Adaptation of Storm to Large Scale Distributed Publish/Subscribe System

MASANAO MATSUURA[1] SATOSHI MATSUURA[1,2] ATSUO INOMATA[1]
KAZUTOSHI FUJIKAWA[1]

Abstract: Publish/Subscribe model is a solution for real-time stream processing on wide area sensor networks. However there is still a significant issue that Publish/Subscribe system does not guarantee message reachability. Large scale distributed systems are required to achieve scalability. To tackle these issues, we implement a real-time Publish/Subscribe sensor networking system using Storm, one of a complex event processing engines which has an ability to guarantee processes reachability in cluster system. In this demonstration, we visualize our system and show its performances.

1. Introduction

Large scale distributed networks such as a weather sensor network are deployed to wide areas and generate a large amount of data. Important requirements on these systems are push delivery mechanism, real-time and scalability. Publish/Subscribe (Pub/Sub) mechanism is an effective method to satisfy these requirements. On the other hand, traditional Pub/Sub systems have problems in reliability and scalability. The reliability means reachability of messages and integrity of messages. We tackle to solve the reachability issue and the scalability issue in this demonstration.

To achieve the large scale distributed Pub/Sub system, we have been developing a Pub/Sub architecture using a geographical overlay network [1] . The aim of this demonstration is to show an adaptation of Storm processor [2] to our Pub/Sub system. Storm is an open-source complex event processing (CEP) engine which has an ability to guarantee to process without data loss. However Storm works as a cluster system. When we deploy Storm to large scale distributed environments, it is unclear how it works well. Additionally, systems on such environments are required to realize scalability. Therefore, in this demonstration, we implement our previous work with Storm and show how the system works well with performance monitoring.

2. Publish/Subscribe System for Wide Area Sensor Network

Pub/Sub mechanism is an asynchronous messaging model. Nodes in a Pub/Sub system have two types; first type is publisher and another type is subscriber. Publisher is data provider such as sensors. Subscriber is data consumer such as users. A publisher continuously pushes generated data stream to the relay nodes on the system. A subscriber registers subscriptions based on own interests to the relay nodes. A relay node checks whether the received data match the local subscriptions. If there are matching data, the relay node forwards it to next relay node toward the subscriber.

On the other hand, large scale distributed systems such as wide area sensor networking systems are required messaging in real-time and push delivery. Pub/Sub mechanism is a solution for such requirements. However, traditional Pub/Sub systems have following two problems in reliability. First problem is reachability of messages. Such systems are required to certainly deliver important messages to subscribers within short delay. Second problem is integrity of the messages. Every message is required to be accurately filtered or calculated in delivery processes. In some critical cases such as the occurence of localized torrential rain, systems facing these problems cause some serious matters. These problems are caused by a characteristic of the Pub/Sub mechanism which is loose coupling between a publiser and a subscriber. Furthermore, scalability is an important requirement in large scale distributed systems. In this demonstration, we show an approach to tackle first problem and achieving scalability.

In previous work, we proposed a Pub/Sub architecture for a large scale distributed sensor network using a geographical structured overlay network and in-network processing. Many sensor networking systems such as a weather sensor network have strong relevance to geographical information since the sensors are deployed to real world and subscribers require data which are associated particular areas. The previous work has geographical ID-space. Nodes on the architecture communicate with each other by ID-based routing. Therefore subscribers are allowed to describe geographically-

[1] Nara Institute of Science and Technology
[2] National Institute of Information and Communication Technology

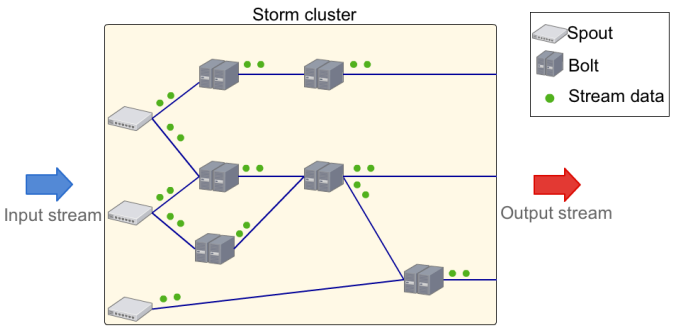

Fig. 1 Example of Storm topology.

