

Min-Cost Matchmaker Problem in Distributed Publish/Subscribe Infrastructures

Zihui Ge, Ping Ji, Jim Kurose and Don Towsley
Computer Science Department,
University of Massachusetts at Amherst,
{gezihui,jiping,kurose,towsley}@cs.umass.edu

Abstract

The publish/subscribe (pub/sub) paradigm provides content-oriented data dissemination, where communication channels between content providers and content consumers are set up on the basis of interest matches between content provided by the publishers and content requested by the subscribers. In this paper, we study a distributed matchmaker system which resides on the data dissemination path, intercepts both publication announcements and subscription requests so as to match publishers to interested subscribers. We formalize an optimization problem, namely the Min-Cost Matchmaker problem, whose objective is to minimize the number of publication/subscription messages being maintained and transmitted throughout the system. We study the complexity of the Min-Cost Matchmaker problem and propose algorithms to solve the problem efficiently. Our simulation results show that a system implementing such a Min-Cost Matchmaker structure can significantly reduce the number of overall publication/subscription messages maintained/transmitted compared to existing systems where either publication announcements or subscription requests are broadcast. These results cast light on a new direction of designing a more efficient matchmaker structure for publish/subscribe systems.

1 Introduction

The publish/subscribe (*pub/sub*) paradigm provides content-oriented data dissemination, where communication channels between content providers and content consumers are set up on the basis of interest matches between content being provided and requested. In such a paradigm, publishers (i.e., content providers) announce the data specifications, such as data topics or some other content attributes, before data flow actually comes. In the meantime, subscribers (i.e., content consumers) reveal their interests to the network, and to the potential publishers, so as to be able to receive data in which they have declared an interest. A matchmaker system, consisting of a set of distributed “mediators”, can then sit along the data dissemination paths between the publishers and subscribers, intercepting both publication announcements and subscription requests so as to facilitate communication channel setup and scoped content delivery (for example, through setting up data filters [19, 15]). Such an architecture is very useful in building distributed simulations [19], large scale event notification systems [4, 5], etc.

Due to their inherent reliance on group communication, most pub/sub systems [19, 16, 3, 18] are built on top of IP multicast or overlay multicast [11]. The matchmaker system can therefore reside in active routers or intermediate relay nodes, matching publication and subscription messages and triggering the setup of communication channels. However, all currently proposed systems require either publication announcements or subscription messages to be propagated throughout the network. Although either scheme provides sufficient information for matchmakers to function, neither is optimal in terms of total messages being propagated and recorded throughout the system.

In this paper, we present a new scheme where, by marking each link on the multicast dissemination paths as either a publish link or a subscribe link, both publication messages and subscription messages can be pruned in the middle of the network. We formalize an optimization problem — the Min-Cost Matchmaker problem, whose objective is to minimize the number of publication/subscription messages being maintained and transmitted throughout the system. Under two different scenarios, one where the underlying multicast dissemination topology is a single shared-tree and the other where the data is disseminated over per-source trees, we study the complexity of the Min-Cost Matchmaker problem and propose algorithms that solve the problem efficiently. We evaluate the benefit of utilizing a “min-cost matchmaker” over using broadcast publications or broadcast subscriptions through simulations of various system configurations. Our results show that using a min-cost matchmaker can significantly reduce the overall number of pub/sub messages transmitted and stored and can scale well with both a large number of publishers and a large number of subscribers.

The remainder of this paper is organized as follows. In Section 2, we describe and formalize the Min-Cost Matchmaker problem in general multicast-based pub/sub systems. In Section 3, we present polynomial time algorithms for the Min-Cost Matchmaker problem for both shared and per-source based multicast topologies. Our simulation settings and simulation results are presented in Section 4. In Section 5, we address issues related to implementing the min-cost matchmaker and integrating it into a pub/sub system. Finally, in Section 6, we conclude our work and describe related future research.

2 Problem Description and Formalization

2.1 Problem Description

As mentioned in the Introduction, most pub/sub systems are built on top of IP multicast or overlay multicast; publication announcements and subscription requests, contained in signaling messages, are propagated and maintained at the intermediate nodes and the end hosts on the multicast data dissemination paths. These intermediate nodes and end hosts, comprising the *matchmaker* system, can conduct matches among the upstream publication announcements and downstream subscription requests, and trigger the process of setting up communication channels between publishers and interested subscribers. We will discuss implementation issues related to integrating the matchmaker into pub/sub systems in Section 5.

