Minimizing the Subscription Aggregation Cost in the Content-based Pub/Sub System

Ning Wang and Jie Wu
Department of Computer and Information Sciences, Temple University, USA
Email: {ning.wang, jiewu}@temple.edu

***Abstract*—Considering the heterogeneous subscriptions of the subscribers in the content-based publish/subscribe (pub/sub) system, the subscription aggregation technique is used to optimize the system performance, e.g., reducing the routing table, simplifying the matching procedure. However, introducing this technique also has disadvantages. If some subscribers leave the network, the brokers which aggregate subscriptions should re-configure the subscription aggregation strategy with its descendants. During this period false-positive publications, which are no longer needed by subscribers, are still propagated into the network. Therefore, it becomes paramount to examine the issue of how to conserve network resources through subscription aggregation, while simultaneously minimizing the false positive publication propagation. In this paper, we first prove the above problem is NP-hard. Then, we provide the dynamic programming approach when the re-configuration delay can be regarded as constant time. In the general case, we propose a greedy algorithm, and the corresponding performance bound is analyzed. Finally, we propose an overlay construction scheme to further fit the subscription aggregation. Extensive experimental results show that proposed algorithms achieve a good performance.**

***Index Terms*—publish/subscribe (pub/sub) system, subscription aggregation, content dissemination.**

I. Introduction

The publish/subscribe (pub/sub) is an important content dissemination paradigm for building large-scale distributed applications, such as news distribution, service discovery, stock exchange, electronic auctions, network monitoring, environmental monitoring and others. The large-scale pub/sub systems are increasingly common in industry, economic and our daily life. For example, Google's Google Cloud Pub/Sub systems [1], Microsoft's Biztalk server [2] and Yahoo's Message Broker [3]; Retailers such as WalMart and Target exchange supply chain information using the Global Data Synchronization Network (GDSN) pub/sub network [4]; the exponential growth of social networks, such as Twitter, Rich Site Summary (RSS) feeds, and music subscriptions, such as Spotify [5], further demonstrate the need for increasing the scalability expectations of these systems by using pub/sub systems.

To save the system resource of the content-based pub/sub system [6, 7], i.e., reducing the bandwidth consumption, routing tables, and accelerating the routing decisions, this paper considers the subscription aggregation technique. The subscription aggregation technique exploits the similarity among subscriptions and takes advantage of the property that a router need not index a specific subscription when there is a more general subscription [8]. That is, considering a content-based router with a general subscription $\mathbb{S}$ and a set of more specific subscriptions, $\{S_1, S_2, \cdots, S_k\}$, and $\mathbb{S} \supseteq \cup_{i=1}^{k} S_i$, the subscription aggregation will only forward $\mathbb{S}$ instead of forwarding the $\{S_1, S_2, \cdots, S_k\}$.

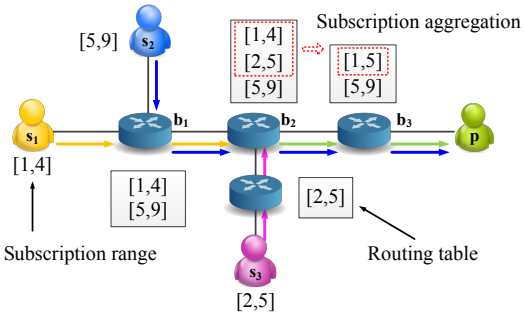


Fig. 1. The illustration of the subscription aggregation problem, where the range in the bracket is the subscription range of each subscriber.

However, the subscription aggregation technique has its costs in a dynamic network, which is the situation in reality. When the subscription, S_i, is removed at runtime, it becomes necessary to replace $\mathbb{S}$ with the more precise, but also more numerous, covered subscriptions. $\{S_1, S_2, \cdots, S_k\}$. During the re-configuration, false-positive publications, i.e., no subscribers subscribe to them any more, are still propagated, and waste the network resource. This is the cost of the subscription aggregation. It is easy to see the huge cost when one considers the millions of data generated in the Twitter, RSS feeds.

An illustration of the network model is shown in Fig. 1. There exist three subscribers, s_1, s_2, s_3, and one publisher. Each subscriber has a subscription range, which is denoted by a pair of numbers within the bracket. The subscriptions can be aggregated to save the network resource. In Fig. 1, the subscriptions $[1, 4]$ and $[2, 5]$ are aggregated at broker b_2, so that broker b_2 will further forward $[1, 5]$ rather than $\{[1, 4], [2, 5]\}$. We might not aggregate $[1, 4]$ and $[5, 9]$ at broker b_1. The reason is that though it can save the network resource, once a subscriber leaves, the possible false-positive publications might be large.

The objective of this paper is to reduce the network resource consumption by using the subscription aggregation technique, meanwhile the false-positive publications can be minimized. Specifically, there are two problems: (1) Given the subscription propagation tree, the overlay structure, what is the optimal

strategy, so that we can minimize the false-positive publications, while still saving a predetermined amount of system resources? (2) Given the network topology, how to select a good overlay structure so that we can minimize the false-positive publications, while the system resource consumption can be saved into a pre-determined amount.

The question of how to trade-off the benefit and cost of the subscription aggregation technique is non-trivial. In this paper, we prove the proposed two problems are NP-hard. Then, we start with the situation, where the re-configuration delay can be regarded as constant time. In this case, we solve the problem by using the dynamic programming approach. After that, we release the assumption of the constant re-configuration delay and propose a greedy algorithm. The corresponding performance bound is analyzed. Finally, we propose an overlay construction scheme to further reduce the false-positive publication propagation.

The contributions of this paper are threefold:

- We consider the subscription aggregation to reduce the network resource in the pub/sub system, as well as the corresponding cost.
- We propose a dynamic programming approach, when the subscription re-configuration takes constant time. Then, we release this assumption and propose a greedy algorithm, which satisfies the sub-modularity. The corresponding performance bound is further analyzed.
- We propose an overlay construction scheme which considers both the subscription range and the cost during the subscription aggregation.