based interests in the subscriptions. Additionally, the architecture provides a geographical dynamic load balancing method by partitioning, dividing and reassigning methods of the subscriptions. To load balance and reduce the traffic, the previous architecture is constructed by three-steps-tree topology. The topology is consisted of Edge node, Joint node and Root node. Edge and Root work to filter the messages. Joint works to filter and calculate the messages. An Edge is connected to publishers. A Joint is connected to Edges and Roots. A Root is connected to Joints and subscribers. This topology is possible to reduce traffic on the whole of a system because data are filtered by each Edge on nearby publishers.

However, the previous work does not guarantee reachability of messages. Storm is an open-source CEP engine which has an ability to guarantee processes in cluster system. Thus we adapt Storm to our previous work. We monitor the how it works and verify the performances of Storm on a large scale distributed system.

3. Adaptation of Storm Processor to Large Scale Distributed Publish/Subscribe System

Storm is one of a CEP engines for stream data in cluster system which is consisted of two kinds of nodes. First type is called Spout. Another type is called Bolt. Spout is a master node which is source of streams in a computation and checker of the streams to guarantee the processes without data loss. Spout manages data streams by assigning unique IDs. Bolt is a worker node which is processor of streams. Storm has the class of Spouts and Bolts to stream processing that is called topology as shown Figure.1. The topology includes specific processing details for the stream processing, every topology is needed to fix before run process of Storm. Storm provides a function that messages of generated by a Spout are fully processed. When Bolt has completed a process, the Bolt sends ack messages to all associated Spouts using own topology information. If Spout does not receive an ack message within limited time, the Spout detects a fault of the Bolt. Then the Spout is automatically reassigning the task on the failed Bolt to another alive Bolt. This recovery process does not need reconstruction of topology because the contents of totally processes at the topology are not changed. In other words, in Pub/Sub architecture, processed subscriptions on Storm are not changed. By these features, Storm guarantees reachability of stream processing without data loss. However, if a data consumer requires to deploy new tasks to Storm, Storm is needed reconstruction of topology including processing details. Due to the characteristic, the system has to kill the current process then start newer one in each time.

We propose a reliable Pub/Sub system with our previous work and Strom. The proposed system aims to guarantee reachability of messages and realize scalability on the system. To guarantee the reachability, we adapt Spout to Edge and adapt Bolt to Joint and Root as shown Figure.2. In Figure.2, $E1$, $J1$ and $R1$ mean Edge1, Joint1 and Root1 respectively.

While our previous work can support dynamic routing, Storm supports static routing. Storm has to determine a fixed topology in advance. Therefore Storm has to be reconstruct the topology when the system is received a new subscritpion, for example a new subscriber has be joined. To keep flexibility by dynamic routing in previous work, proposed system has following abilities to reconstruct a topology. When a Root or a Joint received a new subscription, the node registers the subscription to itself. After that, the node notices information of the subscription to the related Edges. After receiving subscription information, an Edge chooses new Joints and Roots to construct a new topology. The Edge chooses the suitable new nodes using geographical information according to the received subscription. The candidate nodes are choosed which are around an original topology. If the workload on a candidate node is light, the Edge decides on the new node. The new node receive a new subscription from the Edge. At last, the Edge sends the new topology information to relative Joints and Roots. Moreover, if a Joint or a Root is failed, the failure event is handled as specific case of reconstruction of a topology. There is a difference between registering a new subscription and handling a failure event. An Edge is required to choose the new Bolt to replace the failed node. The criteria of selecting nodes to replace are the same as registering a new subscription. Update information is propagated to all associated Joints and Roots, then the Edge stops an old topology and runs stream to a new topology. An example of a replacing process of Figure.2 is shown in Figure.3. At first, there are following four topologies. A topology $T_1 \ni \{J1, R1\}$ is constructed by $E1$. Similarly, there are $T_2 \ni \{J2, R2\}$, $T_3 \ni \{J2, R2\}$ and $T_4 \ni \{J2, J3, R2\}$. When $J2$ is crashed, $E4$ detects the event of failuer. And $E4$ picks up a new node from own topology information to replace $J2$. In this case, $E4$ chooses $J3$. If the CPU load on $J3$ is lower than a threshold, replications of the subscriptions are made on $J3$. The threshold had been defined by an administrator of the system before the system is started. The new topology is $T4' \ni \{J3, R2\}$. $E4$ updates own topology information, then $E4$ sends it to relative nodes that are $J3$ and $R2$. Finally, $E4$ runs stream to $T4'$ and stops $T4$. In these

