International Journal of Multimedia and Ubiquitous Engineering
Vol.8, No.6, (2013), pp.245-256
http:// dx.doi.org/10.14257/ijmue.2013.8.6.25

Gossip-based Causal Order Delivery Protocol Respecting Deadline Constraints in Publish/Subscribe Systems

Chayoung Kim and Jinho Ahn[1]

Dept. of Comp. Scie., Kyonggi Univ., Iuidong, Yeongtong, Suwon 443-760 Gyeonggi, Republic of Korea

{kimcha0, jhahn}@kgu.ac.kr

Abstract

Publish/subscribe systems based on gossip protocols are 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. Especially in real-time collaborative applications, it is possible for the messages to take each a different time to arrive at end users. So, these applications should be based on P/S infrastructure including dealing with message deadlines 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 a new causal order protocol based on scalable P/S architecture for real-time collaborative applications in social web platforms to guarantee causally ordered message delivery, respecting deadline-constraints 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 a certain time. But, every broker disseminates a multicast message with a 1-dimensional vector, the time-stamped information that represents the maximum number of gossip rounds to subscribers because all messages disseminated by brokers have the same deadline as the maximum number of gossip rounds. Therefore, the proposed protocol for P/S based on gossiping results in very low communication overhead from brokers to subscribers in the context of respecting deadline-constraints.

Keywords: *publish/subscribe, group communication, reliability, scalability, deadline-constrained causal order*

1. Introduction

Recently, social web platforms such as Facebook, Twitter, Google+, and LinkedIn have become real-time social communication platforms and embedded themselves in the daily routines of millions of users for people to share a rich set of information, including blog posts, photos, videos, chats, and discussions [5]. As the role of social web platforms shifts from being portals for largely historic data towards providing platforms for real-time data analytics, we can observe that their focus is shifting from the management of past data to the on-the-fly processing of fresh data [5]. And there are researches[5, 6] that the design of scalable architectures for social web platforms focuses on the dissemination, processing and caching of fresh data based on a content-based distributed P/S(publish/subscribe) infrastructure consisting of a cluster of message processing brokers. When new contents, such as blog posts have been submitted to social web platforms, many application services in the platforms require on-the-fly data processing because users expect them to be made available immediately to all interested other users or the platform providers want to perform sophisticated on-the-fly analytical processing of the new data to associate them with matching advertisements [5]. And many social web applications such as real-time collaborative video watching applications [1] should

[1] Corresponding author: Tel.:+82-31-249-9674; Fax:+82-31-249-9673.

ISSN: 1975-0080 IJMUE

allow people to collaborate in real-time around the same video played on their independent network infrastructures, their different locations and types of devices. So, the social web platforms should provide an adequate communication infrastructure to alleviate hot spots and to elastically scale in and out for better exploiting network and computational resources by using a content-based distributed publish/subscribe infrastructure that act as a scalable communication backbone for social web platforms, such as ASIA [5, 6]. In ASIA [6], the content-based 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 [4]. 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 [4]. 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 protocols. 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, which allows a publish/subscribe middleware to be defined timely that means, timeliness: given a certain time deadline Δ, all non-faulty subscribers should be notified of a published event before Δ is expired [4]. On the other hand, 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 [1]. In real-time collaborative applications based on social web platforms, the synchronization messages may take each a different time to arrive and these message deadlines 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 web platforms could deal with message deadlines and ordering consistencies, real-time collaborative applications might focus on synchronizing such video playback on multiple devices with different playback engines and network bandwidth.

Therefore, in this paper, we present a new causal order protocol based on scalable P/S architecture for social web platforms, such as [5, 6] to guarantee causally ordered messages delivery of collaborative multicast messages, respecting deadline-constraints from brokers(providers) to subscribers for real-time collaborative applications, such as [1]. Our proposed protocol extends deadline-constraints causal order protocol [9] in the context of scalable distributed P/S architecture for social web platforms.

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 a 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 web platforms, 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. Gossip protocols have turned out to be adequate for large scale settings to build an overlay allowing completely decentralized solutions and easily to scale in and out and to be deployable.