The remainder of the paper is organized as follows. The related works are in Section II. The network model and problem formulation are introduced in Section III. Then, when the re-configuration can be regarded as constant time, an approach is provided in Sections IV. A greedy approach in a more general case is presented in Section V. The evaluation and results are shown in Section VI. We conclude the paper in Section VII. A proof is provided in the Appendix.

II. Related Works

There are a number of works on the pub/sub system design [8–10]. Some examples are Gryphon [6], Cobra [7]. In the following, we will briefly summarize the techniques and challenges in the content-based pub/sub systems.

Subscription optimization: Aggregating subscriptions in a content-based network has been implemented in many publish/subscribe systems [11]. The most simple technique is the covering optimization. The covering relationship between filters can be depicted as: A filter F covers a filter G denoted by $F \supseteq G$, Then, we can remove the filter G to remove redundant subscriptions from the network, maintain compact routing tables and reduce network traffic. The merging optimization goes a little bit further. Formally, a filter F is a merger of set of filters $F_1, \cdots, F_n$, if $F \supseteq \cup_{i=1}^{n} F_i$. The publication set of the merger can be exactly equal to the union of the publication sets of the original filters or a little bit larger. Merging subscriptions can reduce the number of subscriptions,

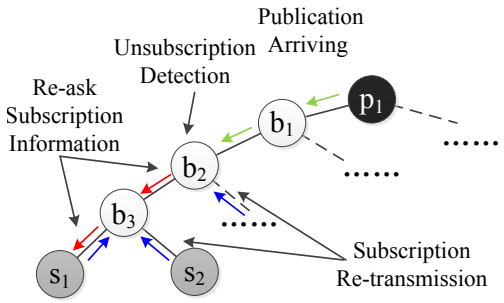


Fig. 2. An example of the subscription re-configuration, where broker b_2 receives the subscription release message. Then it requests its descendants to re-send subscriptions. In this period, the false-positive publications generate.

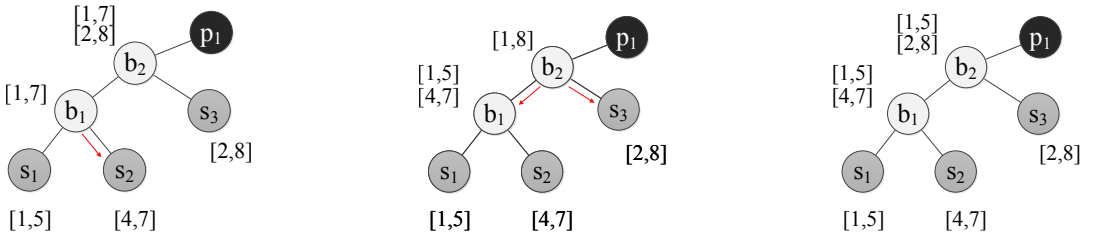
but may allow publications to be forwarded that do not match any of the original subscriptions. In order to apply merging technique, it must be possible to efficiently compute mergers and if imperfect merging is performed, the number of the unwanted publications must be small.

Subscription releasing: In the PADRES system [12], two variants of the covering optimization are implemented. The first one is called active covering. With active covering, when a subscription S' arrives at a broker after a subscription S that covers S', the broker does not forward S. Moreover, if S' arrives before S, S is forwarded, and also brokers that see both S' and S delete S' from their routing tables. This helps to ensure more compact routing tables, but requires more processing and network traffic to clean up unnecessary S' routing state. Under lazy covering on the other hand, in the case where S' arrives before S, S is simply forwarded and none of the S' routing state is cleaned up. This is a cheaper operation but results in larger routing tables over time. *Topology overlay network:* Previous research in the area of Application Layer Multicast (ALM) [13] has shown that the knowledge of the underlying (router-level) network topology is beneficial to achieve low data dissemination delay and high bandwidth data dissemination [14]. In [15], the authors exploited the knowledge of event traffic, user subscriptions and topology of the underlying physical network to perform efficient routing in a publish/subscribe system. In [16], the authors further considered the rich content formats in video dissemination, except the basic task, the overlay should be able to minimize the computation in format transformation.

In this paper, we propose to use the merging technique to reduce the system resource. The disadvantage of the merging technique during the subscription releasing is also considered. Additionally, the overlay construction is addressed.

III. System Model and Problem Formulation

In this section, we first introduce the content-based pub/sub system model. Then, we gradually quantify the benefit and the cost of using the subscription aggregation technique. After that, we formulate the problem in this paper.



(a) S_1 and S_2 are aggregated at b_1 (b) S_1, S_2 and S_3 are aggregated at b_2 (c) S_2 and S_3 are aggregated at b_2

Fig. 3. An example to explain the trade-off of the subscription aggregation, where the numbers in the bracket is the subscription range of each subscriber, (a) saves 2 bandwidth units, (b) saves 2 bandwidth units and (c) saves 1 bandwidth unit. However, once s_1 leaves the network, the cost of (a) is 3. The cost of (b) is 1, and there is not cost for (c).

A. Network Model

In the content-based pub/sub system, there are three types of nodes: publishers, brokers, and subscribers. A core-network, built by the brokers connects the subscribers and publishers. This broker-core structure is widely used in the Internet-based pub/sub system [17], such as Twitter and Facebook. Note that the subscribers might be publishers, and the publishers can be subscribers for different publications. Each subscriber will send its subscription to the publisher, thus a subscription propagation tree is built. As for the subscription aggregation, if two subscriptions via the same broker are sent to the publisher, the broker can do subscription aggregation ($\lambda = 1$) or not ($\lambda = 0$). The publication will follow the reverse order of the subscription tree to the corresponding subscribers.

B. Benefit and Cost of the Subscription Aggregation

The different subscribers' subscriptions can be aggregated by exploiting the similarity among subscriptions. It takes advantage of the property that a router need not index a specific subscription when there is a more general one [8]. Therefore, the subscription aggregation technique can lead to fewer subscriptions, and thus smaller routing tables. For example, for a router with a general subscription $\mathbb{S}$ and a set of more specific subscriptions, $\{S_1, S_2, \cdots, S_k\}$, and $\mathbb{S} \supseteq \cup_{i=1}^{k} S_i$. The subscription aggregation will only forward $\mathbb{S}$, instead of forwarding the $\{S_1, S_2, \cdots, S_k\}$. To quantify the system resource saving, G_i, we assume it is proportional to the number of bandwidth saving. In the above example, the bandwidth is reduced to $\frac{1}{k}$ of the original consumption.

