  //Randomly Selected Local View
Update member=q and epoch in Overlapped_Multicast_Vector
Procedure RECEIVE_LEAVE    //Leave MSG is received
if q.Last_m.Seq not received then  //Check the LAST MSG
   send Solicit_Retransmission(m) to Latest_Gossip_Sender
Merge q.Local_View with p.Local_View
Adjust p.Local_View by Fixed_Size  //Randomly Selected Local View
Update don't care member=q and epoch in Overlapped_Multicast_Vector
Procedure RECEIVE_DIGEST
If m in Digest is not in Pendling_List or not delivered then
   send solicit_Retransmission(r) to Latest_Gossip_Sender
   //Check the VT'summary whether or not interest MSG is not received
Procedure RECEIVE_SOLICITATION  //Request MSG is received
if m in in Pendling_List or delivered then
   Digest(Overlapped_Multicast_Vector)
   send Gossip(m) to REQUESTER  //Send the MSG piggybacked with VT
```

Fig. 4 Algorithm Description (Cont'd)

```
Procedure INITIALIZE // Initialing Vector and Local View
Overlapped_Multicast_Vector=(Group1(0,0,0,*,)$_1$,
   Group2(*,0,0,0)$_1$,…):*:not a member, subscript$_1$: epoch$_1$
Local_View={pid}
Procedure SEND_MULTICAST
Gossip_Count=ZERO
for All_Interest_Groups do //Sending Seq.=Sending Seq.+1
   Overlapped_Multicast_Vector(GroupID(pid)) =
         Overlapped_Multicast_Vector(GroupID(pid)+1)
m = (pid, All_Interest_Groups, Gossip_Count,
    Overlapped_Multicast_Vector) //Msg piggybacked with Vector
Unreliable_Multicast(m)  //Using unreliable MCAST
Procedure SEND_JOIN
m = (pid, All_Interest_Groups, Gossip_Count)
Unreliable_Multicast(m)
Procedure SEND_LEAVE
m = (pid, All_Interest_Groups, Gossip_Count, Last_m.Seq, Local_View)
Unreliable_Multicast(m)
Procedure SEND_DIGEST //Select gossip-target from Local View
for All_Interest_Groups and All_Subscribers in
    Overlapped_Multicast_Vector do // Periodically gossip summary
         Digest(Overlapped_Multicast_Vector)
call Procedure SEND_GOSSIP
Procedure SEND_GOSSIP//Select gossip-target from Local View
for each p in Local_View and All_Subscribers s
       with probability rate //Randomly Selecting Small Set
   m.Gossip_Count=m.Gossip_Count+1
   send m including Local_View to p //Gossip about MSG with VT
   send m to s
do Garbage_Collection
Procedure RECEIVE_MULTICAST
if m is not in their interest then //MSG is received but not in interest
   check m.Overlapped_Multicast_Vector and send
Solicit_Retransmission(m)
         //Check the VT whether or not interest MSG is not received
else if m not in Pending_List then //MSG is received and in interest
   put m into Pending_List
call Procedure SEND_GOSSIP
delivery = TRUE
for All_Interest_Groups do     //Check VT of MSG for causal order
   if(m.Overlapped_Multicast_Vector(GroupID(id) >
         p.Overlapped_Multicast_Vector(GroupID(id) and id=p) then
              delivery = FALSE, BREAK
if(delivery = TRUE) then     //If causal order is satisfied
   remove m from Pending_List
   deliver m to APPLICATION //Deliver the MSG to app.
```

Fig. 5 Algorithm Description

4. Performance Evaluation

In this section, we compare average throughput of our protocol based on sensor-broker with that of a previous protocol based on traditional reliable committee [3]. In this comparison, we rely on a set of parameters referred to Bimodal Multicast [4] and LPBCast [7] for gossiping parameters. And we assume that processes gossip in synchronous rounds, gossip period is constant and identical for each process and maximum gossip round is logN. The probability of network message loss is a predefined 0.1% and the probability of process crash during a run is a predefined 0.1% using UDP/IP. The group size of each sub-figure is 32(2), 64(4), 128(8) and 256(16).