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 multicast message including a 1-dimensional vector, the time-stamped information that represents the maximum gossip round, which means its deadline, to subscribers only using gossips. In between brokers, because collaborative multicast messages are based on IP-Multicast and gossip protocols to ensure bimodal delivery [2] to all interested brokers, their

messages have their unique deadline 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, all messages disseminated by brokers have the same deadline as the maximum number of gossip rounds. Each gossip round can be characterized as a unique notation represented using color. So, the time-stamped information is represented using colors. If two messages A and B are generated in different gossip rounds respectively, they can be represented in two different colors. The maximum number of gossip rounds, the deadline represented in colors of the lifetime of each message sent by every broker, is also piggybacked on each multicast message and is transmitted to subscribers because the subscribers can verify the observation of causal ordering relation among all messages which sensor brokers have received or sent before the message. That is, the proposed protocol needs a vector, whose size is the number of brokers because one color of the lifetime represents the deadline of the last message of each broker. Therefore, our proposed protocol for P/S systems based on gossiping results in very low communication overhead from brokers to subscribers in the context of respecting deadline-constraints because of the same deadline of all messages.

2. The Proposed Protocol

2.1. Basic Idea

The proposed protocol respects deadline-constrained causal order using 2 dimensional vector representing the knowledge for a broker to know the last message sent by broker x to broker y has been sent at time t, like in the protocols of Rodrigues *et. al*., [9]. 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. In between brokers, their messages have their unique deadline for each collaborative application. 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 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. Each gossip round can be characterized as a unique notation represented using color. So, the time-stamped information is represented using colors. The maximum number of gossip rounds, that means the deadline represented in colors of the lifetime of each message sent by every broker, is also piggybacked on each multicast message and is transmitted to subscribers. That is, the proposed protocol needs a vector, whose size is the number of brokers 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 communication overhead from brokers to subscribers. If a message m has bypassed its deadline and if its delivery violates causal order, then it should be discarded.

2.2. Algorithm description

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

The proposed protocol respects deadline-constrained causal order using MCP_i (Message Causality Past) between brokers, like in the protocols of Rodrigues *et al.*, [9]. As an example, in Figures 1 and 2, MCP_i describes the causality past of the messages a broker i know each other broker have sent and received and $MCP_i[x,y]=t$ describes broker i knows the last message sent by broker x to broker y has been sent at time t. Figures 4 and 5 show the proposed protocol respecting deadline-constrained causally ordered message delivery protocol based on P/S paradigms between brokers and subscribers. The example in Figure 1 shows how each broker = {A, B, C, D} participates in BrokerGroup1 = {A, B, C} and BrokerGroup2 = {A, B, D}. In this proposed protocol, when broker i sends a multicast message

to BrokerGroups 1 and 2, it associates the message with a specific deadline ($deadline_m$) and updates the entry of the array $MCP_i[i,j]$ corresponding to every destination broker j.

Let us show first a simple description of the protocol using figure 4 (lines MB1-MB2). $\forall(x,y)$: $MCP_i[x, y]$ is stored in MCP_m and the multicast message m with MCP_m is sent for all destination brokers. So, if broker j is one of the destinations, $MCP_i[i,j]$ is updated to the sending time of m, t_m. So, we use $m \rightarrow m' \Leftrightarrow MCP_m < MCP_{m'}$ according to this approach, like in the protocol Rodrigues *et. al.* [9].

In this proposed protocol, when a broker receives multicast 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, it should be discarded. Let us show a description of the protocol using Figure 5 (lines RMB2) for delivering a message. If the predicate $Del\_ok \equiv ((MCP_i[j,i] < MCP_m[j,i]) \wedge (\forall x \neq j : (MCP_m[x,i] \leq {}_{MCPi}[x,i]))$ 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 $logical\_deadline_m \equiv (current\_time = \min(\{Deadline\_arr\_succ_m\}))$ is true, then m can be delivered, like in the protocol Rodrigues *et. al.* [9]. This is the case of message m4 arrived at destination D, depicted in Figure 1. It means $m1 \rightarrow m4 \Leftrightarrow MCPm1 < MCPm4$ and m4 have arrived and are not yet delivered at destination D not receiving m1. And, if the predicate $logical\_deadline_m > \min(\{Deadline\_arr\_succ_m\}))$ is true, then then m can be delivered. This is the case of destination D delivering m_2, m_3 and m_4 to the application layer, depicted in Figure 1. In this case, m4 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, depicted in Figure 1, the logical deadline of messages m_2 and m_3 becomes $deadline_{m4}$.