The subscription re-configuration, caused by the dynamic network topology in reality [8], is described as follows: once a subscriber leaves the network, it sends the release-subscription message to the corresponding broker, where the subscriptions are aggregated. After the release-subscription message arrives at the broker, the broker will ask its descendants to re-send their subscriptions. An illustration is shown in Fig. 2. In Fig. 2, when broker b_2 receives the subscription-releasing message, it requests all its descendants to re-send their subscriptions, which are the red arrows in Fig. 2. Before the broker b_2 receives all the response(s), where are the blue arrows in Fig. 2, the false-positive publications, i.e., the publications without subscription will be propagated. The amount of the false-positive publications are related to the false-positive range and the re-configuration delay. We regard the amount of false-positive publications as the cost for the subscription aggregation technique.

The false-positive publication amount caused by the subscription aggregation can be estimated by the probability approach. It is reasonable to assume that at most one subscriber leaves the network in a period. Then, the key observation is that if a subscription range is only subscribed by one subscriber, called unique subscription, once it leaves the network, there will be false-positive publications during the re-configuration. As the result, the unique subscription range size has an impact on the false-positive publication. Besides, the re-configuration delay is the other factor which influences the false-positive publication amount. Let us denote the increasing of the unique subscription range and the re-configuration delay by adding aggregation λ_{ij} as Δu_{ij} and r_{ij}, respectively. The publication generating rate, which represents the rate that a publisher generates the publications in the unit subscription range is denoted as g. The subscriber leaving rate, which represents the rate that a subscriber leaves the network is denoted l. Therefore, the false-positive publications caused by the subscription aggregation, C_{ij}, can be estimated as a function with Δu_{ij} and r_{ij}. If we assume the uniform publication generating rate for all the subscription range, then l and g are the constant value.

$$C_{ij} = f(\Delta u_{ij}, r_{ij}) = gl \cdot (\Delta u_{ij} \times r_{ij})$$

To minimize the C_{ij}, we need to jointly minimize the Δu_{ij} and r_{ij}, the subscription similarity and re-configuration delay. We leave the heterogeneous publication generating rate in different subscription range as our future work.

Fig. 3 is a detailed example to explain the benefit and cost of the subscription aggregation. If no subscription aggregation is applied, each subscription consumes a certain number of bandwidth, which is equal to number of hops from the subscriber to the publisher. Once two subscriptions are aggregated at some broker, the bandwidth consumption is reduced to a half in the following route to the publisher. Fig. 3 provides three subscription aggregation strategies. In Fig. 3(a), the subscriptions 1 and 2 are aggregated at broker b_1, it saves 2 bandwidth units. It is because that 1 bandwidth

unit between b_1 and b_2, and 1 bandwidth unit between b_2 and p_1 are saved. However, once s_1 leaves the network, the publications in subscription range $[1, 4]$ (the size is 3) become into the false-positive publications. In this example, we assume the link delay between two connected brokers is 1 unit. Broker b_1 can re-configure its descendants' subscriptions in one hop, the total cost is 3×1 units. For the strategy b, if we aggregate subscriptions S_1, S_2 and S_3 at broker b_2 as shown in Fig. 3(b), we can also save 2 bandwidth units (reducing 3 subscriptions to 1). In this case the publications in the subscription range $[1, 2]$ become false-positive publications, when s_1 leaves the network. The broker b_2 needs to reconfigure the subscription with its two children b_1 and s_3 within one-hop. So, the cost is 1×1 in this situation. As for the strategy c, shown in Fig. 3(c), where the subscription aggregation of S_2 and S_3 are aggregated at broker b_2, the leaving of s_1 does not cause the false-positive publication of the network.

C. Cost Minimization Problem

To save a target amount of network resources, while the amount of false-positive publications is minimized at the same time, is a vital issue in the practical system. Therefore, we focus on the following cost minimization problem:

$$\begin{aligned} \min \quad & \sum_{i \in X} C_{ij} \times \lambda_{ij} \\ \text{s.t.} \quad & \sum_{i \in X} G_{ij} \times \lambda_{ij} \geq \theta \\ & \lambda_{ij} \in \{0, 1\} \end{aligned}$$

where θ is the pre-defined target gain that we would like to achieve through subscription aggregation technique. In reality, the size of θ reflects the stress for reducing system resource. That is, a larger θ means that we would like to reduce more system resources, at the cost of generating some false-positive publications, and vice visa.

Theorem 1. *The proposed cost minimization problem in this paper is NP-hard.*

Proof. The proposed problem can be reduced into the 0-1 integer programming problem [18] under the situation that the users' subscriptions are independent with each other. In this case, the C_{ij} and G_{ij} become constant value, since they are not related to the other aggregation. The C vector and G vector can be regarded as the vector in the object function and in the inequality constraint. In this case, the object function and the inequality constraints become linear function. Additionally, we have a set of 0-1 aggregation selection. Its formulation is the same as the 0-1 integer programming's formulation. Therefore, the proposed cost minimization problem is NP-hard. □

IV. Constant Re-configuration Delay

In this section, we first solve the proposed problem in the special case, where we consider the re-configuration delay in different aggregation strategies to be a constant value. In this case, the proposed problem is simplified, since the false-positive publications are only decided by the unique subscription range size. However, in this case, the simplified problem is still a NP-hard problem. The only difference is that all the elements in vector G are the same.

1) Dynamic Programming: If the subscription range are all integral numbers, we can get the optimal solution by using the dynamic programming approach. Otherwise, we discretize the subscription, then the solution of dynamic programming is bounded by the discretization degree [19]. The following is the description of this approach. We sort the subscriptions from the minimum to maximum, and aggregate subscriptions based on this order, then, the original problem can be divided into several sub-problems. That is, if we order the subscribers based on their subscription range from 1 to n, where n is the number of the subscribers. The optimal solution from the first i subscribers is within the optimal solution for the first j subscribers plus the aggregation result of the subscription from $j+1$ to i.