Previous studies [5, 13, 14] of pub/sub systems take for granted that either publish announcements or subscription requests should be broadcasted. However, as we will see, a matchmaker system utilizing both publication and subscription messages can prune unnecessary publication announcements and subscription

requests, thus potentially reducing the overall signaling messages propagated and maintained throughout the system. To accomplish this, the matchmaker system marks each directed link within the multicast tree as publish link (p-link) or subscribe link (s-link). All publish messages are required to be propagated through p-links, and all subscribe messages are required to be propagated through s-links. For each publish/subscribe message being propagated through a link, the publication announcement/subscription request contained in the message is then stored at the end node (router or host) across the link. Having ensured this, publish messages do not need to propagate through downstream s-links, and subscribe messages do not need to propagate through upstream p-links.

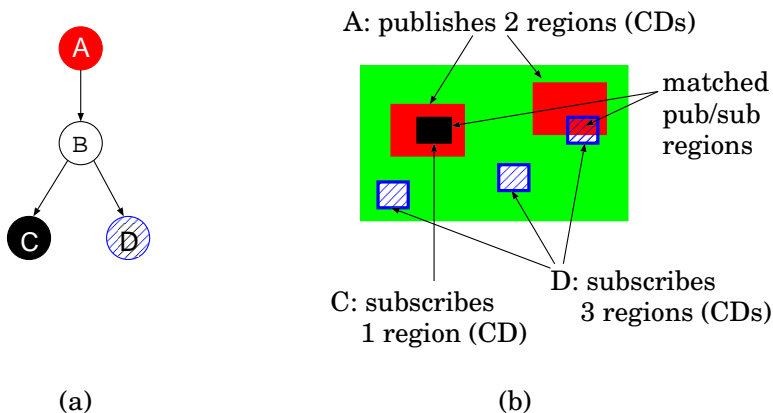


Figure 1: An example of content descriptors (CDs) as geometric regions in a two-dimensional virtual world

Different edge-markings result in different message transmission and storage overheads. To evaluate these overheads, let us consider what is contained in a pub/sub message. Consider a distributed simulation system simulating a 2-dimensional virtual world [19]. A simulation unit (e.g., an entity in the simulation) may only be interested in the events that occur in the area within a certain distance from it. Thus the geometric description of this area is included in its subscription requests. Similarly a unit's publication on some geometric area indicates that its taking actions (i.e. pending publishing data) are regarding to this area. Figure 1 shows an example illustrating such a scenario. Figure 1(a) is a simple multicast topology with one publisher (*A*) and two subscribers (*C* and *D*). Figure 1(b) illustrates the 2-dimensional world, and the regions that *A* is generating data in and the regions that *C* and *D* are subscribing to. Note that one publisher can announce multiple regions, and one subscriber can also subscribe to multiple regions. This is due to the fact that multiple simulation units can be simulated on the same host. More generally, in this paper, we define such descriptive information on predefined variables/categories (e.g., the geometric regions) that determines the boundaries of published/subscribed contents as a piece of **content descriptor (CD)**. As the size of each CD is approximately constant within one system, the number of CDs being propagated and stored throughout the system serves as a useful measure of the overhead. Consequently, one can naturally pose an optimization problem concerned with identifying the edge-marking scheme that requires the least overhead. We illustrate this optimization problem with an example in Figure 2.

Figure 2 compares several different edge-marking schemes for the scenario described in Figure 1. The small rectangles represent the CDs that *A* publishes or *C* and *D* subscribe to. In Figure 2(a), all the links are marked as s-links. This corresponds to a *broadcast subscription* scheme where all of the subscribed