Let us a description of the protocol shown in Figure 5 (lines RMB1) for 2) discarding a message. If the predicate $too\_late \equiv (deadline_m < current\_time)$ or $Del\_viol\_CO \equiv (MCP_m[j,i] \leq MCP_i[j,i])$ 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, m_3 and m_4 to the application layer and $MCP_d[i,d]$ is updated to the sending time of m, t_m, 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.

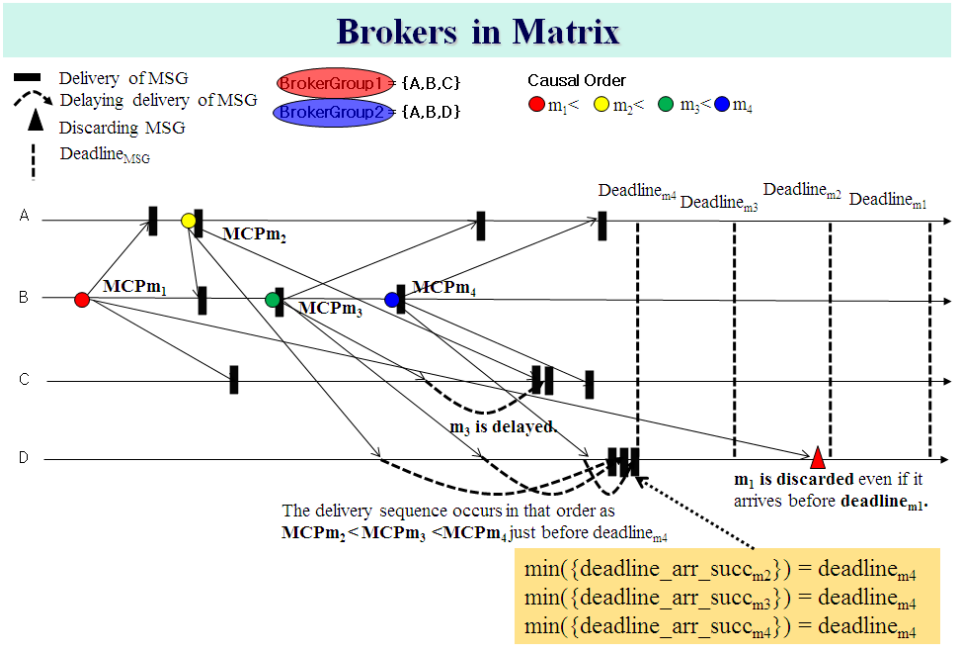

Figure 1. 2 dimensional vector for causal order between brokers

This example of Figure 2 shows how each broker aggregates the information of causally ordered delivery sent to subscribers. In general, gossip protocols [2, 3] take O(logN) rounds to reach all correct nodes, where N is the number of nodes. The proposed protocol is based on gossip protocols like as an environment of [2, 3] from brokers to subscribers, where O(logN) is deadline(lifetime). As an example, in figure 2, there are two broker groups, BrokerGroup1={A, B, C} and BrokerGroup2={A, B, D}, and 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(lifetime). So, each message from brokers to subscribers has the same lifetime. In the proposed protocol, each broker has to send and receive each message with a 2-dimensional vector for causality past using IP-Multicast and periodic gossips [2, 3], respecting deadline-constrained causal order [9]. On the other hand, every broker disseminates the multicast message including a 1-dimensional vector, the time-stamped information that represents the maximum number of gossip rounds, which means its deadline (lifetime), to subscribers only using gossips. So, if message m is disseminated by a broker and has bypassed the maximum number of gossip rounds, $deadline_m$, m should be discarded [9]. 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. So, in our proposed protocol, each gossip round can be characterized as a unique notation represented using colors. This proposed protocol needs (logN + α) colors because the maximum number of gossip rounds is logN and α may be application specific for buffering. In the example of figure 3, it needs 3 colors because the deadline is logN+1=3. So, the order in which they appear is like the following: red, yellow, and blue stand for each gossip round and is the repetition in the maximum number of gossip rounds.

