International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.5 (2014), pp.49-62
http://dx.doi.org/10.14257/ijmue.2014.9.5.05

Broadcast Protocol Guaranteeing Causal Delivery Order Consistency Condition for Social Networks

Chayoung Kim and Jinho Ahn*

Dept. of Comp. Scie., Kyonggi Univ., Iuidong, Yeongtong, Suwon 443-760 Gyeonggi, Republic of Korea
{kimcha0, jhahn}@kgu.ac.kr

Abstract

The social networks are based on sensor networks by mobile phones where social applications, such as real-time collaborative video watching applications, work out well, sensing their events collaboratively and socially. Recently, these networks use a content-based distributed P/S (publish/subscribe) infrastructure that acts as a scalable group communication backbone. P/S based gossip protocols is elastically to scale in and out and provides suitable consistency guarantees for data safety and high availability but, does not deal with end-to-end message delay and message order-based consistency, which are addressed in real-time collaborative applications. Especially in broadcasting in real-time collaboration applications, it is possible for the messages to take different time and arrive at end users in different order. So, these applications should be based on P/S infrastructure including dealing with message timeliness and message ordering consistencies. Gossip communication is becoming one of the promising solutions for addressing P/S scalability problems in providing information propagation functionality by exploiting a mixture of diverse consistency options. In this paper, we present deadline-constraints causal order protocol respecting Δ(lifetime) based on P/S architecture for broadcasting in real-time collaborative applications in social networks to guarantee causally ordered messages delivery from brokers to subscribers. In the proposed protocol, every broker manages a 2-dimensional vector, representing its knowledge of the last message sent by each broker at time t. But, every broker disseminates a broadcast message only with one scalar variable, the time-stamped information that represents the maximum gossip round and is the deadline (lifetime) of the immediate message of it, to subscribers because all messages disseminated by brokers have the same lifetime as the maximum number of gossip rounds. Therefore, the proposed protocol implemented for P/S based on gossip protocols results in very low communication overhead from brokers to subscribers in the context of broadcast respecting Δ.

Keywords: *Publish/Subscribe, broadcast, reliability, scalability, deadline-constraints causal order*

1. Introduction

There is an explosion in the amount of mobile phones generating videos and videos shared on-line especially on social networks [7]. Also, multiple sensors on a mobile phone have become more real-life sensors and this makes sensor networks ubiquitous

* Corresponding author: Tel.:+82-31-249-9674; Fax:+82-31-249-9673.

environments for us to live in [2]. Sensor networks collaboratively detect events and make attempts to extend a simple communication to the social context-sharing by using mobile phones especially on social networks [7]. So, we are interested in the social networks based on sensor networks where social applications work out well, sensing their events collaboratively and socially. Many social applications such as real-time collaborative video watching application [1] should allow people to collaborate in real-time around the same videos played on their sensor network infrastructure, their different locations and their different types of devices. Also, Social TV [8] needs a service infrastructure, which is social networks giving social group communication service. The recent studies [8] provide access to arrange interactive mobile system of socials services, which use the network and computational infrastructure by social widgets. So, the social network based on sensor networks should provide an adequate communication infrastructure to alleviate hot spots and to elastically scale in and out for better exploiting network and computational resources. Recently, these networks use a content-based distributed publish/subscribe infrastructure that acts as a scalable group communication backbone, such as ASIA [6].

In ASIA [6], the content-based P/S(publish/subscribe) architecture builds an overlay network on top of the physical network to provide resilience against failures and overloaded brokers, such as multi-path routing of messages [5]. Multi-path routing is one of the solutions of space redundancy for circumventing the failed node. So, path redundancy is an appealing solution to architect reliable publish/subscribe middleware with timeliness constraints, however, providing path diversity is still a challenging issue, such as network diversity [5]. So, in our proposed protocol, P/S architecture is based on gossip protocols, which seem more appealing in many P/S systems because they are more scalable than traditional reliable broadcast. In gossip protocols, each process exchanges periodically its history of the received notifications with randomly chosen members. So, gossip protocols are one of the temporal redundancy providing techniques, which allows a publish/subscribe middleware to be defined timely that means, timeliness: given a certain time deadline, Δ, all non-faulty subscribers have been notified of a published event before Δ expires [5]. On the other hands, P/S based on gossip protocols is providing suitable consistency guarantees for current social applications, such as replication for data safety and high availability but, does not deal with end-to-end message delay and message order-based consistency, which are addressed in real-time collaborative applications [1]. In real-time collaborative applications based on social networks using mobile phones, the synchronization messages may take different time to arrive and these message timeliness may impact the video synchronization. So, these applications should be based on middleware infrastructure including dealing with message delays and message ordering consistencies. If P/S architecture for social networks could deal with message timeliness and ordering consistencies, real-time collaborative applications, such as [1] might focus on synchronizing such video playback on multiple devices with different playback engines and network bandwidth.

Therefore, in this paper, we present deadline-constraints causal order protocol based on scalable P/S architecture for social networks, such as [6] to guarantee causally ordered messages delivery of collaborative broadcast messages from brokers(providers) to subscribers, especially for broadcast respecting Δ(a certain time deadline) for a real-time collaborative video watching application[1]. Our proposed protocol could extend deadline-constraints causal order respecting Δ [9] in the context of scalable distributed P/S architecture for social networks.