Let us denote the system resource saving and the corresponding false-positive publications by aggregating from the ordered subscribers i with k aggregations as $C[i, k]$.

$$C[i,k] = \begin{cases} 0 & (i = 1) \\ \min\limits_{0 \leq j \leq i} \{C[j, k-1-j+i] + Aggregate[j+1, i]\} & (i > 1) \end{cases} \tag{1}$$

where $Aggregate[j+1, i]$ represents to aggregate from S_{j+1} to S_i. An observation in this case is that if two subscriptions are aggregated, the subscriptions within the union of these two subscriptions should also be aggregated. The reason is that by aggregating the range within these two subscriptions, there are no extra false-positive publications. However, the network resource is further reduced by the subscription aggregation. Based on this observation, we can further reduce the calculation complexity in the procedure.

Then, the problem is transferred into the following problem:

$$\begin{aligned} min \quad & C[n, m] \\ s.t. \quad & \alpha m \geq \theta, \end{aligned}$$

where α is a benefit parameter. In this algorithm, once we find that the current communication saving reaches the target, we will stop there.

V. Uniform Re-configuration Delay

In this section, we discuss the uniform re-configuration delay for different aggregation strategies. That is, the re-configuration delay is proportional to the hop distances. Though two subscribers have similar subscription ranges, if they are far away from each other, aggregating these two subscribers might cause a huge cost for re-configuration. In this case, we have to balance the subscription similarity and distance of two subscribers.

In this scenario, since the difference re-configuration delay for different subscription aggregation, we cannot use the dynamic programming approach anymore. It is because the new subscription aggregation, will change the current aggregation benefit and cost. The optimal solution in a sub-tree might not be the optimal strategy for the whole tree. For example, in Fig.

Algorithm 1 DPA Algorithm

Input: The subscription tree for each publisher, the subscription range of each subscriber $\{S_1, S_2, \cdots, S_n\}$, the threshold θ.

Output: The subscription aggregation result, X.

Sort all the subscription ranges from min to max.
while A subscription range is covered by more than two subscribers **do**
    Aggregate the two subscribers whose similarity is the highest.
Give each remaining subscriber an index from 1 to n, where n is the number of remaining subscribers.
for $i = 0$ to n **do**
    $C[0, i] = 0$ and $G[0, i] = 0$;
for $i = 0$ to n **do**
    $min_{0 \le j \le i}\{C[j, k-1-j+i] + Aggregate[j+1, i]\}$
    Change the corresponding λ_i
    if $G[1, j] \ge \theta$ **then**
        break;
return X

3, from the viewpoint of the broker b_1, the unique subscription range $[1, 3]$. However, from the viewpoint of the broker b_2, the unique subscription range is $[1, 2]$.

To solve the problem in the general case, we can still sort all the subscriptions of the subscribers from the minimum to the maximum, and apply the dynamic programming approach as in Section IV, without considering the re-configuration delay. This algorithm is called the *subscription-range-only algorithm*.

Theorem 2. *The subscription-range-only algorithm achieves h approximation ratio, where h is the ratio of the largest delay and the smallest delay from the subscriber to the publisher.*

Proof. The idea is that we might regard that all the subscribers are connected to the same broker. Then, if we move the subscribers back to their real positions, the cost increasing amount can be bounded. First, if we consider that all cost is the minimum cost, we can use the dynamic programming approach to aggregate subscription. The solution must be a lower bound of the optimal solution. Then, we move the subscribers to their corresponding positions. However, the real cost can increase at most h times for each subscription aggregation, where h is the ratio of the largest and the smallest subscriber to the publisher path delay. □

The above theorem shows that this algorithm works well when the network is small. The advantage is that it considers the subscription ranges of all the nodes in the network globally.

Theorem 3. *The cost of subscription aggregation in the proposed problem is sub-modular.*

Proof. Assume that we have already decided to do the following aggregation in sequence: $\lambda_1, \lambda_2, \cdots \lambda_k$, and $X = \{\lambda_1, \lambda_2, \cdots \lambda_k\}$. λ_{k+1}, λ_{k+2} are two feasible subscription aggregations. If the subscription range of the $(k+1)^{th}$ aggregation might have an overlap with the $(k+2)^{th}$ subscription aggregation, this overlap will reduce the aggregation cost bought by the $(k+2)^{th}$ subscription aggregation

$$\Delta_{\lambda_{k+2}} C(X) \ge \Delta_{\lambda_{k+2}} C(X \cup \{\lambda_{k+1}\}) \tag{2}$$

where the $\Delta_\lambda C(X)$ is the additional cost by adding one more aggregation, λ, based on the current aggregations, X. Based on the above in-equation, the cost is sub-modular. □

Algorithm 2 MEFA Algorithm

Input: The subscription tree for each publisher, the subscription range of each subscriber $\{S_1, S_2, \cdots, S_n\}$, the threshold θ.

Output: The subscription aggregation result.

$X = \phi$
while $\sum G < \theta$ **do**
    if $\exists \lambda$ such that $G_i > 0$ **then**
        Select the λ_i with maximum G_i/C_i.
        $\lambda_i = 1$
return X

Based on the observation that the cost is sub-modular, we propose a greedy algorithm, the-most-efficient-first algorithm (MEFA). In this algorithm, while we have not reached the pre-determined network resource saving target, we sort all the remaining aggregation combinations by the ratio of the benefit and the corresponding cost. Then, we pick the two subscriptions which are the most efficient to be aggregated.

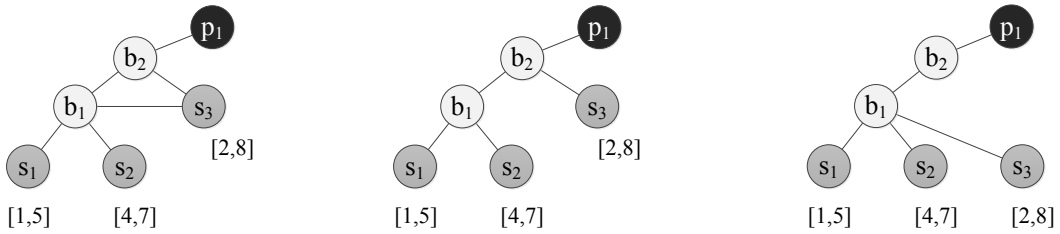
Theorem 4. *The MEFA algorithm achieves $1 + \rho ln\theta$ approximation ratio, where ρ is the curvature of the submodular cost.*

The proof of the Theorem 4 is in the Appendix.