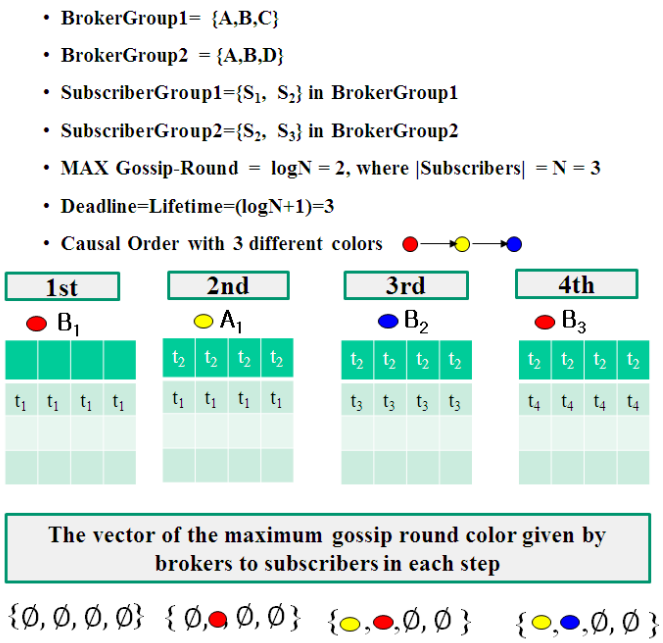

Figure 2. An example of 1-dimensional vector for color of the maximum number of gossip rounds of every message

Let us show a description of 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-dimensional vectors (MCP_m) between brokers. On the other hand, every broker disseminates the multicast message with 1-diemnsional vector(VLC_m) from brokers to subscribers. The meaning of the vector, VLC_i={0,0,0,0} is that each index indicates each broker, *e.g.* A=1, B=2, C=3, and D=4, and the deadline of the last message sent by B is red. So, this message should be delivered before the immediate incoming red. In Figure 3, in the first round, broker B generates the first multicast message and the current gossip round's color is "red", denoted "red_{B1}". Let us show a description of the protocol in Figure 5 (lines MS1-MS4). $\forall j \in Dest_m$: $VLC_i[j]$ is stored in VLC_m and the multicast message m with VLC_m is disseminated for all destination subscribers for deadline-constrained causal order. So, if subscriber j is one of the destinations, $VLC_i[j]$ is updated as the sending group gossip round, $send\_color_m$ = current_color. So, we use $m \rightarrow m' \Leftrightarrow VCL_m < VCL_{m'}$ according to this approach. In the second round, broker A generates the multicast message and the current gossip round's color is "yellow", denoted "$yellow_{A1}$". In the third round, B generates the multicast message and the current gossip round's color is "blue", denoted "$blue_{B2}$". After the first end of the maximum number of gossip rounds, the order in which they appear is like the following: red, yellow, and blue is restarting. So, in the fourth round, broker B generates the multicast message and the current gossip round's color is "red", denoted "red_{B3}".

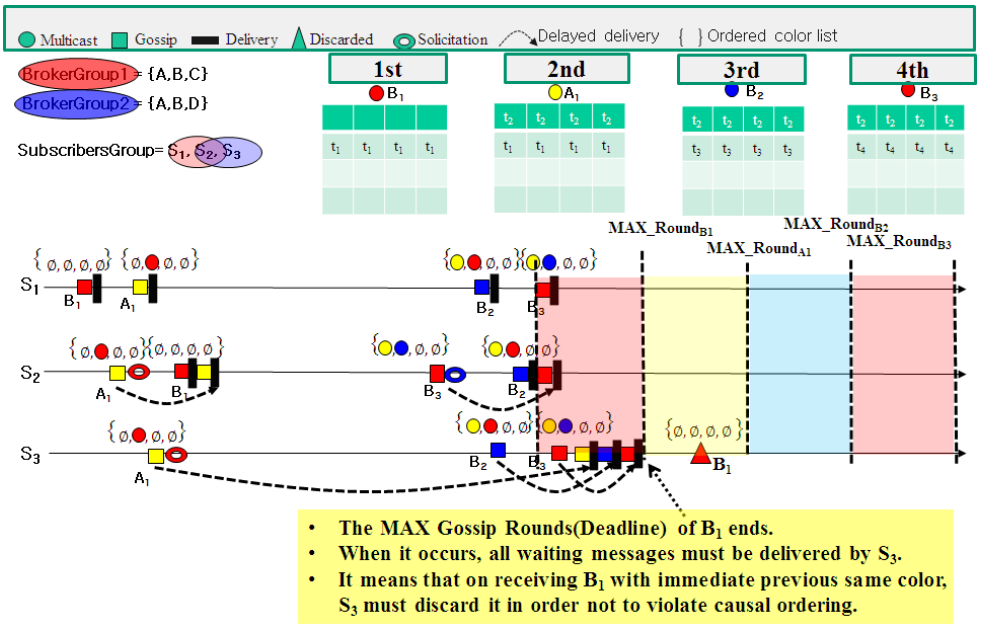

Figure 3. An example of messages with the maximum number of gossip rounds in 1-dimensional vector from brokers to subscribers