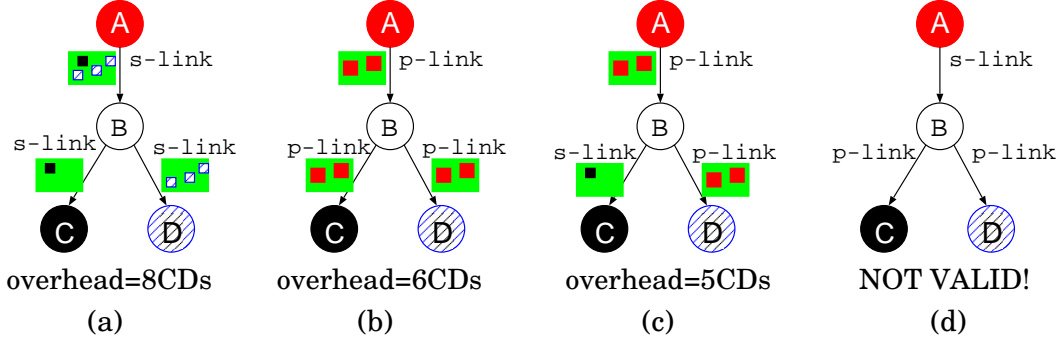


Figure 2: Examples of edge-marking schemes

CDs from subscriber C and D are propagated upstream toward the publisher. We observe that the *broadcast subscription* scheme introduces an overhead of 8 CDs. When a *broadcast publication* scheme is used, as in Figure 2(b), the publish messages generated by A are transmitted toward all of the subscribers. Thus, the overhead is 6 CDs. Figure 2(c) shows another edge-marking scheme, where links AB and BD are marked as p-links and BC is marked as a s-link. The total overhead in this case is reduced to 5 (4 published CDs and 1 subscribed CD). In fact, this edge-marking requires the least overhead, thus it is the *optimal* edge-marking that we are looking for. This example demonstrates the potential benefits of utilizing a “smart” edge-marking scheme as oppose to using *broadcast publication* or *broadcast subscription* schemes.

We notice that not all the edge-markings are valid. To ensure that a matchmaker system works correctly, the publish messages should be able to meet subscribe messages at some node along the data dissemination path from any publisher to any subscriber. This node can thereafter conduct matches between pub/sub messages and trigger establishing the data dissemination path for the matched contents. In terms of valid edge-marking schemes, this requires that, along any data dissemination path, there exists a switchover node whose upstream links are all marked as p-links and whose downstream links are all marked as s-links. Figure 2(d) shows a counter example, where the edge-marking configuration is *invalid*.

As we have already observed, the edge-markings are related to the underlying multicast topologies. Currently, there are two major different multicast routing architectures. In the next subsection, we will briefly describe their distinctions.

2.2 Shared Tree v.s. Per-Source Tree Multicast Architecture

There are two well-known approaches that determine the multicast topology used by applications: the shared tree and the per-source based tree. The shared tree scheme uses a single tree to distribute the traffic for all senders, whereas the per-source based scheme constructs a source specific routing tree for each individual sender.

In the shared tree approach, only a single routing tree is constructed for the entire multicast group. For example, the single multicast tree shown with thicker shaded lines in Figure 3(a), connects routers A , B ,

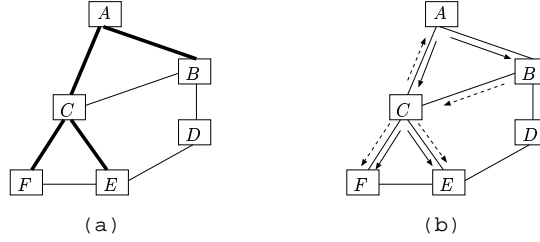


Figure 3: A single, shared tree (a), and two per-source based trees (b)

C, E, and F. Multicast packets flow only over the shaded links. Note that the links of a shared multicast tree are bi-directional, since packets can flow in either direction on a link. Core-based Tree (CBT) [2] and PIM-Sparse Mode [6] are two shared-tree routing protocols.

Under the per-source based approach, an individual routing tree is constructed for each sender in the multicast group. In a multicast group with N hosts, N different routing trees are constructed. Packets are routed to multicast group members in a source-specific manner. In Figure 3(b), two source-specific multicast trees are shown, one rooted at A and another rooted at B . Note that not only are there different links than in the group-shared tree case (for example, the link from BC is used in the source-specific tree rooted at B , but not in the group-shared tree in Figure 3(a)), but that some links may also be used only in a single direction. PIM-Dense Mode [1] and PIM-Sparse Mode with the option of using source-specific shortest path trees [6] are source-based multicast routing protocols.

Here we should mention that, since the links of both kinds of multicast trees can be bi-directional, one need to mark each direction of a link separately and independently when considering the edge marking schemes.

2.3 Model and Notation

In this section, we formalize the min-cost matchmaker problem in publish/subscribe infrastructures. We first consider the case where the multicast data dissemination uses source specific multicast trees. As we shall see in Section 3.2, the shared multicast tree structure can be described as a more constrained version of this problem formalization.