VI. Overlay Construction

In this section, we propose an overlay construction approach to fit the subscription aggregation technique. The reason is that the subscriber's subscription similarity is not considered in existing approaches.

In previous work [8], the subscription propagation tree construction procedure is as follows: the publisher first advertises a description of the data they are about to send, and this advertisement message is flooded across the network. Each node chooses the sender of the first arrived advertisement message as its parent node. Therefore, a shortest path tree rooted at the publisher is generated. Next, subscribers issue a subscription defining their interests, and this subscription is routed hop-by-hop along the reverse path of matching advertisement trees towards publishers. Finally, publications from publishers are disseminated hop-by-hop following the reverse paths of matching subscriptions until they are delivered to interested subscribers. After a certain period of time, the publisher receives all the subscriptions, and a subscription propagation tree, the overlay structure, is built.



(a) Physical lay (b) Overlay 1 (c) Overlay 2

Fig. 4. An example to explain the trade-off of different overlays. Fig. 4(a), represents the physical lay in the pub/sub system. In overlay 1 all three subscribers are connected to the publisher through the shortest path. However, in overlay 2, the delay for subscriber 2 increases, at the benefit of reducing the unique range size under broker b_1.

An example is shown in the Fig. 4. There exist two different overlay structures for the subscribers s_1, s_2, and s_3. For overlay 1, it is the shortest path tree from the publisher to these three subscribers. However, in this case, for broker b_1, it can only consider the subscription range of subscribers s_1 and s_2 to do the aggregation decision. In this example, the unique subscription range size for subscribers s_1 and s_2 is 5. However, for subscriber s_3, it has two options. One is to connect with broker b_2, which costs less delay. Another option is to connect with the inter-mediate broker, b_1, which cost more delay. However, in this case, for the broker b_1 to do aggregation decision, the unique range size reduces to 2 units as shown in Fig. 4(c). There is a delay-cost trade-off.

In this paper, we propose a greedy overlay construction method. The key idea of the overlay construction is to make the subscribers with similar subscription ranges under the same broker. Therefore, this overlay construction approach jointly considers the subscriptions of subscribers and the delay, it should fit the subscription aggregation technique better.

This algorithm has two stages: 1) we build the shortest-path tree as the subscription propagation tree for each publisher and its subscribers. This stage is aimed to minimize the publications propagation delay. After the first stage, each subscriber chooses a broker, called the subscriber-end-broker (SEB), to connect into the network. 2) In the second stage, each subscriber searches the brokers within a certain delay. If the subscriber is assigned to a broker, and this new broker can reduce the unique subscription range size by δm. Here, we propose a metric, $\frac{\delta m}{\delta d}$, which means if a re-assignment can reduce the unique subscription range size δm, but increase δd more delay, we will re-assign the subscriber to the new broker. We keep doing this procedure until we cannot find an aggregation which satisfies $\frac{\delta m}{\delta d} > \gamma$. The parameter γ represents the importance of the unique subscription range over the re-configuration delay. Clearly, if the re-configuration delay is large, we might not re-shape the overlay after step 1. So, the γ should be assigned to a large value. On the other hand, we might need to assign the γ to a relatively small value.

Algorithm 3 Overlay Construction Algorithm

Input: The network topology of the subscription tree for each publisher, the subscription range of each subscriber $\{S_1, S_2, \cdots, S_n\}$, the threshold γ.

Output: The overlay construction result.

1: The publisher floods the advertisement to all the network.
2: Once a subscriber receives the advertisement, it chooses the sender as its SEB.
3: **while** $\exists \frac{\delta m}{\delta d} > \gamma$ **do**
4: Select the modification with maximum $\frac{\delta m}{\delta d}$.
5: Change its SEB to the new broker.
6: **return** The modified overlay structure.

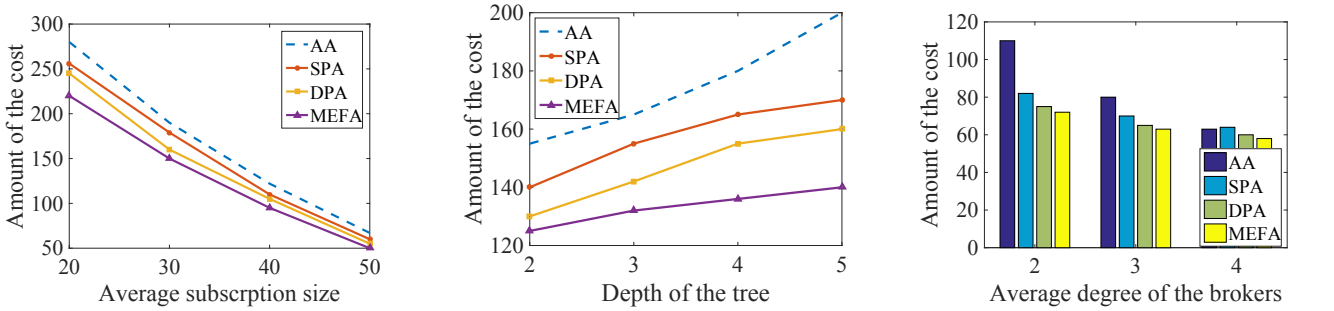
VII. Experiments

In this section, we compare several algorithms mentioned in this paper by extensive trace-driven experiments. We first introduce the experimental settings and their parameters. Then, we will discuss the performance evaluation results.