Figure 6 shows the average throughput as a function of perturb rate for various group sizes. The x-axis is the group size (the number of overlapping groups) and the y-axis is the number of messages processed in the perturb rate, (a)20%, (b)30%, (c)40% and (d)50%. In the four sub-figures from 6(a) to 6(d), the average throughput of causally ordered delivery protocol based on sensor-broker is not a rapid change than that of the protocol based on traditional reliable committee. Especially, the two protocols are compared to each other in terms of scalability by showing how the number of messages required for maintaining membership list in perturbed networks with processes join and leave. The proposed sensor-broker protocol is more scalable because the brokers are selected by peer-sampling services [15, 20] and all messages including join and leave are gossiped by them.

And then you compare the message overhead of our protocol based on sensor-broker with that of the previous protocol based on local views [16]. We consider sensor nodes and query nodes. While sensor nodes are stationary, query nodes are mobile. The query node periodically, every 60s, sends it query and the sensors publish their current value every 50s(this value is able to be varied). We do the cell size could be set to 300m and the default AoI contains roughly 40 to 50 sensors with a total number of 600 sensors in coordinates 600<=x,y<=900. In the four sub-figures from 7(a) to 7(d), the message overhead of our protocol based on sensor-broker is more slightly lower than that of the previous protocol based on local views [16]. We can evaluate the effects of the query node mobility and scalability with respect to the number of sensors and query nodes. We can see message overhead of sensor-broker protocol increases fast for low numbers of query nodes but then the increase diminishes with increasing number of query nodes. It shows that the sensor coverage fluctuates to some degree. But with a high number of query nodes, it becomes more likely that a broker to which a query node sends a subscription already has an active subscription. Therefore the increase in the

message overhead eventually diminishes once all the subscriptions and update messages are no longer sent to the query nodes. In contrast, the message overhead of the protocol based on local views [16] sub-linearly increases with the number of query nodes. This is an attractive indication for the scalability and mobility of sensor-broker protocol because the message overhead increases with increasing number of query nodes and the sensor coverage is nearly the same for the numbers of sensor nodes.
However it is not always so, approached from a study in A.-M. Kermarrec [12]. Especially, online social networks are still growing regularly by the day. These networks constitute huge live platforms that are exploited in many ways. And it is clearly appealing to perform large-scale general purpose computations on such platforms and one might be tempted to use a central authority for that, namely one provided by the company orchestrating, likely the broker coordinating. Yet, this poses several privacy problems. In [12], they argue that a decentralized approach where the participants in the social network keep their own data and perform computations in a distributed fashion without any central authority. So it depends on the user applications which approach between sensor-broker and local views is more preferable.

5. Conclusions

In this paper, we present a causal order guaranteeing multicast protocol. The protocol based on sensor-brokers(publisher-brokers) as delegates periodically gossips about the messages in overlapping multicast groups and guarantees causally ordered delivery of the messages. In the protocol based on sensor-broker, some sensors might lead to make overlay multicast groups and query nodes subscribe to the corresponding sensor-brokers networks of their interest and receive the results of the queries, that is, aggregated messages of the information. The vector information of each interesting group piggybacked with every multicast message for causally ordered delivery are sent or received periodically using gossip-based protocols by sensor-brokers. So, these features of this protocol might be that causally ordered delivery properties by sensor-brokers are the same as those properties by traditional committee, but result in its very low communication cost compared with the cost incurred by the traditional committee because of gossip-style disseminations. And for future works, we clearly show the pros and cons according to applications by comparing two different protocols, the sensor-broker based one and the fully decentralized one based on local views.