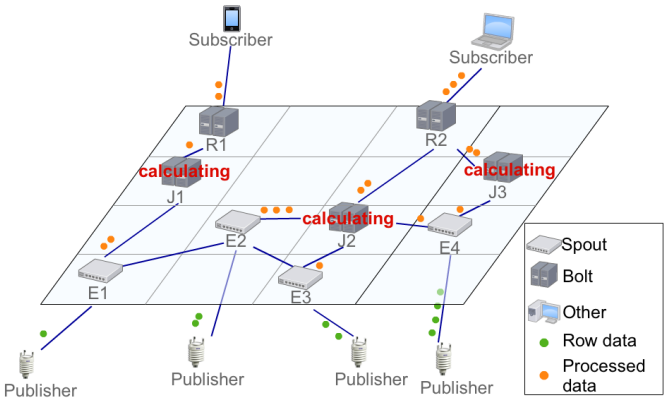

Fig. 2 Proposal architecture to guarantee reachability.

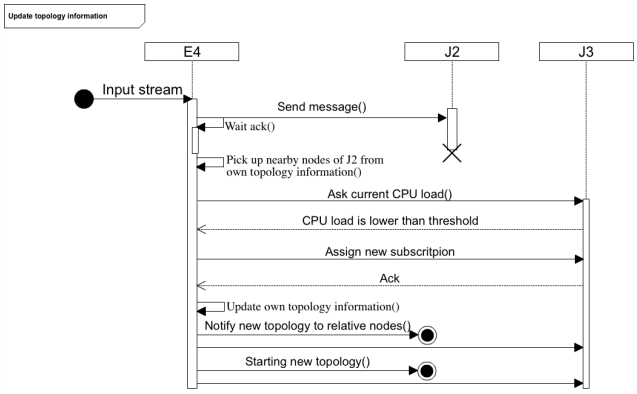

Fig. 3 Sequence of topology reconstruction.

processes, $E4$ temporarily keeps data of associated $T4$ till starting the $T4'$.

On the other hand, due to the topology management messages, this method increases CPU load on relative nodes of the reconstruction and number of messages. Such trade-offs may often cause a scalability issue in large scale distributed environments. Additionally, if a lot of new subscriptions are arrived in a moment, the system has to frequently reconstruct the topologies then it will make serious number of management messages. Therefore we verify these impacts on the system by an emulation. We clarify the performances and characteristics of the proposed system by the emulation.

4. Demonstration of Emulation

To evaluate the proposed system by an emulation, we construct it on a virtual environment. In this emulation, nodes are deployed based on following configurations. The number of publishers, Edges, Joints, Roots and types of subscriptions are fixed. The number of subscribers are arbitrary; each subscriber can join on and leave off the system at any time. We construct these environments using virtual machines. Each virtual machine is corresponded to a node. Then we analysis these logs to mesure the performance limitations.

We adapt extreme scenario for this emulation to verify the performance limitations of Storm on a large scale distributed sensor network. The scenario is assumed that is a weather senseor network. And we adopt some solid subscriptions and simple types of publications; temperature, humidity, rainfall and wind speed. Each publisher generates hundreds/thousands messages per second. Then a published raw message is defined following fields which are $< ID, from, time, temperature, humidity, rainfall, windspeed >$. The ID means a generated area based on geographical overlay network. And filtered or calculated messages are same fields, however, it is allowed that some fields are blank. At the same time, a subscription is defined following fields which are $< area, time-range, operator, type >$. The *operator* field includes following expressions which are *raw*, *average* and *summation*. For example, a user requires average of temperature for last five minutes on a area, then it is described as $< 100, 300, average, temperature >$. In this experimental environments, we control several probabilities of events (e.g., calculation failures of Bolts, communication failures/delay on links) . Changing these parameters, we measure performances, and verify reachability and scalability.