A publish/subscribe system consists of:

- A network $G = (V, E)$ where V is the set of end hosts and routers and E is the set of directed links.
- A set, $P \subseteq V$, of publishing nodes that provide information.
- A function, $f : P \rightarrow \mathbb{N}^+$, that maps each publishing node to the number of CDs that it publishes, i.e., $\forall x \in P, f(x) =$ the number of content descriptors (CDs) that x publishes.
- A set, $S \subseteq V$, of subscriber nodes that request information.

- A function, $g : S \rightarrow \mathbb{N}^+$, that maps each subscriber node to the number of content descriptors (CDs) that it is interested in, i.e., $\forall y \in S, g(y) =$ the number of CDs that y is interested in.
- A set \mathcal{T} , of multicast trees, where $T_x \in \mathcal{T}$ is the set of directed edges that compose the multicast tree for publisher x , i.e.,

$$T_x \subseteq E : \forall e \in T_x, e \text{ is within the source-based multicast tree rooted at } x, \text{ where } x \in P.$$

- Partial order relations that represent the multicast data dissemination tree for each publisher.

$$\mathfrak{R}_x \subseteq T_x \times T_x : \forall (t, q) \in \mathfrak{R}_x, t \text{ is an upstream link of } q \text{ in the multicast tree of } x, \text{ where } x \in P.$$

Note that \mathfrak{R}_x is a transitive relation, i.e., $\forall r, q, t \in T_x, (r\mathfrak{R}_x q \wedge q\mathfrak{R}_x t) \rightarrow r\mathfrak{R}_x t$.

Our goal is to find an edge marking function that maps the directed links within any multicast tree to $\{0, 1\}$,

$$\mathcal{M} : T \rightarrow \{0, 1\}$$

where $T = \bigcup_{x \in P} T_x$ represents the set of directed links that lie within at least one multicast tree for some source(s). For $t \in T$, $\mathcal{M}(t) = 0$ means that t is an s-link and that all subscription messages should be propagated through t , and $\mathcal{M}(t) = 1$ means that t is a p-link and that all publish messages should be propagated through t . This edge marking function must satisfy the following validity constraint:

Validity constraint: Along the data dissemination path from each publisher to each subscriber, the publish messages should meet the subscribe messages at some place, i.e., p-links connecting to the publisher should join s-links connecting to the subscriber at a switchover point. This requires that, for multicast tree of publisher x ,

$$\mathcal{M}(t) = 0 \rightarrow (\forall q \in E), t\mathfrak{R}_x q \Rightarrow \mathcal{M}(q) = 0 \quad (1)$$

or equivalently,

$$\mathcal{M}(t) = 1 \rightarrow (\forall q \in E), q\mathfrak{R}_x t \Rightarrow \mathcal{M}(q) = 1$$

We define the cost of an edge marking scheme as the number of CDs either published or subscribed through all the directed links within the multicast trees, i.e.,

$$Cost(\mathcal{M}) = \sum_{t \in T} (\sigma^t(\mathcal{M}) + \tau^t(\mathcal{M})) \quad (2)$$

where for link $t = (m, n) \in T$, σ^t is the number of published CDs on link t , and τ^t is the number of subscribed CDs on link t . More precisely,

$$\sigma^t(\mathcal{M}) = \begin{cases} 0, & \text{if } \mathcal{M}(t) = 0 \\ \sum_{x \in P} f(x) \cdot \mathbf{1}(x \in U_x^t(\mathcal{M})), & \text{if } \mathcal{M}(t) = 1 \end{cases} \quad (3)$$

Here $\mathbf{1}(\cdot)$ is a predicate function which returns one when the predicate is true and zero otherwise, and U_x^t is defined as

$$U_x^t(\mathcal{M}) = U_x^{(m,n)}(\mathcal{M}) = \{u \mid \text{there is a path consisting of only p-links between } u \text{ and } m \text{ on tree } T_x\} \quad (4)$$

i.e.,

$$U_x^{(m,n)}(\mathcal{M}) = U_{x,0}^{(m,n)}(\mathcal{M}) \cup U_{x,1}^{(m,n)}(\mathcal{M}) \cup U_{x,2}^{(m,n)}(\mathcal{M}) \cup \dots$$

where

$$\begin{aligned} U_{x,0}^{(m,n)}(\mathcal{M}) &= \begin{cases} \emptyset, & \text{if } (m,n) \notin T_x \\ \{m\}, & \text{if } (m,n) \in T_x \end{cases} \\ U_{x,i+1}^{(m,n)}(\mathcal{M}) &= \{u | \exists v \in U_{x,i}^{(m,n)}(\mathcal{M}), (u,v) \mathfrak{R}_x(m,n) \wedge \mathcal{M}((u,v)) = 1\} \end{aligned} \quad (5)$$