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 [9]. To prevent causal order violation, either message might be forced to wait for messages in their past or late messages might have to be discarded [9]. For real-time collaborative applications [1] for social networks, the first approach is not suitable since when a message has bypassed its deadline, all messages that causally depend on it might be forced to bypass their deadlines. In real-time collaborative applications, such as [1], it makes more sense to allow messages bypassing their deadlines to be dropped than to force many other causality related messages to bypass their deadlines [9]. So, our proposed protocol is based on gossip protocols giving preferences to local members to significantly reduce the number of messages traversing the long-distance network links. So, in gossip protocols, it is turned out that reducing the long traversing makes reducing messages bypassing their deadlines. And gossip protocols are adequate for large scale settings to build an overlay allowing completely decentralized solutions, and easily deployable to scale in and out.

In our proposed protocol, because every broker knows about each other, it manages a 2-dimensional vector like in the protocol [9], representing its knowledge that the last message sent by a broker x to broker y has been sent at time t. On the other hand, every broker disseminates the broadcast message including a scalar variable, whose size is 1 for one number, *i.e.*, the time-stamped information that represents the maximum gossip round, which means its deadline(lifetime), to subscribers by using gossips. In between brokers, because collaborative broadcast messages are based on IP-Multicast and gossip protocols to ensure bimodal delivery [3] to all interested brokers, their messages have their unique deadlines for each collaborative applications. But, from brokers to subscribers, because all messages are based on P/S using gossip protocols and dependent on each periodic gossip round in which only one member can generate and send a message and the maximum number of gossip rounds is deadline(lifetime), all messages disseminated by brokers have the same lifetime as the maximum number of gossip rounds. And the time-stamp of the generated gossip round is as same as the one of the maximum gossip round because every message is dropped when the deadline expired. Each gossip round can be characterized as a unique notation represented in colors. So, the time-stamped information is represented in terms of colors. If two messages A and B are generated in different gossip rounds respectively, they can be represented in two different colors. The maximum gossip rounds, the deadline represented in colors as the lifetime of the immediate message is piggybacked on each broadcast message and transmitted to subscribers in order for the subscribers to verify the observation of causal ordering relation among all messages which sensor brokers have received or sent before. Therefore, the proposed protocol implemented for P/S paradigms based on gossip protocols results in very low communication overhead from brokers to subscribers in the context of deadline-constrains causal order broadcast respecting Δ because all messages sent by brokers have the same deadline and the subscriber group has changed infrequently.

2. The Proposed Protocol

2. 1. Basic Idea

The proposed protocol, in between brokers, guarantee causally ordered delivery, respecting deadline-constraintscausal order using 2-dimensional vector representing the knowledge for a broker to know the last message sent by i to j has been sent at time t, like in the protocols of Rodrigues *et al.* [9]. In the protocol [9], every message has a unique lifetime known as deadline. The brokers might aggregate the reporting information based on the application subscribers' needs, while guaranteeing the causally ordered delivery of messages. The subscribers receive the aggregated information from their chosen brokers by gossip-style disseminations. But, from brokers to subscribers, all messages disseminated by brokers have the same lifetime (deadline) as the maximum number of gossip rounds because all messages are dependent on each periodic gossip round in which only one member can generate and send a message. And in our proposed protocol, especially for broadcasting TV, the subscriber group has changed little during the communication. So, each gossip round for one subscriber group can be characterized as a unique notation represented using colors. So, the time-stamped information is represented in colors. Also, if a message m has bypassed its deadline and if its delivery violates causal order, then it should be discarded. The time-stamp of the generated gossip round is as same as the one of the maximum gossip round because every message is dropped when the deadline is. So, the time-stamp represented in a color of the generated gossip round is as same as the one of the maximum gossip round. Therefore, only the time-stamp of the immediate message represented in colors of the lifetime is also piggybacked on each broadcast message and is transmitted to subscribers. That is, the proposed protocol needs a scalar, which size is 1 for one number, because one color of the lifetime represents the deadline of the latest messages of each broker. So, our proposed protocol is very scalable because of low overhead from brokers to subscribers.

2. 2. Algorithm Description

In this section, we describe our proposed protocol through an example of Figure 1, which shows how in detail each broker generates and aggregates broadcast messages and causally ordered delivery information, and an example of Figure 2 and 3, which shows how messages and information are disseminated from brokers to subscribers.

The proposed protocol respects deadline-constraints causal order using 2-D vector between brokers, like in the protocols of Rodrigues *et al.* [9]. As an example, in figures 1 and 2, 2-D vector describes the causal past of the message each broker i sends and receives for causal order and 2-D $\text{vector}_i[x,y]=t$ describes broker i knows the last message sent by broker x to broker y has been sent at time t. Figure 4 and figure 5 show the proposed protocol respecting deadline-constraints causal order, which introduces a new protocol based on P/S paradigms between brokers and subscribers. The example of figure 1 shows how each broker = {A, B, C, D} is participating in BrokerGroup1 = {A, B, C} and BrokerGroup2 = {A, B, D}. In this proposed protocol, when broker i sends a message, it associates a broadcast message m with a specific deadline (deadline_m) and updates the entries of the array 2-D $\text{vector}_i[i,j]$ corresponding to all the destination brokers.