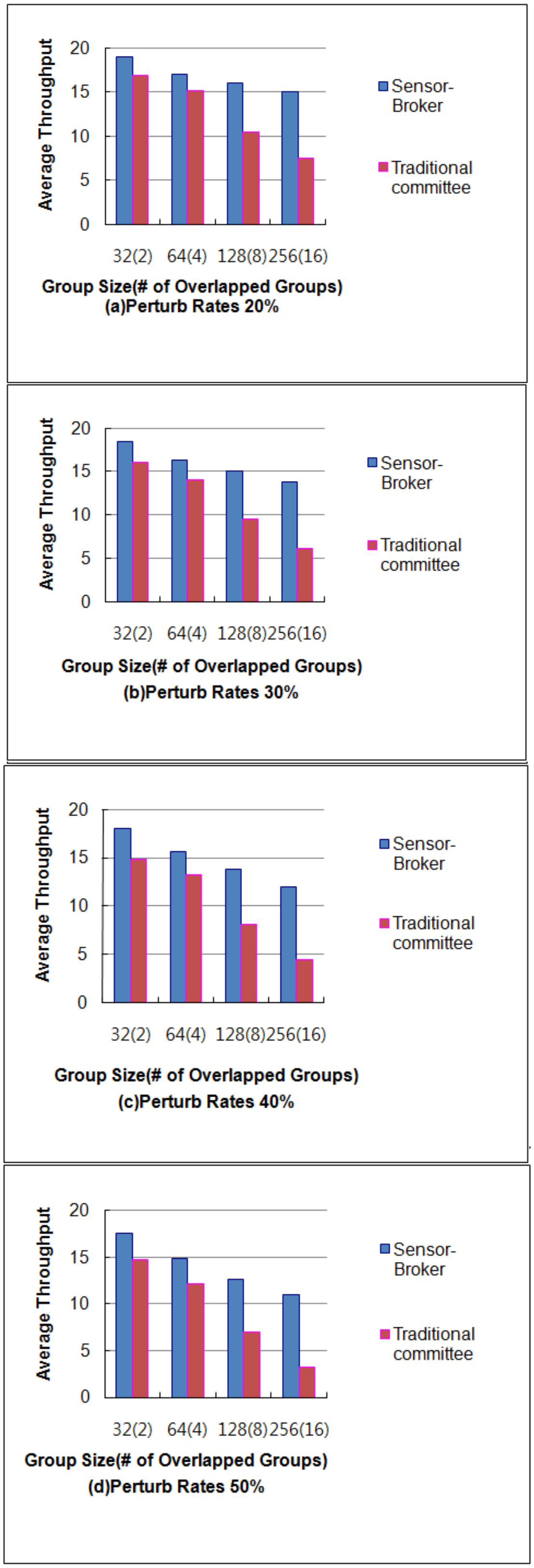
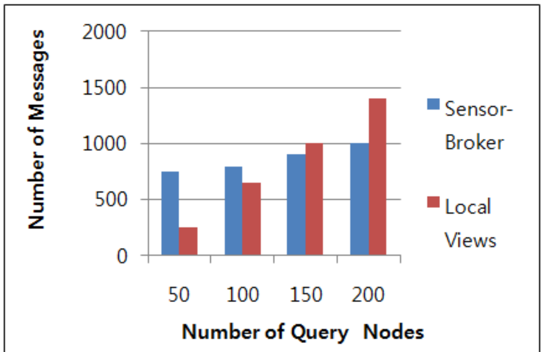
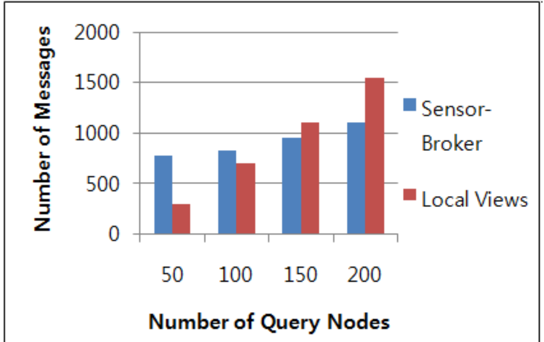


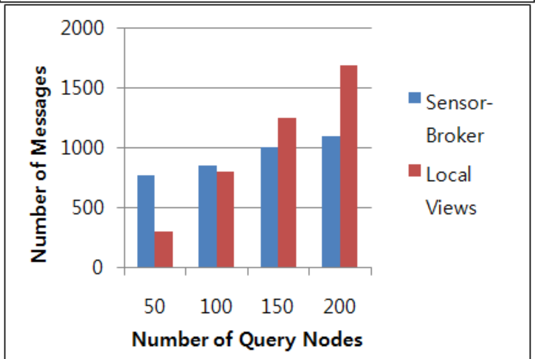
Fig. 6 Average Throughput by Perturb Rates



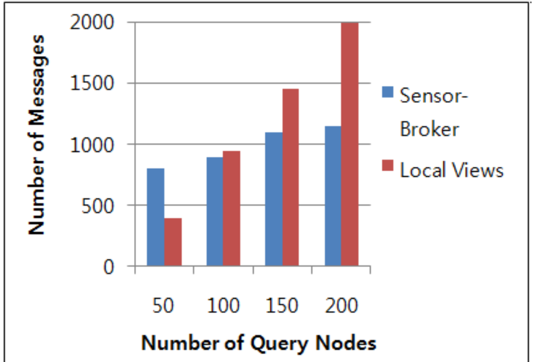
(a)Perturb Rate 20%



(b)Perturb Rate 30%



(c)Perturb Rate 40%



(d)Perturb Rate 50%

Fig. 7 Message Overhead by Query Nodes

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on Sensor Networks”, IEEE Communications Magazine. Vol. 40, No. 8, 2002, pp. 102-114.

[2] X. Bai, M. Bertier, R. Guerraoui, A.-M. Kermarrec, and V. Leroy, “Gossiping Personalized Queries”, in Proceedings of 13th International Conference on Extending Database Technology, Lausanne, Switzerland, Mar. 2010, pp. 87-98.

[3] K. Birman, A. Schiper, and P. Stephenson, “Lightweight Causal and Atomic Group Multicast”, ACM Transactions on Computer Systems. Vol. 9, No. 3, 1991, pp. 272-314.

[4] K. Birman, M. Hayden, O. Ozkasap. Z. Xiao, M. Budiu, and Y. Minsky, “Bimodal Multicast”, ACM Transactions on Computer Systems. Vol. 17, No. 2, 1999, pp. 41-88.

[5] D. Culler, D. Estrin, and M. Srivastava, “Overview of Sensor Networks”, IEEE Computer, Vol. 37, No. 8, 2004, pp. 41-49.

[6] P. Eugster, and R. Guerraoui, “Probabilistic Multicast”, in Proceedings of the 2002 International Conference on Dependable Systems and Networks, Vienna, Austria, Jun. 2002, pp. 313-324.