The total number of subscribed CDs through link t is defined as

$$\tau^t(\mathcal{M}) = \begin{cases} 0, & \text{if } \mathcal{M}(t) = 1 \\ \sum_{y \in S} g(y) \cdot \mathbf{1}(y \in \bigcup_{x \in P} D_x^t(\mathcal{M})), & \text{if } \mathcal{M}(t) = 0 \end{cases} \quad (6)$$

where D_x^t is defined as

$$D_x^t(\mathcal{M}) = D_x^{(m,n)}(\mathcal{M}) = \{v | \text{there is a path consisting of only s-links between } n \text{ and } v \text{ on tree } T_x\} \quad (7)$$

i.e.,

$$D_x^{(m,n)}(\mathcal{M}) = D_{x,0}^{(m,n)}(\mathcal{M}) \cup D_{x,1}^{(m,n)}(\mathcal{M}) \cup D_{x,2}^{(m,n)}(\mathcal{M}) \cup \dots$$

where

$$\begin{aligned} D_{x,0}^{(m,n)}(\mathcal{M}) &= \begin{cases} \emptyset, & \text{if } (m,n) \notin T_x \\ \{n\}, & \text{if } (m,n) \in T_x \end{cases} \\ D_{x,i+1}^{(m,n)}(\mathcal{M}) &= \{v | \exists u \in D_{x,i}^{(m,n)}(\mathcal{M}), (m,n) \mathfrak{R}_x(u,v) \wedge \mathcal{M}((u,v)) = 0\} \end{aligned} \quad (8)$$

Given the above definitions, the **Min-Cost Matchmaker** problem is concerned with finding an edge marking function \mathcal{M} that minimizes the cost function (2) while satisfying the validity constraint (1). In the next section, we will present the complexity analysis for the Min-Cost Matchmaker problem, and describe algorithms that can solve the problem.

3 Algorithms of Min-Cost Matchmaker Problem

The Min-Cost Matchmaker problem has a very large solution space. Since each link in T can be marked 0 or 1 independently, there are 2^k ways of assigning values to $k = |T|$ links. If we only consider the edge markings that satisfy the validity constraint, in the worst case, there will still be a number of solutions that grows exponentially with k (see [8] for details). While this suggests the infeasibility of the brute force search on the whole solution space for large problems, fortunately, there exist polynomial time algorithms that can solve the Min-Cost Matchmaker problem, as we show in the following subsections.

3.1 Properties of the Min-Cost Matchmaker Problem

Before providing the algorithms of Min-Cost Matchmaker problem, we first present several important properties.

- **Property 1:** *There exists at least two valid solutions of the Min-Cost Matchmaker problem.*

We define

$$\begin{aligned}\mathcal{M}^P(\cdot) : \quad & \forall t \in T, \mathcal{M}^P(t) = 1; \\ \mathcal{M}^S(\cdot) : \quad & \forall t \in T, \mathcal{M}^S(t) = 0.\end{aligned}$$

Here, \mathcal{M}^P is the edge marking that marks all the links on multicast trees as p-links, and \mathcal{M}^S is the edge marking that marks all the links on multicast trees as s-links. They correspond to traditional schemes that broadcast either publish announcements or subscription requests respectively. Obviously, both \mathcal{M}^P and \mathcal{M}^S are valid solutions to the Min-Cost Matchmaker problem.

- **Property 2:** *For any valid edge marking scheme, the cost associated with a link is either the cost of that link when all links are marked as p-links or the cost of that link when all links are marked as s-links.*

More precisely, we have the following Lemma.

$$\mathbf{Lemma 1:} \text{ If } \mathcal{M} \text{ is valid, } \forall t \in T, \sigma^t(\mathcal{M}) + \tau^t(\mathcal{M}) = \begin{cases} \sigma^t(\mathcal{M}^P) & \text{if } \mathcal{M}(t) = 1 \\ \tau^t(\mathcal{M}^S) & \text{if } \mathcal{M}(t) = 0 \end{cases}$$

Proof: We first show by induction that if \mathcal{M} is valid, $\forall t \in T, \mathcal{M}(t) = 1 \Rightarrow U_{x,i}^t(\mathcal{M}) = U_{x,i}^t(\mathcal{M}^P)$.