Let us first a simple description of the protocol shown in figure 4 (lines MB1-MB2). $\forall(x,y)$, 2-D $\text{vector}_i[x, y]$ is stored in 2-D vector_m and the broadcast message m with 2-D vector_m is sent for all destination brokers. So, if broker j is one of the destinations, then 2-D $\text{vector}_i[i,j]$ is updated to the sending time of m, t_m. So, we use $m \rightarrow m' \Leftrightarrow$2-D $\text{vector}_m <$2-D $\text{vector}_{m'}$ according to this approach, like in the protocol Rodrigues *et al.* [9].

In this proposed protocol, when a broker sends and receives broadcast messages, there are two cases, 1) the messages might be delivered to the application layer or 2) the messages might have to be discarded. In the proposed protocol shown in figure 5, if m, which are sent by j and received by i, has bypassed deadline$_m$ and if its delivery violates causal order, then it should be discarded. Let us a description of the protocol shown in figure 5 (line RMB2) for 1) delivering a message. If the predicate Del_ok ≡ ((2-D vector$_i$[j,i] <2-D vector$_m$[j,i]) ∧ (∀x≠j : (2-D vector$_m$[x,i] ≤ 2-D vector $_i$[x,i])) is true, that means its delivery does not violate causal order, then m can be delivered, like in the protocol Rodrigues et. al[9]. Also, if the predicate logical_deadline$_m$ ≡ (current_time = min({Deadline_arr_succ$_m$})) is true, then m can be delivered, like in the protocol Rodrigues et. al[9]. This is the case of message m$_3$ arriving at destination D in Figure 1. It means m$_1$→m$_3$⇔2-D vector$_{m1}$< 2-D vector$_{m3}$ and m$_3$ has arrived and isn't yet delivered at destination D not receiving m$_1$. And, if the predicate logical_deadline$_m$>min({Deadline_arr_succ$_m$})) is true, then m can be delivered. This is the case of destination D delivering m$_2$ and m$_3$ to the application layer, depicted in figure 1. In this case, m$_3$ must be delivered before deadline$_{m4}$ in order not to violate deadline-constraints causal order. As an example, upon the arrival of message m$_4$ at destination D in Figure 1, the logical deadline of message m$_2$ becomes deadline$_{m3}$.

Let us a description of the protocol shown in Figure 5 (line RMB1) for 2) discarding a message. If the predicate too_late ≡ (deadline$_m$<current_time) or Del_viol_CO ≡ (2-D vector$_m$[j,i] ≤ 2-D vector$_i$[j,i]) is true, then m is discarded. The predicate too_late is the case of m$_1$ arriving at destination D in figure 1. On receiving message m$_4$, destination D delivers m$_2$and m$_3$ to the application layer and 2-D vector$_d$[i,d] is updated to the sending time of m$_3$in order not to violate deadline-constraints causal order. The predicate Del_viol_CO is this case, which means, if the delivery of m$_1$ violates causal order, m$_1$ should be discarded.