A. Trace Introduction

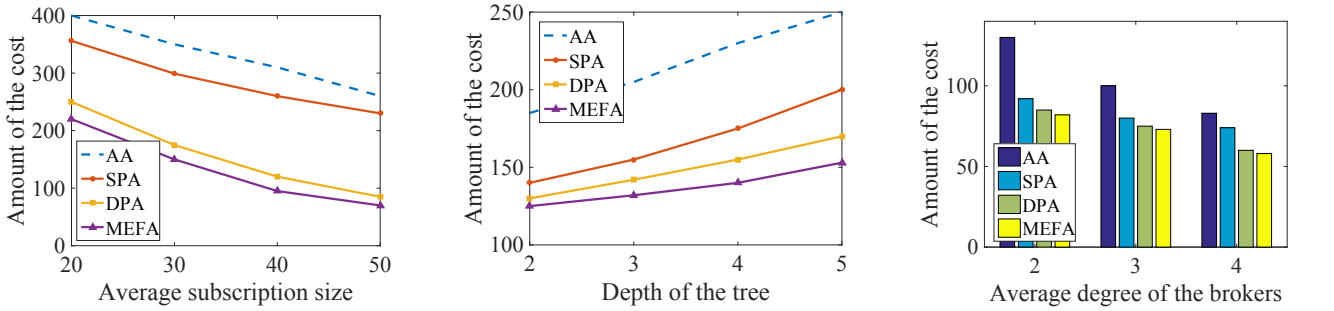
1) Synthetic trace: The network topology that we refer the topology in [8]. In their experiments, the network consists of 49 brokers, the 5 brokers are denoted as core brokers. Publishers connect to one of the core-broker, and subscribers are randomly distributed among the remaining edge brokers. This might represent a messaging platform for the business workflows in a large enterprise with multiple departments and partners. The components in each workflow would subscribe to triggers of interest, and these workflows are invoked by publications from external clients. In the uniform distribution, the subscriber has the similar probability to be interested in every piece of the subscription range. However, in the exponential distribution, the majority of subscribers are interested in a special piece of the subscription range. Based on this basic topology, in our experiments, we also change the depth of the tree and change the average number of children in the broker.

2) Real trace: Here, we use the topology from the Facebook trace which is from Stanford Large Network Dataset Collection [20]. The social networking pub/sub workload is used from a trace from the Facebook [21]. In this trace, the number of brokers is 120. In the experiment, we randomly pick a small network with 120 broker nodes from [20]. Among the nodes connected to these 120 brokers, one node will be randomly selected as the publisher, and the remaining nodes



(a) average depth 2, average degree 2 (b) average subscription size 50, average degree 2 (c) amount subscription size 50, average depth 2

Fig. 5. Performance comparison in the uniform subscription distribution.



(a) average depth 2, average degree 2 (b) average subscription size 50, average degree 2 (c) average subscription size 50, average depth 2

Fig. 6. Performance comparison in the exponential subscription distribution.

are selected as the subscribers. In [21], each publication has an ID. We select the top 10 publications for each subscriber. Then, the publication from the smallest ID to the largest ID are considered as the subscription range of a subscriber.

B. Experiment Setting

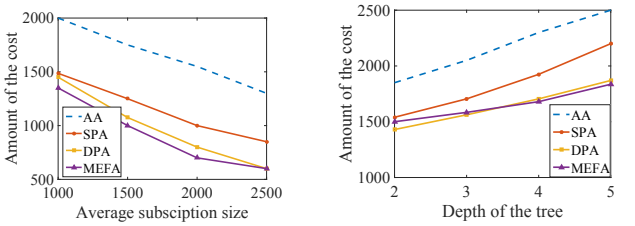
In the synthetic trace, we apply the similar topology as in the [8]. Furthermore, in our experiments, we change the depth of the tree (2 to 4) and the average degree of the brokers (2 to 4). For the subscribers' subscription range, we randomly generate the uniform and exponential distribution for the subscribers. The subscribers' subscription range are extracted from the Facebook dataset. In the experiments, a subscriber only has one subscription range. The publisher randomly generate publications which has a certain value in the subscription range every second. Every subscriber has equal probability to leave the network. The amount of cost is the amount of the false positive publication.

C. Algorithm Comparison

The algorithm comparison is mainly in two parts. In the first part, we compare the different algorithms' performance regarding the amount of false positive publications (amount of the cost). (1) All aggregation (AA) algorithm, which means the brokers will decide to aggregate all the subscription or not, based on their similarity. This algorithm is proposed in the [8]. (2) Similarity-pair (SPA) aggregation algorithm, which means that the subscriptions are aggregated based on the subscribers' subscription similarity. (3) Dynamic programming based aggregation (DPA) algorithm, which means that subscribers calculate the best subscription strategy without considering the distance. (4) The-most-efficient-first (MEFA) aggregation algorithm, which is the greedy algorithm that we proposed in this paper. The second part of the performance comparison is mainly focused on the overlay construction method. (1) The overlay construction algorithm, which only considers the shortest path from the publisher to the subscriber, the subscription range similarity of different subscribers is not considered called SO algorithm. (2) The proposed overlay construction algorithm, which jointly considers the distance to the publisher and the subscription ranges among several near by subscribers called SA algorithm.

D. Experiment Results

In the experiment, we mainly consider the performance of different algorithms under the three different settings: (1) the comparison for different average subscription size. 2) the comparison for different cost, and (3) the comparison for different average number of subscribers under a broker.



(a) depth 2, degree 2 (b) subscription size 528, degree 2

Fig. 7. Performance comparison in Facebook dataset

E. The Performance Results

1) The uniform subscription distribution: Fig. 5 shows the performance results. The MEFA algorithm achieves the best performance, followed by the DPA algorithm, SPA algorithm, and the AA algorithm in three different settings. In Fig. 5(a), along with the increasing of average subscription size, the performance of the proposed four algorithms becomes even more similar. The reason is that, since the subscription range are uniformly distributed, the cost for each aggregation is nearly the same, all the algorithms will keep aggregating. In Fig. 5(b), along with the increase of depth, the performance between these four algorithms increase. in Fig 5(c), we get the similar result.

2) The exponential subscription distribution: Fig. 6 shows the performance results of the exponential subscription distribution. In Fig. 6(a), the MEFA algorithm achieves the best performance, followed by the DPA algorithm, SPA algorithm, and the AA algorithm in three different settings. However, the performance difference between these four algorithm is significant. The performance of the DPA and MEFA algorithms are much better than the remaining two algorithms. The reason might be that in exponential subscription distribution, many subscribers will subscribe some similar subscriptions. If we only consider the similarity of the subscribers, it might lead to a huge cost, since two similar subscribers' distance in the network might be huge. Fig. 6(b), along with the increase of the depth, the performance difference between these four algorithms increase. It is the similar when we increase the average number of the broker, as shown in Fig 6(c).