For $t = (m, n)$, by definition, $U_{x,0}^{(m,n)}(\mathcal{M}) = \{m\} = U_{x,0}^{(m,n)}(\mathcal{M}^P)$.

Now assume $U_{x,i}^{(m,n)}(\mathcal{M}) = U_{x,i}^{(m,n)}(\mathcal{M}^P)$. We have

$$\begin{aligned}U_{x,i+1}^{(m,n)}(\mathcal{M}^P) &= \{u | \exists v \in U_{x,i}^{(m,n)}(\mathcal{M}^P), (u, v) \mathfrak{R}_x(m, n) \wedge \mathcal{M}^P((u, v)) = 1\} \\ &= \{u | \exists v \in U_{x,i}^{(m,n)}(\mathcal{M}^P), (u, v) \mathfrak{R}_x(m, n)\} && (\mathcal{M}^P(t) \equiv 1) \\ &= \{u | \exists v \in U_{x,i}^{(m,n)}(\mathcal{M}), (u, v) \mathfrak{R}_x(m, n)\} && (\text{induction assumption}) \\ &= \{u | \exists v \in U_{x,i}^{(m,n)}(\mathcal{M}), (u, v) \mathfrak{R}_x(m, n) \wedge \mathcal{M}((u, v)) = 1\} && (\mathcal{M}(t) = 1, \mathcal{M} \text{ is valid}) \\ &= U_{x,i+1}^{(m,n)}(\mathcal{M})\end{aligned}$$

Thus, $U_x^t(\mathcal{M}) = U_{x,0}^t(\mathcal{M}) \cup U_{x,1}^t(\mathcal{M}) \cup U_{x,2}^t(\mathcal{M}) \cup \dots = U_{x,0}^t(\mathcal{M}^P) \cup U_{x,1}^t(\mathcal{M}^P) \cup \dots = U_x^t(\mathcal{M}^P)$.

According to the definition of $\sigma^t(\mathcal{M})$ and $\tau^t(\mathcal{M})$, if \mathcal{M} is valid and $\mathcal{M}(t) = 1$,

$$\sigma^t(\mathcal{M}) + \tau^t(\mathcal{M}) = \sum_{x \in P} f(x) \cdot \mathbf{1}(x \in U_x^t(\mathcal{M})) + 0 = \sum_{x \in P} f(x) \cdot \mathbf{1}(x \in U_x^t(\mathcal{M}^P)) = \sigma^t(\mathcal{M}^P)$$

Similarly, we can see that if \mathcal{M} is valid and $\mathcal{M}(t) = 0$,

$$\sigma^t(\mathcal{M}) + \tau^t(\mathcal{M}) = \tau^t(\mathcal{M}^S)$$

This completes the proof.

We will find Property 2 very useful. Even though it does not reduce the overall solution space, it makes evaluating the cost associated with each valid solution straightforward. Also, understanding this property helps us find efficient algorithms for min-cost matchmaker problem as we will see in Sections 3.2 and 3.3.

3.2 Min-Cost Matchmaker Problem for Shared Multicast Tree

In this section, we focus on a special instance of the Min-Cost Matchmaker problem where the underlying data dissemination structure is a shared multicast tree. In such a system, there is only one data path between any two nodes, regardless of the source. More precisely,

$$\begin{aligned} \forall m, n \in V, \quad & ((m, u_1) \in T \wedge (u_1, u_2) \in T \wedge \cdots \wedge (u_{l_1}, n) \in T \\ & \wedge (m, v_1) \in T \wedge (v_1, v_2) \in T \wedge \cdots \wedge (v_{l_2}, n) \in T) \\ \Rightarrow & (l_1 = l_2 \wedge u_1 = v_1 \wedge u_2 = v_2 \wedge \cdots \wedge u_{l_1} = v_{l_2}). \end{aligned} \quad (9)$$

This implies that for any given links t, q , if t is an upstream link of q on some multicast tree, then for any other multicast tree, t is either not within the tree or it is an upstream link of q , i.e.,

$$(\exists x \in P, t \mathfrak{R}_x q) \Rightarrow (\forall y \in P, t \notin T_y \vee t \mathfrak{R}_y q) \quad (10)$$

It can be shown that (10) can be derived from (9).