Furthermore, we develop a visualization tool for the system and real-time analysis tool for the visualization tool. Nodes on the experimental system generate logs. Then the analysis tool caluculates performances from the each logs and analysis the data flow. The visualization tool shows real-time performances and data flow from the analyzed data as shown Figure.4.

5. Related Works

Scalable Internet Event Notification Architecture (SIENA) [3] is one of a wide-area distributed Pub/Sub systems to reduce the traffic of event notification. SIENA achieve it by filtering messages on nodes of nearby a publisher and cutting off the overlapped processes using inclusive relation of the subscriptions. SIENA use broadcast routing in messaging. Due to the routing method, SIENA have scalability issue in large scale distributed environment. This system is not considered to guarantee the messages.

Christian E. et al. proposed an approach to achieve the Internet-scale fault torelant Pub/Sub system [4] . This study aimed node/link crash in large scale complex critical infrastructure such as air traffic control system. They defined three requirements for such systems which are timeliness, reliability and scalebility. Reliability means that every message is reachable even if some links or nodes become crashed. This study adopted hybrid peer to peer architecture because the system clusters the nodes using any information of underlayer such as autonomous system number. Each recognized cluster has a coordinator. These coordinators are constructed by mesh topology. The nodes in a cluster are constructed by multi-tree topology. To communicate with a node in other cluster, every node in a cluster has to via coordinator in the own cluster. And this study construct multi-overlay network. First layer is nodes layer which is intra-cluster layer. Another layer is coordinators layer which

is inter-cluster layer. The proposal uses IP multicast to intra-cluster routing and uses peer to peer multicast to inter-cluster routing. To fault torelant in nodes layer, each cluster has multiple coordinators and backup nodes. At same, to fault torelant in coordinators layer, a coordinator prepares redundant routes to other clusters. Althogh this method guarantee the reachability and fault torelant, the delay will commensurate increase due to the type of topology when a huge number of the nodes on the system. Additionally, this system needs redundant routes and nodes. Because the feature consume large amount of resources, the scalebility may be not enough in large scale distributed environments.

Reza S. K. et al. proposed a mathod to guarantee reachability of data for distributed Pub/Sub system [5] . The proposal is broker Pub/Sub model which is constructed by tree topology. This study aimed reachability of data and availability of the system. The availability means that if a node failed, the node can be recovered to the system at a later time. Every node in the system has local topology map to routing. The proposed method uses confirmation to guarantee reachability of messages. To reach the all massages, this system makes duplication of the data on some nodes. When a node is failed, neighbors of the node detect the fault event. The neighbors reconfigure own routing information. Accordingly own topology map, the neighbors notify the updated routing information to other nodes. The failed node recovered, the system nofifies a synchronize message to the node and sends duplicated messages to the recovered node. This method guarantee the reachability and availability. However, this method also has scalability issue when large amount of nodes on the system.

Thadpong P. et al. proposed a method of probabilistic formulation to predict probability of reachability of messages for a broker Pub/Sub systems over a best-effort network [6] . The formulation aimed an acyclic tree topology. The method gives soem probability distribution to interval of publication, messages lifetime, processing time on a broker and link delay. To estimate the reachability, this study introduces queuing theory. This method uses G/G/1 model to handle processes on a broker. Proposal calculates reachability using these statistical information. And the method is evaluated the validation by simulation on ns-2 using realistic data. While this analysis method is good for simple broker Pub/Sub system, it is not considered to the other realistical topology. Therefore, if similar method is used in our system, it has to be considered to scalable topology and probability which depends on geographical information.