[7] P. Eugster, R. Guerraoui, S. Handurukande, P. Kouznetsov, and A.-M. Kermarrec, “Lightweight probabilistic broadcast”, ACM Transactions on Computer Systems, Vol. 21, No. 4, 2003, pp. 341-374.

[8] P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec, “The many faces of Publish/Susbscribe”, ACM Computing Surveys, Vol. 35, No. 2, 2003, pp. 114-131.

[9] P. Eugster, “Atomic Probabilistic Broadcast”, EPFL, IC_TECH_REPORT_200303.

[10] P. Felber, and F. Pedone, “Probabilistic Atomic Broadcast”, in Proceedings of 21st IEEE Symposium on Reliable Distributed Systems (SRDS'02), Osaka, Japan, Oct. 2002, pp.170-179.

[11] D. Freedman, Ken. Birman, K. Ostrowski, M. Linderman, R. Hillman, and A. Frantz, “Enabling Tactical Edge Mashups with Live Objects”, in Proceedings of the 15th International Command and Control Research and Technology Symposium(ICCRTS '10), Information Sharing and Collaboration Processes and Behaviors Track. Santa Monica, CA, USA, Jun. 2010, (Best Paper in Track; Best Paper in Conference.)

[12] A. Giurgiu, R. Guerraoui, K. Huguenin, and A.-M. Kermarrec, “Computing in Social Networks”, in Proceedings of 12th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS), New York, USA, Sep. 2010, LNCS Vol. 6366, pp.332-346.

[13] R. Guerraoui, and A. Schiper, “Genuine Atomic Multicast in Asynchronous Distriubted Systems”, Theoretical Computer Science, Vol. 254, Issue 1-2, 2001, pp. 297-316.