The shared tree structure simplifies the Min-Cost Matchmaker problem. We show that a greedy approach given in Figure 4 can find the solution efficiently. Basically, for each link t , this algorithm first computes the number of publish CDs in the case that all links are p-links ($\sigma^t(\mathcal{M}^P)$) and the number of subscribe CDs in the case that all links are s-links ($\tau^t(\mathcal{M}^S)$). After that, each link, $t \in T$, is marked as p-link if $\sigma^t(\mathcal{M}^P) < \tau^t(\mathcal{M}^S)$; otherwise t is marked as s-link.

1. for each $t \in T$
2. compute $\sigma^t(\mathcal{M}^P), \tau^t(\mathcal{M}^S)$
3. for each $t \in T$
4. if $\sigma^t(\mathcal{M}^P) < \tau^t(\mathcal{M}^S)$
5. $\mathcal{M}(t) \leftarrow 1$;
6. else $\mathcal{M}(t) \leftarrow 0$;
7. return \mathcal{M} ;

Figure 4: A greedy approach for Min-Cost Matchmaker problem in shared multicast tree

Claim 1: The greedy approach shown in Figure 4 can find the solution of the Min-Cost Matchmaker problem for a shared multicast tree.

The success of the greedy approach is due to a very nice monotonicity property of the shared multicast tree structure. We state this **Monotonicity property** below.

Property 3: For all $t, q \in T$, if $(\exists x, t \mathfrak{R}_x q)$ then $\sigma^t(\mathcal{M}^P) \leq \sigma^q(\mathcal{M}^P)$ and $\tau^t(\mathcal{M}^S) \geq \tau^q(\mathcal{M}^S)$.

We skip the proof of the Monotonicity property. Interested readers can find the proof in our forthcoming technical report [8].

The correctness of Claim 1 now can be shown by verifying the validity and the optimality of a solution produced by the algorithm in Figure 4. Suppose \mathcal{M} is the solution found by the greedy approach. If some link t is marked as a p-link, $\mathcal{M}(t) = 1$, we know from the algorithm that $\sigma^t(\mathcal{M}^P) < \tau^t(\mathcal{M}^S)$. For any upstream link q and $q \mathcal{R}_x t$ regarding to some publisher x , according to the Monotonicity property $\sigma^q(\mathcal{M}^P) \leq \sigma^t(\mathcal{M}^P) < \tau^t(\mathcal{M}^S) \leq \tau^q(\mathcal{M}^S)$. Thus, $\mathcal{M}(q) = 1$. i.e. q will also be marked as p-link. This proves the validity of the result. The optimality of solution \mathcal{M} is obvious given Property 2. Hence, this establishes that the greedy approach shown in Figure 4 finds the solution of the Min-Cost Matchmaker problem for shared multicast tree.

3.3 Per-source Tree

The greedy approach presented above provides the optimal solution for shared multicast trees. However, it does not work for per-source based multicast trees. This is because the Monotonicity property does not always hold in a per-source based multicast structure. Figure 5 shows an example where the Monotonicity property is violated.

In Figure 5, P_1 and P_2 are publishers, S is a subscriber and A, B, C, D are intermediate routers. The data dissemination tree from P_1 to S is marked using solid lines, and the data dissemination tree from P_2 to S is marked using dashed lines. Suppose that $f(P_1) = 2$, $f(P_2) = 2$ and $g(S) = 3$, we compute the cost of each link when all links are marked as p-links ($\sigma^t(\mathcal{M}^P)$) and when all links are marked s-links ($\tau^t(\mathcal{M}^S)$). The results are listed on the right of Figure 5. Unfortunately, we find that AB is an upstream link of BC ($AB \mathcal{R}_{P_1} BC$), however, $\sigma^{AB}(\mathcal{M}^P) > \sigma^{BC}(\mathcal{M}^P)$, violating the Monotonicity property.