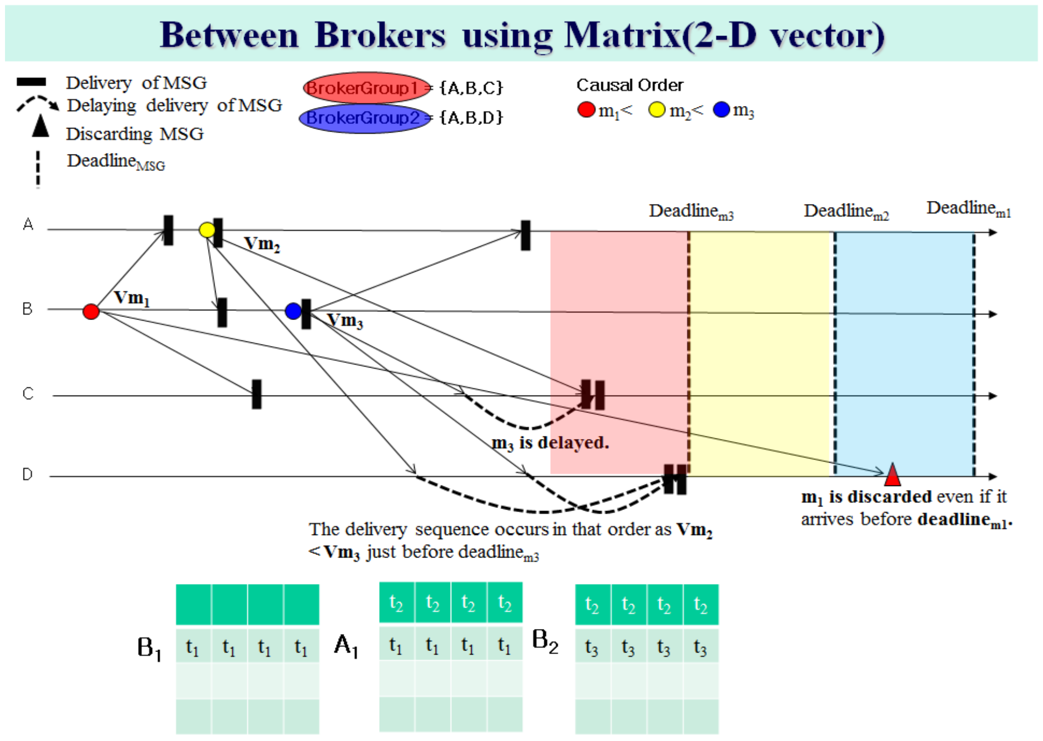
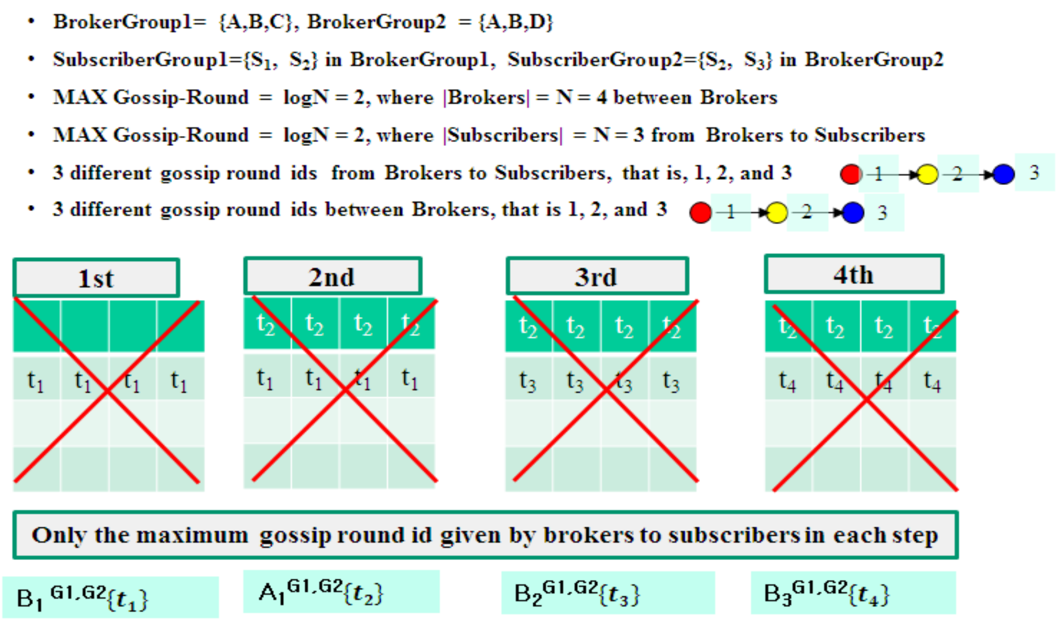


Figure 1. 2-D(Dimensional) Vector for Causal Order Between Brokers

This example of Figure 2 shows how in detail each broker aggregates the information of causally ordered delivery sent to subscribers. In general, gossip protocols [4] take O(logN) rounds to reach all nodes, which are not fault, where N is the number of nodes. The proposed protocol is based on gossip protocols like as an environment of [4] from brokers to subscriber, where O(logN) maximum gossip rounds is deadline(lifetime). As an example, in Figure 2, there are the two broker groups, BrokerGroup1={A, B, C} and BrokerGroup2={A, B, D}, the two subscriber groups, SubscriberGroup1={S_1,S_2} participating in BrokerGroup1 and SubscriberGroup2={S_2,S_3} participating in BrokerGroup2, where N is the number of subscribers and logN=2 is the maximum number of gossip rounds, that is, deadline. In our proposed protocol, each broker has to send and receive each message with a 2-D vector for causal past using IP-Multicast and periodic gossips [4], respecting deadline-constraints causal order [9]. On the other hand, every broker disseminates the broadcast message including one scalar variable, the time-stamped information that represents the maximum gossip round of the immediate message, which means its deadline, to subscribers by gossiping. If message m disseminated by a broker has bypassed, deadline_m, then m should be discarded [9]. So, the time-stamp of the generated gossip round is as same as the one of the maximum gossip rounds. Therefore, the proposed protocol implemented for P/S paradigms based on gossip protocols results in very low communication overhead from brokers to subscribers in the context of respecting deadline-constraints because all messages sent by brokers have the same deadline and especially for broadcasting, the subscriber group has changed infrequently. In our proposed protocol, each gossip round can be characterized as a unique notation represented using color. This proposed protocol needs logN + α colors because the maximum number of gossip rounds is logN and α may be application specific for buffering. As an example, in Figure 3, it needs 3 colors because the deadline is logN+1=3. So, the order in which they appear is as follows; a red, yellow, and blue stand for each gossip round and is the repetition in the maximum gossip round.