In this proposed protocol, when a broker disseminates multicast 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 show a description of the protocol in Figure 5 (lines RMS2) for delivering a message and an example in Figure 3. This is the case of message red_{B3} arriving at subscriber S3. Suppose the predicate $Del\_ok \equiv ((VCL_i[j] < VCL_m[j]) \wedge (\forall x \neq j : (VCL_m[x] \leq VCL_i[x])) \vee logical\_deadline_m \equiv (current\_color = \min(\{Deadline\_arr\_succ_m\}) \wedge deadline\_color_{min} = send\_color_{min})$ 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.

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 S3, 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 discarding a message, when same-color$_m$→ same-color$_{m'}$⇔ VCL$_{\text{same-color m}}$< VCL$_{\text{same-color m'}}$, same-color$_{m'}$ has arrived and is not yet delivered at subscriber S_i not receiving same-color$_m$. This means that same-color$_m$ was generated in the |logN+1|-th gossip round ahead of the gossip round of m' and same-color$_m$ has bypassed its deadline, and if its delivery violates causal order, it should be discarded. This is the case of message red$_{B1}$ arriving at subscriber S_3 in Figure 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-constrained causal order. So, if red$_{B1}$ has bypassed its deadline=|logN+1|=3, red$_{B1}$ should be discarded because its delivery violates causal order. Let us show a description of the protocol in Figure 5 (lines RMS1) for discarding message red$_{B1}$. If the predicate too_late ≡ ((deadline_color$_m$ < current_color) ∧ VCL$_m$[j] < VCL$_i$[j]) or Del_viol_CO ≡ (VCL$_m$[j] ≤ VCL$_i$[j]) is true, m is discarded. The case of red$_{B1}$ is that the predicate too_late is true. Message red$_{B1}$ was generated in the |logN+1|-th gossip round ahead of the current gossip round and it has bypassed its deadline=|logN+1|=3. Also, VCL$_i$[x] (lines RMS3) is updated after deliveries of yellow$_{A1}$, blue$_{B2}$ and red$_{B3}$ in order not to violate deadline-constrained causal order. So, red$_{B1}$ should be discarded.

Procedure multicast(m, deadline$_m$, Dest$_m$) from brokers to brokers
(MB1) send_time$_m$ ← current_time;
(MB2) ∀j ∈ Dest$_m$ do MCP$_i$[i,j] ← send_time$_m$ od;
(MB3) let MCP$_m$ = MCP$_i$;
(MB4) ∀j ∈ Dest$_m$ do send(m, deadline$_m$, MCP$_m$) od;

Procedure gossip(m, Partial$_m$) from brokers to brokers
(GB1) select partial brokers(PartialBk$_m$) from local_view;
(GB2) ∀j ∈ Partial$_m$ do gossip(m, summarize(MCP$_i$), PartialBk$_m$) od;

Procedure multicast(m, send_color$_m$, Dest$_m$) from brokers to subscribers
(MS1) send_color$_m$ ← current_color;
(MS2) ∀j ∈ Dest$_m$ do VLC$_i$[j] ← send_color$_m$ od;
(MS3) let VLC$_m$ = VLC$_i$;
(MS4) ∀s ∈ Dest$_m$ do send(m, deadline_color$_m$, VLC$_m$) od;

Procedure gossip(m, PartialSub$_m$) from brokers to subscribers
(GS1) select partial subscribers(PartialSub$_m$) from local_view
(GS2) ∀j ∈ (PartialSub$_m$) do gossip(m, summarize(VLC$_i$), PartialSub$_m$) od;