3) The Facebook trace result: The experimental results are shown in Fig. 7. In Fig. 7(a), the performance of the MEFA and DPA algorithms are much better than the remaining two algorithms. Compared to the AA algorithm, the cost saving is up to 50% . In Fig. 7(b), along with the increasing of the depth of the tree, the advantage of the MEFA and DPA algorithms increases. When the depth of tree becomes 5, the performance saving of the MEFA and DPA can save 50% more than AA algorithm. Compared to the SPA algorithm, the MEFA and DPA algorithms still save more than 30%. The results demonstrate the necessity of jointly considering the unique subscription range and the re-configuration delay.

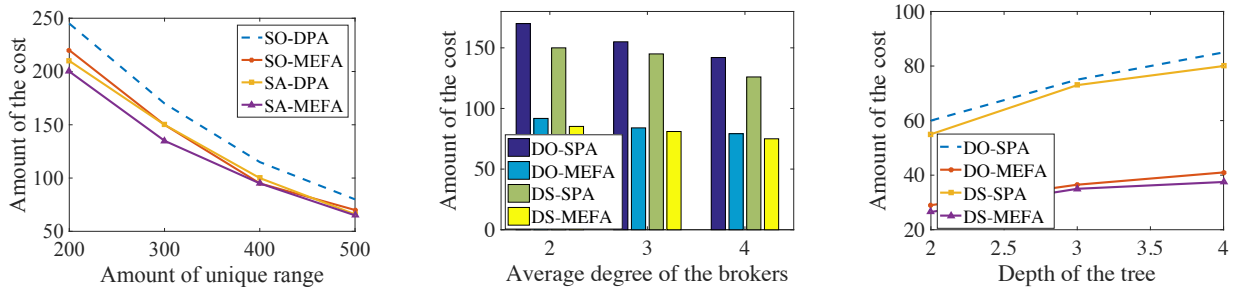
4) The overlay construction result: The MEFA and DPA algorithms are applied into the overlays which are constructed by two overlay construction algorithms. The results are shown in Fig. 8. Fig. 8(a) shows that when the subscribers' subscriptions' difference is large, the proposed SA algorithm can reduce about 20% about the cost. Figs. 8(b) and 8(c) also show that when the depth of tree is large or the average degree of broker is small, a good overlay structure can improve the performance of subscription aggregation.

VIII. Conclusion

In this paper, we focus on the subscription aggregation technique in the content-based pub/sub system to optimize the system performance. The advantage and disadvantage of the subscription aggregation is jointly considered. We first prove that the above problem is NP-hard. Then, we provide a dynamic programming approach when the re-configuration delay can be regarded as a constant time. In a general case, there exists a trade-off between the subscription similarity and the re-configuration delay. We propose a greedy algorithm with sub-modularity property. The theory proof shows that the greedy algorithm has the $1 + \rho ln\theta$ approximation ratio. Finally, we propose an overlay construction scheme to further fit the subscription aggregation.

References

[1] https://cloud.google.com/pubsub/overview/, 2015.
[2] R. Seroter, M. Brimble, J. Cooper, C. Dijkgraaf, and M. Morar, *SOA Patterns with BizTalk Server 2013 and Microsoft Azure*. Packt Publishing Ltd, 2015.
[3] B. F. Cooper, R. Ramakrishnan, U. Srivastava, A. Silberstein, P. Bohannon, H.-A. Jacobsen, N. Puz, D. Weaver, and R. Yerneni, "Pnuts: Yahoo!'s hosted data serving platform," *Proceedings of the VLDB*, 2008.
[4] K. Nakatani and T.-T. Chuang, "Global data synchronisation: deployment and implementation issues of a global standard-based information system," *International Journal of Services and Standards*, vol. 8, no. 2, pp. 157–173, 2012.
[5] V. Setty, G. Kreitz, R. Vitenberg, M. van Steen, G. Urdaneta, and S. Gimåker, "The hidden pub/sub of spotify:(industry article)," in *Proceedings of the ACM DEBS*, 2013.
[6] M. K. Aguilera, R. E. Strom, D. C. Sturman, M. Astley, and T. D. Chandra, "Matching events in a content-based subscription system," in *Proceedings of the ACM PODC*, 1999.
[7] I. Rose, R. Murty, P. R. Pietzuch, J. Ledlie, M. Roussopoulos, and M. Welsh, "Cobra: Content-based filtering and aggregation of blogs and rss feeds." in *Proceedings of the NSDI*, 2007.
[8] M. Chen, S. Hu, V. Muthusamy, and H.-A. Jacobsen, "Congestion avoidance with incremental filter aggregation in content-based routing networks."
[9] N. K. Pandey, K. Zhang, S. Weiss, H.-A. Jacobsen, and R. Vitenberg, "Minimizing the communication cost of aggregation in publish/subscribe systems."
[10] M. H. Hajiesmaili, L. T. Mak, Z. Wang, C. Wu, M. Chen, and A. Khonsari, "Cost-effective low-delay cloud video conferencing," in *Proceedings of the IEEE ICDCS*, 2015.
[11] G. Li, S. Hou, and H.-A. Jacobsen, "A unified approach to routing, covering and merging in publish/subscribe systems based on modified binary decision diagrams," in *Proceedings of the IEEE ICDCS*, 2005.
[12] E. Fidler, H.-A. Jacobsen, G. Li, and S. Mankovski, "The padres distributed publish/subscribe system." in *FIW*, 2005, pp. 12–30.
[13] X. Jin, W. Tu, and S.-H. G. Chan, "Scalable and efficient end-to-end network topology inference," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 6, pp. 837–850, 2008.



(a) average depth 2, average degree 2

(b) average subscription size 50, average degree 2

(c) average subscription size 50, average depth 2

Fig. 8. Performance comparison in different overlay construction.