- BrokerGroup1= {A,B,C}, BrokerGroup2 = {A,B,D}
- SubscriberGroup1={S_1, S_2} in BrokerGroup1, SubscriberGroup2={S_2, S_3} in BrokerGroup2
- MAX Gossip-Round = logN = 2, where |Brokers| = N = 4 between Brokers
- MAX Gossip-Round = logN = 2, where |Subscribers| = N = 3 from Brokers to Subscribers
- 3 different gossip round ids from Brokers to Subscribers, that is, 1, 2, and 3
- 3 different gossip round ids between Brokers, that is 1, 2, and 3

Only the maximum gossip round id given by brokers to subscribers in each step

Figure 2. An Example of One Scalar Variable for the Maximum Gossip Rounds of the Immediate Message

Let us explain our proposed protocol in Figure 4 and 5 and an example of Figure 3. Each broker has to manage the whole set of these vectors, 2-D vector between brokers. On the other hand, every broker disseminates the broadcast message with deadline_color$_m$ from brokers to subscribers. In Figure 3, the deadline of the last sent message sent by B is red. So, this message should be delivered before the immediate oncoming red. In this example, in the first round, broker B generates the first broadcast message and the current gossip round color is "red", denoted "red$_{B1}$". Let us explain the protocol shown in Figure 5 (lines MS1-MS4). The current_color is send_color$_m$ and send_color$_m$ is deadline_color$_i$ and deadline_colo$_m$. The broadcast message m with deadline_colo$_m$ is disseminated for all destination subscribers for deadline-constraints causal order. So, if subscriber j is one of the destinations, then subscriber i is updated as the sending group round, send_color$_m$ = current_color. So, we use m→m'⇔deadline_color$_m$<deadline_color$_{m'}$ according to this approach. In the second round, broker A generates the broadcast message and the current gossip round color is "yellow", denoted "yellow$_{A1}$". In the third round, B generates the broadcast message and the current gossip round color is "blue", denoted "blue$_{B2}$". And after the end of the first maximum gossip round, the order in which they appear is as follows; red, yellow, and blue is restarting. So, in the fourth round, broker B generates the broadcast message and the current gossip round color is "red", denoted "red$_{B3}$".

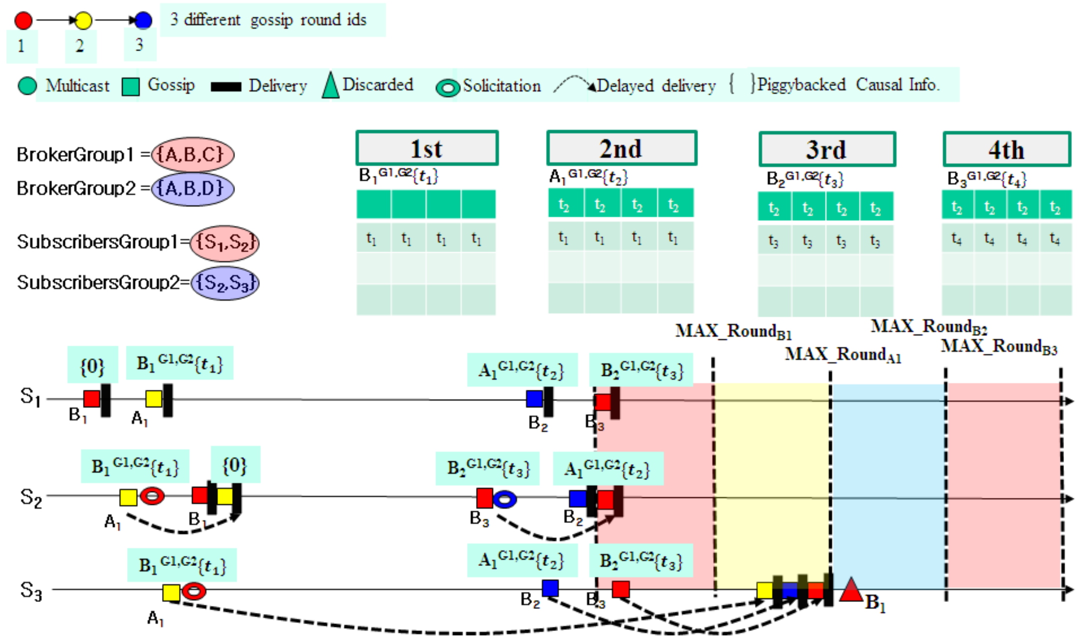


Figure 3. An Example of Messages with One Scalar Variable from Brokers to Subscribers

In this proposed protocol, when a broker disseminates broadcast messages to its subscribers, there are two cases, 1) the messages might be delivered or 2) the messages might have to be discarded. Let us explain the protocol shown in Figure 5 (line RMS2) for 1) delivering a message and an example in Figure 3. This is the case of message red$_{B3}$ arriving at subscriber S$_3$. If the predicate Del_ok ≡ (deadline_color$_m$>deadline_color$_i$)∨logical_deadline$_m$ ≡

(current_color = min({Deadline_arr_succ$_m$}) ∧deadline_color$_{min}$=current_color$_{min}$) is true. This is the case of message red$_{B3}$ arriving at subscriber S$_3$ not receiving B$_1$. In the current red round, message red$_{B3}$ should be delivered because the color of its deadline round is as same as the color of the current round. And also, if the predicate (logical_deadline$_m$>min({Deadline_arr_succ$_m$})) is true, message m should be delivered. This is the case of deliveries of yellow$_{A1}$ and blue$_{B2}$ at subscriber S$_3$. Upon the deadline of message red$_{B3}$ at subscriber S$_3$, the logical deadline of messages yellow$_{A1}$ and blue$_{B2}$ becomes the deadline of red$_{B3}$ in order not to violate causal order. As an example, upon the arrival of message m$_4$ at destination D, the logical deadline of messages m$_2$ and m$_3$ becomes deadline$_{m4}$.