Figure 4. Sending procedures by each broker

When one of brokers, P_i receives multicast(m, $deadline_m$, MCP_m) from one of brokers, P_j :

let $Deadline\_arr\_succ_m \equiv \{deadline_{m'}$ such that $_{m'}$ arrived and $MCP_m \leq MCP_{m'}\}$;

let $too\_late \equiv (deadline_m < current\_time)$;

let $logical\_deadline_m \equiv (current\_time = \min(\{Deadline\_arr\_succ_m\}))$;

let $Del\_viol\_CO \equiv (MCP_m[j,i] \leq MCP_i[j,i])$;

let $Del\_ok \equiv ((MCP_i[j,i] < MCP_m[j,i]) \wedge (\forall x \neq j : (MCP_m[x,i] \leq MCP_i[x,i]))$;

(RMB1) if too_late ∨ Del_viol_CO then discard(m)

(RMB2) else wait (Del_ok ∨ $logical\_deadline_m$) Delivery Condition: DC(m)

(RMB3) $\forall(x,y) : MCP_i[x,y] \leftarrow \max(MCP_i[x,y] \leq MCP_m[x,y])$;

(RMB4) Delivery of m to P_i

(RMB5) endif

When one of brokers, P_i receives gossip(m, summarize(MCP_i), $Partial_m$) from one of brokers, P_j :

(RMB1) $\forall(x,y)$: if $(MCP_i[x,y] < MCP_m[x,y])$ then Request(m) endif

When one of subscribers, S_i receives multicast(m, $send\_color_m$, VLC_m) from one of brokers, P_j :

let $Deadline\_arr\_succ_m \equiv \{deadline\_color_{m'}$ such that $_{m'}$ arrived and $VCL_m \leq VCL_{m'}\}$;

let $too\_late((deadline\_color_m < current\_color \vee VCL_m[j] < VCL_i[j]))$;

let $logical\_deadline_m \equiv (current\_color = \min(\{Deadline\_arr\_succ_m\}) \wedge deadline\_color_{min} = send\_color_{min})$;

let $Del\_viol\_CO \equiv (VCL_m[j] \leq VCL_i[j])$;

let $Del\_ok \equiv ((VCL_i[j] < VCL_m[j]) \wedge (\forall x \neq j : (VCL_m[x] \leq VCL_i[x]))$;

(RMS1) if too_late ∨ Del_viol_CO then discard(m)

(RMS2) else wait (Del_ok ∨ $logical\_deadline_m$) Delivery Condition: DC(m)

(RMS3) $\forall(x) : VCL_i[x] \leftarrow \max(VCL_i[x] \leq VCP_m[x])$;

(RMS4) Delivery of m to S_i

(RMS5) endif

Figure 5. Receiving procedures between brokers and from brokers to subscribers

4. Performance Evaluation

In this section, we compare average throughput of our protocol based on gossip-style dissemination protocols between brokers and subscribers with that of a previous deadline-constraints causal order protocol based on traditional reliable group communications. In this comparison, we rely on a set of parameters referred to Bimodal Multicast [2] and LPBCast [3] for gossiping parameters. 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

throughtput 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 compared with that of the protocol based on traditional reliable group communications between all members.

Especially, the two protocols 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, traditional reliable deadline-constraints causal order protocol [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 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 depends on the maximum number of gossip rounds. So, the management cost of the information is cheaper than that of the previous protocol [9].

We know that which approach is more preferable depends on the user applications. But, in the network layers not using some sort of ACK mechanism to ensure reliability, such a use of more information for causal order is very expensive. In gossip-style dissemination networks, there is no ACK mechanism because members periodically gossip about the summary of received messages and our proposed protocol can use this periodic mechanism for causal order without further mechanisms. So, we verify that gossip-style dissemination approach performs computations in a distributed fashion without any information computation mechanisms. 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 on the number of brokers and brokers are more stable than subscribers.