[14] S. Frischbier, A. Margara, T. Freudenreich, P. Eugster, D. Eyers, and P. Pietzuch, "Asia: application-specific integrated aggregation for publish/subscribe middleware," in *Proceedings of the ACM Middleware*, 2012, p. 6.

[15] M. A. Tariq, B. Koldehofe, and K. Rothermel, "Efficient content-based routing with network topology inference," in *Proceedings of the ACM DEBS*, 2013.

[16] H. Jafarpour, B. Hore, S. Mehrotra, and N. Venkatasubramanian, "Ccd: a distributed publish/subscribe framework for rich content formats," *IEEE Transactions on Parallel and Distributed Systems,*, vol. 23, no. 5, pp. 844–852, 2012.

[17] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu, and G. Fox, "Twister: a runtime for iterative mapreduce," in *Proceedings of the ACM HPDC*, 2010.

[18] C. L. Morefield, "Application of 0-1 integer programming to multitarget tracking problems," *IEEE Transactions on Automatic Control*, vol. 22, no. 3, pp. 302–312, 1977.

[19] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas, *Dynamic programming and optimal control*. Athena Scientific Belmont, MA, 1995, vol. 1, no. 2.

[20] J. Leskovec and J. J. Mcauley, "Learning to discover social circles in ego networks," in *Advances in neural information processing systems*, 2012, pp. 539–547.

[21] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy, and B. Y. Zhao, "User interactions in social networks and their implications," in *Proceedings of the ACM EuroSys*, 2009, pp. 205–218.

APPENDIX

A. Proof of Theorem 4

Let $\lambda_1, \lambda_2, \cdots, \lambda_k$ be the sequence of subscription aggregations selected by the proposed greedy algorithm, and X be the elements selected in the optimal solution. Define the curvature of the sub-modular cost C to be $\rho = min\frac{\sum_{\lambda \in X} C(\lambda)}{C(X)}$. Note that if C is linear, then $\rho = 1$.

Set $X_0 = \phi$, and $X_i = \{\lambda_j : 1 \leq j \leq i\}$ for each $1 \leq i \leq k$. Denote $\mu_0 = \frac{C_i}{\Delta_{x_i} G(X_{i-1})}$ for each $1 \leq i \leq k$. The parameter μ_i is related to the effective factor of x_i for each $1 \leq i \leq k$. We claim that $\mu_0 \leq \mu_1 \leq \cdots \leq \mu_k$.

This is because that

$$\mu_i = \frac{C(\lambda_i)}{\Delta_{\lambda_i} G(X_{i-1})} \leq \frac{C(\lambda_{i+1})}{\Delta_{\lambda_{i+1}} G(X_{i-1})} \leq \frac{C(\lambda_{i+1})}{\Delta_{\lambda_{i+1}} G(X_i)} = \mu_{i+1} \tag{3}$$

where the first inequality follows from the greedy rule and the second inequality follows from the sub-modularity of f.

Define X is the optimal solution. Then, $\sum_{e \in X} C(e) = \rho \cdot opt$. Set $X_0 = \phi$ and $X_i = \{\lambda_j : 1 \leq j \leq i\}$ for each $1 \leq i \leq k$. For each $0 \leq i \leq k$, let $l_i = \theta - G(X_i)$ be the remaining target at the end of the iteration i.

$$\begin{aligned} &\frac{l_{i-1} - l_i}{C(\lambda_i)} = \frac{\Delta_{\lambda_i} G(X_{i-1})}{C(\lambda_i)} \\ &\geq max_{e \in X} \frac{\Delta_e G(X_{i-1})}{C(e)} \geq \frac{\sum_{e \in X} \Delta_e G(X_{i-1})}{\sum_{e \in S} C(e)} \\ &\geq \frac{G(X) - G(X_{i-1})}{\sum_{e \in X} C(e)} = \frac{\theta - G(X_{i-1})}{\rho \cdot opt} = \frac{l_{i-1}}{\rho \cdot opt} \end{aligned} \tag{4}$$

Thus, the other inequality $C(\lambda_i) \leq l_{i-1} - l_i$ follows from the assumption that $\Delta_{\lambda_i} G(X_{i-1})/C(x_i) > 1$, based on the reality.

Since $l_0 > l_1 > \cdots > l_k = 0$. Using the inequalities

$$C(\lambda_i) \leq \frac{\rho \cdot opt}{l_{i-1}} (l_{i-1} - l_i) \tag{5}$$

For $1 \leq i \leq t+1$, we have

$$\begin{aligned} &\sum_{i=1}^{max} C(\lambda_i) + \frac{l_t - opt}{l_t - l_{t+1}} C(\lambda_t + 1) \\ &\leq \rho \cdot \int_{opt}^{l_0} \frac{1}{y} dy = \rho \cdot opt \cdot ln \frac{l_0}{opt} = \rho \cdot opt \cdot ln \frac{\theta}{opt} \end{aligned} \tag{6}$$

Using the inequalities $C(\lambda_i) \leq l_{i-1} - l_i$ for $t+1 \leq\leq k$, we have

$$\begin{aligned} &\frac{opt - l_{t+1}}{l_t - l_{t+1} c(\lambda_{t+1})} + \sum_{i=t+2}^{k} C(\lambda_i) \\ &\leq opt - l_{t+1} + \sum_{i=t+2}^{k} (l_{i-1} - l_i) \\ &= opt - l_k \leq opt \end{aligned} \tag{7}$$

By sum the Eqs. 6 and 7, we can get the following result.

$$\sum_{i=1}^{k} C(\lambda_i) \leq (1 + \rho ln \frac{\theta}{opt}) opt \leq (1 + \rho ln \theta) opt. \tag{8}$$

$\frac{r_{i-1} - r_i}{C_i} = \frac{G_i}{C_i} \geq max_{\lambda \in S} \frac{\Delta_\lambda f(X_{i-1})}{C(\lambda)} \geq \frac{\sum_{\lambda \in S} \Delta_\lambda f(X_{i-1})}{\sum_{\lambda \in S} C(\lambda)} \geq \frac{f(S) - f(X_{i-1})}{\sum_{\lambda \in S} C(\lambda)} = \frac{\theta - f(X_{i-1})}{\rho \cdot opt} = \frac{r_{i-1}}{\rho \cdot opt}$