For 2) discarding a message, when deadline_color$_m$≤ deadline_color$_{m'}$, at subscriber S$_i$, deadline-color$_{m'}$ has arrived and deadline_color$_m$ is not delivered to the application layer yet. This means that if the two messages are represented in the same color, then the one, m, was generated in |logN+1|-rounds ahead of the gossip round of the other, m’. So, m has bypassed deadline$_m$ and if its delivery violates causal order, then it should be discarded. In figure 3, this is the case of message red$_{B1}$ arriving at subscriber S$_3$. On receiving message red$_{B3}$, subscriber S$_3$ delivers yellow$_{A1}$, blue$_{B2}$ and red$_{B3}$ to the application layer in order not to violate deadline-constraints causal order. So, if red$_{B1}$ has bypassed deadline=|logN+1|=3, red$_{B1}$ should be discarded because its delivery violates causal order. Let us explain the protocol shown in Figure 5 (line RMS1) for 2) discarding message red$_{B1}$. If the predicate too_late(deadline_color$_m$<current_color) or Del_viol_CO ≡ (deadline_color$_m$≤ deadline_color$_i$) is true, then m is discarded. The case of red$_{B1}$ is that the predicate too_late is true. Message red$_{B1}$ was generated in the gossip round, |logN+1| rounds, ahead of the current gossip round and it has bypassed deadline=|logN+1|=3. And also, the latest message for all destinations is updated after deliveries of yellow$_{A1}$, blue$_{B2}$ and red$_{B3}$ in order not to violate deadline-constraints causal order. So, red$_{B1}$ should be discarded.

```
Procedure multicast(m, $deadline_m$, $Dest_m$) from brokers to brokers
(MB1) $send\_time_m \leftarrow$ current_time;
(MB2) $\forall j \in Dest_m$ do 2-D $vector_i[i,j] \leftarrow send\_time_m$ od;
(MB3) let 2-D $vector_m$ = 2-D $vector_i$ ;
(MB4) $\forall j \in Dest_m$ do send(m, $deadline_m$, 2-D $vector_m$) od;

Procedure gossip(m, $Partial_m$) from brokers to brokers
(GB1) select partial brokers($PartialBk_m$) from local_view;
(GB2) $\forall j \in Partial_m$ do gossip(m, summarize(2-D $vector_i$), $PartialBk_m$) od;

Procedure multicast(m, $send\_color_m$, $Dest_m$) from brokers to subscribers
(MS1) $send\_color_m \leftarrow$ current_color;
(MS2) $\forall j \in Dest_m$ do $deadline\_color_i$, $\leftarrow send\_color_m$ od;
(MS3) let $deadline\_color_m$ = $deadline\_color_i$;
(MS4) $\forall s \in Dest_m$ do send(m, $deadline\_color_m$) od;

Procedure gossip(m, $PartialSub_m$) from brokers to subscribers
(GS1) select partial subscribers($PartialSub_m$) from local_view
(GS2) $\forall j \in (PartialSub_m)$ do gossip(m, summarize($deadline\_color_m$))od;
```