[14] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed diffusion: A scalable and robust communication paradigm for sensor networks”, in Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCOM '00), Boston, MA, Aug. 2000, pp. 56-67.

[15] M. Jelasity, S. Voulgaris, R. Guerraoui, and A.-M. Kermarrec, and M. Steen, “Gossip-based Peer Sampling”, ACM Transactions on Computer Systems, Vol. 25, No. 3, 2007, pp. 1-36.

[16] C. Kim, and J. Ahn, “Decentralized Multi-Group Communication Protocol Supporting Causal Order”, in

Proceedings of the Second International Conference on Communication Software and Networks, 2010(ICCSN'10), Singapore, Feb. 2010, pp. 444-448

[17] J. Lifton, M. Laibowitz, D. Harry, N.-W. Gong, M. Mittal, J. and A. Paradiso, "Metaphor and Manifestation—Cross-Reality with Ubiquitous Sensor/Actuator Networks", IEEE Pervasive Computing, Vol. 8, No. 3, 2009, pp. 24-33.

[18] C. Meesookho, S. Narayanan, and C. S. Raghavendra, "Collaborative Classification Applications in Sensor Networks", in Proceedings of Sensor Array and Multichannel Signal Processing Workshop, Rosslyn, USA, Aug. 2002, pp. 370-374.

[19] L. Peterson, N. Buchholzand, and R. Schlichting, "Preserving and using context information interprocess communication", ACM Transaction Computer Systems, Vol. 7, No. 3, 1989, pp. 217-246.

[20] S. Pleisch, and K. Birman, "SENSTRAC: Scalable Querying of SENSor Networks from Mobile Platforms Using TRACking-Style Queries", International Journal of Sensor Networks. Vol. 3, Issue 4, 2008, pp. 266-280.

[21] G. Pottie, and W. Kaiser, "Wireless Integrated Network Sensors", Communications of the ACM, Vol. 43, No. 5, 2000, pp. 51-58.

[22] L. Rodrigues, R. Baldoni, E. Anceaume, and M. Raynal, "Deadline-Constrained Causal Order," in Proceedings of Third IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, 2000, pp. 234-243.

[23] N. Tolgyesi, and M. Jelasity, "Adaptive Peer Sampling with Newscast", in Proceedings of Euro-Par 2009, LNCS, vol. 5704, Delft, the Netherlands, Aug. 2009, pp. 523-534.

[24] Y. Vigfusson, K. Birman, Q. Huang, and D. Nataraj, "GO:Platform Support For Gossip Applications", in Proceedings of IEEE Ninth International Conference on Peer-to-Peer Computing(P2P 2009), Las Vegas, USA, Jan. 2009, pp.222-231.

[25] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey", Computer Networks, Vol. 52, Issue 22, Aug. 2008. pp. 2292-2330.

CHAYOUNG KIM B.S. and M.S. degrees from the Sookmyung Women's University, Seoul, Korea, in 1996 and 1998, respectively and Ph.D. degree from the Korea University in 2006. From 2005 to 2008, she was a senior researcher in Korea Institute of Science and Technology Information, Korea, where she has been engaged in National e-Science of Supercomputing Center. Since 2009, she has been a researcher at Contents Convergence Software Research Center in Kyonggi University, Korea. Her research interests include distributed computing, group communications and peer-to-peer computing.

JINHO AHN(Corresponding author) received his B.S., M.S. and Ph.D. degrees in Computer Science and Engineering from Korea University, Korea, in 1997, 1999 and 2003, respectively. He has been an associate professor in Department of Computer Science, Kyonggi University. He has published more than 70 papers in refereed journals and conference proceedings and served as program or organizing committee member or session chair in several domestic/international conferences and editor-in-chief of journal of Korean Institute of Information Technology and editorial board member of journal of Korean Society for Internet Information. His research interests include distributed computing, fault-tolerance, sensor networks and mobile agent systems.

---

[*] Corresponding author: Tel.: +82 31 249-9674