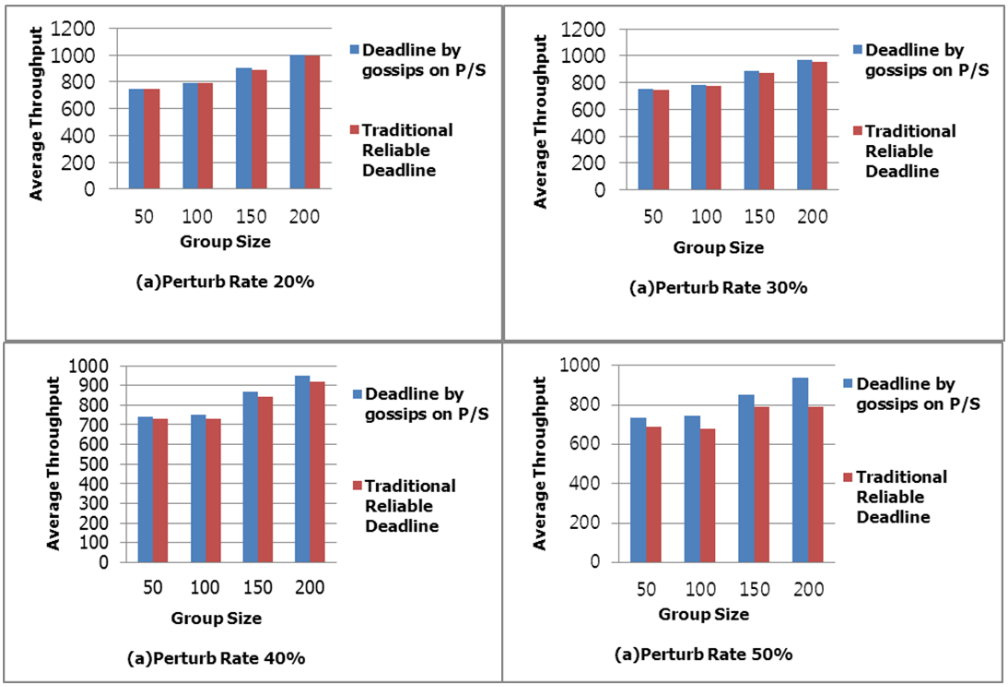

Figure 6. The average throughput as a function of perturb rate for various group sizes

5. Related Works

ASIA [6], an integrated aggregation mechanism for distributed publish/subscribe systems, is a core building block for enabling self-monitoring and adaptation. SENSTRACT [8] is mapping from queries to topics and then the corresponding underlying sensor network structure. It 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. Social Television [7] is a general term for technology that supports communication and social interaction in the context of watching television. In [1], a mobile application allows a group of people to collaborate remotely in real-time through watching the same video on their mobile devices. [5] proposes a content-based distributed publish/subscribe infrastructure for online social web platforms with better capabilities for on-the-fly analytical and data processing. Birman *et al.*, [2] 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 [3] proposes a gossip-style broadcast mechanism based on local views instead of a global view. It 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, it if arrives on time, never misses its deadline due to preceding messages.

6. Conclusions

In this paper, we present an efficient causal order protocol based on scalable P/S architecture consisting of a cluster of stable brokers for real-time collaborative applications in social web platforms to guarantee causally ordered messages delivery, respecting deadline-constraints from brokers to subscribers. In between brokers, because collaborative multicast messages are based on IP-Multicast and gossip protocols, their messages have their unique deadline. But, as all messages from brokers to subscribers are based on P/S using gossiping, they have the same deadline as the maximum number of gossip rounds. From brokers to subscribers, each message with deadline represented in colors of the lifetime of each message sent by every broker is disseminated using gossip protocols. So, the proposed protocol needs a vector, whose size is the number of brokers 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 communication overhead from brokers to subscribers. In addition, if a message m has bypassed its deadline and if its delivery violates causal order, then it should be discarded. Therefore, our proposed protocol for P/S paradigms based on gossiping results in very low communication overhead from brokers to subscribers in the context of respecting deadline-constraints.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (grant number 2012R1A1A2044660).

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] 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.
[3] 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.
[4] C. Esposito, D. Controneo and S. Russo, "On reliability in publish/subscribe services", Computer Networks, vol. 57, Issue 5, **(2013)** April, pp. 1318-1343.
[5] 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)**.
[6] A. Margara, S. T. Frischbier, P. Freudenreich, P. Eugster, D. Eyers and P. Pietzuch, "ASIA: Application-specific integrated aggregation for publish/subscribe systems", Technical report, **(2012)**, http://www.cs.otago.ac.nz/staffpriv/dme/asia/ASIA2011.pdf.
[7] K. Mitchell, *et al.*, "Social TV: Toward Content Navigation Using Social Awareness", In Proceedings of EuroITV2010, **(2010)**, pp. 283-291.
[8] 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, (**2008**), pp. 266-280.
[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)**.

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 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.