Figure 4. Sending Procedures for each Broker

4. Performance Evaluation

In this section, we compare average throughput of our protocol based on gossip-style dissemination protocols with that of a previous deadline-constraints causal order protocol [9] in which all messages have the same timeliness constraints because of broadcast. In this comparison, we rely on a set of parameters referred to Bimodal Multicast [3] and LPBCast [4] for gossiping parameters. And we assume that processes gossip in periodic rounds, the gossip round is constant and identical for each process and the 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 50, 100, 150, and 200. Figure 6 shows the average throughput as a function of perturb rate for various group sizes. The x-axis is the group size and the y-axis is the average throughput in the perturb rate, (a)20%, (b)30%, (c)40% and (d)50%. In the four subfigures from 6(a) to 6(d), the average throughput of causally ordered delivery protocol based on gossip protocols from brokers to subscribers is not a rapid change than that of the protocol based on traditional reliable broadcast between all members.

```
When one of brokers, $P_i$ receives multicast(m, $deadline_m$, 2-D $vector_m$) from one of brokers, $P_j$ :
let$Deadline\_arr\_succ_m \equiv$ {$deadline_{m'}$ such that $_{m'}$arrived and 2-D $vector_m \leq$ 2-D $vector_{m'}$};
lettoo_late $\equiv$ ($deadline_m$<current_time);
letlogical_$deadline_m \equiv$ (current_time = min({$Deadline\_arr\_succ_m$}));
letDel_viol_CO $\equiv$ (2-D $vector_m[j,i] \leq$ 2-D $vector_i[j,i]$);
let Del_ok $\equiv$ ((2-D $vector_i[j,i]$ <2-D $vector_m[j,i]$) $\wedge$ ($\forall x \neq j$ : (2-D $vector_m[x,i] \leq$ 2-D $vector_i[x,i]$));

(RMB1) if too_late$\vee$Del_viol_CO then discard(m)

(RMB2) else wait (Del_ok$\vee$logical_$deadline_m$) Delivery Condition: DC(m)

(RMB3) $\forall$(x,y) : 2-D $vector_i[x,y] \leftarrow$ max(2-D $vector_i[x,y] \leq$ 2-D $vector_m[x,y]$);

(RMB4) Delivery of m to $P_i$
(RMB5) endif


When one of brokers, $P_i$ receives gossip(m, summarize(2-D $vector_i$), $Partial_m$) from one of brokers, $P_j$ :
(RMB1) $\forall$(x,y) : if (2-D $vector_i[x,y]$ <2-D $vector_m[x,y]$) then Request(m) endif


When one of subscribers, $S_i$ receives multicast(m, deadline_$color_m$) from one of brokers, $P_j$ :
let$Deadline\_arr\_succ_m \equiv$ {deadline_$color_{m'}$ such that $_{m'}$arrived and deadline_$color_m \leq$ deadline_$color_{m'}$ };
lettoo_late(deadline_$color_m$<current_color);
let logical_$deadline_m \equiv$ (current_color = min({$Deadline\_arr\_succ_m$}) $\wedge$deadline_$color_{min}$=current_$color_{min}$);
letDel_viol_CO $\equiv$ (deadline_$color_m \leq$ deadline_$color_i$);
letDel_ok $\equiv$ (deadline_$color_m$>deadline_$color_i$);
(RMS1) if too_late$\vee$Del_viol_CO then discard(m)

(RMS2) else wait (Del_ok$\vee$logical_$deadline_m$) Delivery Condition: DC(m)

(RMS3) $\forall Dest_m$: update deadline_$color_m$;

(RMS4) Delivery of m to $S_i$
(RMS5) endif
```

Figure 5. Receiving Procedures between Brokers and from Brokers to Subscribers

Deadline-constraints causal order for broadcast respecting Δ [9] is less expensive than the protocol for multicast because of having the same timeliness constraints. But, in this case, our proposed protocol and [9] are compared to each other in terms of scalability by showing how many members execute by phases. In perturbed networks with subscribers join and leave, deadline-constraints causal order for broadcast respecting Δ [9] is very expensive because events of sending and receiving messages are governed by all members without distinguishing between brokers and subscribers. On the other hands, the proposed protocol based on P/S is more scalable than the previous protocol [9] because communications between brokers and subscribers are based on gossip-style disseminations and the information are managed only by brokers and the deadline of the information is one scalar variable, depending on the maximum gossip round. So, the management cost of the information is cheaper than that of the previous protocol [9].

We know where the proposed protocol is useful. The approach is more preferable especially for broadcast with hardly probable change of subscriber group membership. In the network layers using some sort of ACK mechanism to ensure reliability, such a use of more information for causal order is very expensive. If all members send and receive ACK in broadcast, ACK implosion has incurred lots of problems, but gossip-style dissemination has no ACK because members periodically gossip about the summary of received messages. And this periodic summary can be used for our proposed causal order protocol. So, we argue that gossip-style dissemination approach outperforms the existing ones in a broadcast group with little change of membership in the context of a distributed fashion. Especially in terms of memory requirements, there are no needs of big memory between brokers and subscribers because subscribers receive aggregated causal order delivery information from them. Also, we think our proposed protocol is more attractive for mobility of subscribers because the message overhead of the mobility depends only on the number of brokers and brokers are more stable than subscribers.

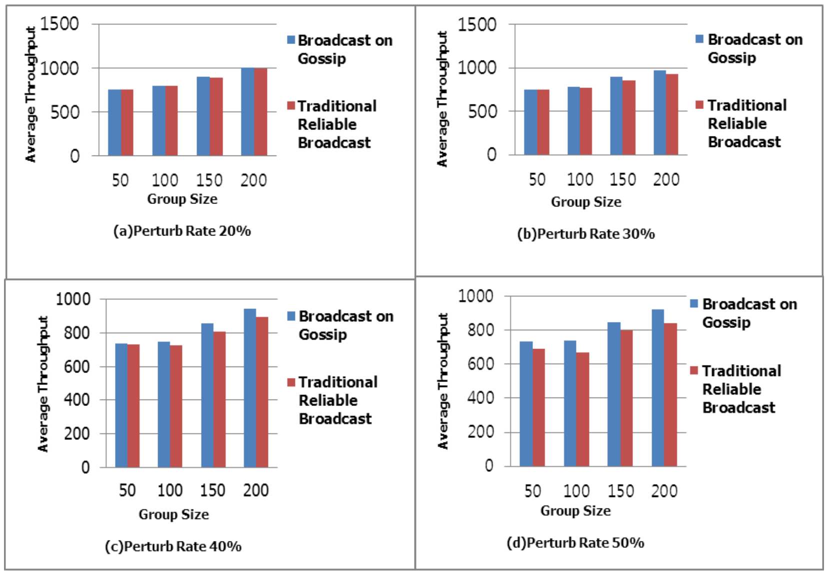


Figure 6. The Average throughput as a Function of Perturb Rate for Various Group Sizes