Arnd S. I. et al. proposed a method to performance analysis and capacity planning for a Pub/Sub system [7] . This study aimed an acyclic broker Pub/Sub system. The proposed method covers two types of overlay routing. First is peer to peer architectures. Another is hierarchical architectures. Every broker on the proposed system has local routing table. The brokers are routed using inclusive relation of subscription. Additionally, this method adopts queuing theory to modeling in underlay network using M/G/1 model.

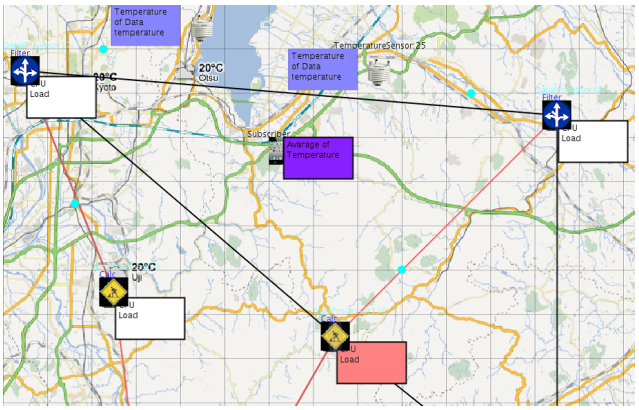

Fig. 4 Sample of demonstration.

To give probabilistic formulation to performance analysis, this mehtod gives probalbilities to routing table size, message rate, broker utilization and some delays. This method is using M/G/1 model, however, the ratio of sensor stream data distribution is not only poisson distribution. Then we have to consider to this method in G/G/1 model and another flexible topology.

Dempsy is one of framework to process streams in real-time for distributed system [8] . The framework provides dynamic topology configuration, elastic and al-least once message handling. To routing, Dempsy is based on message key but the detail is defined by application. Furthermore, node fault is handled in the way of especially case of dynamic routing. However, in some critical cases, Dempsy is not safty since it did not implement fualt tolerant architecture.

6. Conclusions

While Pub/Sub mechanism is a solution for real-time stream processing, traditional Pub/Sub systems have two problems in large scale distributed environments. First problem is that such systems do not guarantee reahchability of messages. Another problem is that such systems are required to realize scalability. We proposed a Pub/Sub architecture to solve these problems using a geographical overlay network and Storm. The architecture provides a method to reconstruct topologies on a Pub/Sub system using Storm. When a new subscription is notified or node/link fails happen, the proposed architecture reconstructs the topologies. To verify the performances of proposed method, we construct an emulation environment with virtual machines. We adopt an extreme scenario to conduct an experiment. We demonstrate the proposed system and show its performances on several scenarios.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number 24700068.

References

[1] T.Fukui, S.Matsuura, A.Inomata, and K.Fujikawa. A two-tier overlay publish/subscribe system for sensor data stream us-

ing geographic based load balancing. IEEE AINA2013, March 2013. SMPE-8.

[2] Storm. `http://storm-project.net/`.

[3] A.Carzaniga, D.S.Rosenblum, and A.L.Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19:332–383, August 2001.

[4] C.Esposito, D.Cotroneo, and A.Gokhale. Reliable publish/subscribe middleware for time-sensitive internet-scale applications. ACM International Conference on Distributed Event-Based Systems 2009, 2009. No. 16.

[5] R.S.Kazamzadeh and H.A.Jacobsen. Reliable and highly available distributed publish/subscribe service. pages 41–50. IEEE International Symposium on Reliable Distributed Systems 2009, 2009.

[6] T.Pongthawornkamol, K.Nahrstedt, and G.Wang. Probabilistic qos modeling for reliability/timeliness prediction in distributed content-based publish/subscribe systems over best-effort networks. pages 185–194. IEEE/ACM International Conference on Autonomic Computing and Communications 2010, 2010.

[7] A.Schröter, G.Mühl, S.Kounev, H.Parzyjegla, and J.Richling. Stochastic perfoemance analysis and capacity planning of publish/subscribe systems. pages 258–269. ACM International Conference on Distributed Event-Based Systems 2010, 2010.

[8] Dempsy. `http://dempsy.github.io/Dempsy/`.