5. Related Works

Mobicast [7] is a system for mobile live video streaming, allowing for novel collaborative mobile multimedia applications that can enhance the viewing experience of mobile-casted events. MoVi(Mobile Phone based Video Highlights)[2] is exploring “social activity coverage”. This system [2] collaboratively senses the ambience through multiple mobilephones and captures social moments worth recording. Theshort video-clips from different times and viewing angles are stitched offline to form video highlights of the social occasion. Social TV [8] is a general term for technology that supports communication and social interaction in either the context of watching television, or related to TV content. [1] is a mobile application allowing a group of people to collaborate remotely in real-time through watching the same video on their mobile devices. ASIA [6] proposes a content-based distributed publish/subscribe infrastructure for online social web platforms with better capabilities for on-the-fly analytical and data processing. Early work in gossip-style protocol, Birman *et al.* [3] proposes 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 gossip-based retransmissions. But, Lpbcast [4] proposes gossip-style broadcast mechanisms based on a local view instead of a global view. Lpbcast [4] is a completely decentralized protocol because of no dedicated brokers for membership management. To ensure causal message ordering, [9] proposes a novel causal ordering abstraction that takes messages deadlines into consideration for distributed soft real-time applications. In deadline-constrained causal order, each message has its own deadline and, if it arrives on time, never misses its deadline due to preceding messages.

6. Conclusions

In this paper, we present deadline-constraints causal order protocol respecting Δ based on scalable P/S architecture consisting of a cluster of stable brokers for broadcast of real-time collaborative applications in social networks to guarantee causally ordered messages delivery from brokers to subscribers. In between brokers, because collaborative broadcast messages are based on IP-Multicast and gossip protocols, their messages have their unique deadlines. But, from brokers to subscribers, all messages disseminated by brokers have the same lifetime as the maximum number of gossip rounds because all messages are based on P/S using gossip protocols, in which every round is fixed and periodic. So, the maximum number of gossip rounds is the deadline of all messages sent by every broker. From brokers to subscribers, the maximum gossip rounds, the deadline represented in colors as the lifetime of the immediate message is piggybacked on each broadcast message and transmitted to subscribers. And if a message has bypassed its deadline and if its delivery violates causal order, then it should be discarded. The proposed protocol needs one scalar variable because one color of the lifetime represents the deadline of the last message of each broker. So, our proposed protocol is very scalable because of low overhead from brokers to subscribers on P/S paradigms based on gossip protocols in the context of deadline-constraints respecting Δ.

References

[1] A. Attarwala, D. Jagdish and U. Fischer, “Realtime collaborative video annotation using GAE and XMPP”, Cloud Computing (CLOUD), (**2011**), pp. 738 – 739.
[2] X. Bao and R. R. Choudhury, “MoVi: Mobile Phone based Video Highlights via Collaborative Sensing”, In Proceedings of MobiSys’10, (**2010**), pp. 357—370.
[3] K. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu and Y. Minsky, “Bimodal Multicast”, ACM Transactions on Computer Systems, vol. 17, (**1999**), pp. 41—88.
[4] P. Eugster, R. Guerraoui, S. Handurukande, P. Kouznetsov and A. -M. Kermarrec, “Lightweight probabilistic broadcast”, ACM Transactions on Computer Systems, vol. 21, (**2003**), pp. 341—374.
[5] C. Esposito, D. Contronеo and S. Russo, "On reliability in publish/subscribe services", Computer Networks, vol. 57, no. 5, (**2013**), pp. 1318—1343.
[6] D. Eyers, T. Freudenreich, A. Margara, S. Frischbier, P. Pietzuch, and P. Eugster, "Living in the present: on-the-fly information processing in scalable web architectures", In CloudCP, (**2012**).
[7] A. Kaheel, M. El-Saban, M. Refaat, and M. Izz, “Mobicast - A system for collaborative event casting using mobile phones”, in ACM Mobile and Ubiquitous Multimedia - MUM '09, (**2009**).
[8] K. Mitchell, A. Jones, J. Ishmael, and N. J. P. Race, "Social TV: Toward Content Navigation Using Social Awareness", In Proceedings of EuroITV2010, (**2010**), pp. 283—291.
[9] L. Rodrigues, R. Baldoni, E. Anceaume and M. Raynal "Deadline-Constrained Causal Order", 3rd IEEE International Symposium on Object-oriented Real-time distributed Computing, (**2000**), pp. 234 – 241.

Authors

Chayoung Kim, she received 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. From 2009 to 2012, she was a researcher at Contents Convergence Software Research Center in Kyonggi University, Korea. Since 2012, she has been an adjunct professor in Department of Computer Science, Kyonggi University, Korea. Her research interests include distributed computing, group communications and peer-to-peer computing.

Jinho Ahn, he 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 a 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.