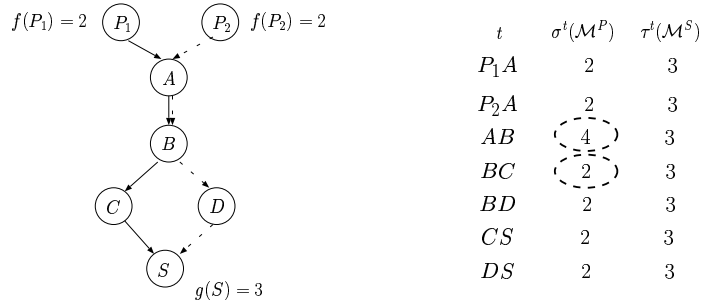


Figure 5: An example of per-source trees where the Monotonicity property does not hold

Realizing that the greedy approach in Figure 4 does not guarantee a valid edge marking for per-source based tree, we find that the Min-Cost Matchmaker problem assuming per-source based multicast tree is much more complicated than that of shared multicast tree. Fortunately, we still can develop a polynomial time algorithm that can solve it. We first describe the intuition behind this algorithm.

Consider a matchmaker initially with all the links marked as p-links. We try to improve this marking scheme by changing some p-links to s-links. From Property 2, we know that remarking a p-link, t , to a s-link will reduce the total number of CDs in the system by η^t , where $\eta^t = \sigma^t(\mathcal{M}^P) - \tau^t(\mathcal{M}^S)$ (negative η^t means an increase in the total number of CDs). Thus, ideally we hope to change all the links with positive

1. for each $t \in T$
2. compute $\sigma^t(\mathcal{M}^P), \tau^t(\mathcal{M}^S), \eta^t \leftarrow \sigma^t(\mathcal{M}^P) - \tau^t(\mathcal{M}^S)$
3. construct bipartite graph $(A \cup B, L)$
 $A = \{t | \eta^t \geq 0\}, B = \{t | \eta^t < 0\}$
 $L = \{(r, q) | r \in A, q \in B, \exists x \in P, r \mathfrak{R}_x q\}$
4. add a flow source Υ and edges $L_1 = \{(\Upsilon, q) | q \in A\}$
5. add a flow sink Ψ and edges $L_2 = \{(q, \Psi) | q \in B\}$
6. construct flow graph $(A \cup B \cup \{\Upsilon, \Psi\}, L \cup L_1 \cup L_2)$
with link capacity $C((r, q)) \leftarrow \begin{cases} \eta^q & , \text{ if } (r, q) \in L_1 \\ -\eta^r & , \text{ if } (r, q) \in L_2 \\ \infty & , \text{ if } (r, q) \in L \end{cases}$
7. $F \leftarrow$ Maximum-Flow on graph $(A \cup B \cup \{\Upsilon, \Psi\}, L \cup L_1 \cup L_2)$
8. for each $t \in A$
9. increase link capacity $C(\Upsilon, t) \leftarrow C(\Upsilon, t) + 1;$
10. $F' \leftarrow$ Maximum-Flow on the new graph;
11. if $(F == F')$
12. $\mathcal{M}(t) \leftarrow 0;$
13. else $\mathcal{M}(t) \leftarrow 1;$
14. recover link capacity $C(\Upsilon, t) \leftarrow C(\Upsilon, t) - 1;$
15. for each $t \in B, \mathcal{M}(t) \leftarrow 1;$
16. for each $(t, q) \in L$
17. if $\mathcal{M}(t) = 0,$ // required by a profitable parent
18. $\mathcal{M}(q) \leftarrow 0;$
19. return $\mathcal{M};$

Figure 6: The algorithm to solve Min-Cost Matchmaker problem in per-source based tree

η^t to s-links. However, due to the validity constraint, if a link is changed to s-link, all its downstream links are forced to be s-links as well. Therefore, what we are really interested in is to find a *profitable* subset (C) of links with positive η value whose overall *gain* ($\sum_{t \in C} \eta^t$) is greater than the sum of the *loss* of their downstream links ($\sum_{t \in C_{\mathfrak{R}}} |\eta^t|$, where $C_{\mathfrak{R}} = \{q | (\exists t \in C)(\exists x \in P) t \mathfrak{R}_x q\}$). To do that, we can construct a bipartite graph where each node corresponds to a link in the matchmaker problem. We place the nodes with positive η^t values on one side and the nodes with negative η^t values on the other side. A directed edge pointing from a positive node to a negative node is added if the positive node is an upstream link of the negative node in the matchmaker problem. We then send flows from the positive side to the negative side through the bipartite graph. Finding a maximum flow will help us find the profitable subset C , in that the maximum flow through C will be limited by its downstream nodes' total loss. Looking for saturated downstream negative nodes will provide enough hints on how to find the subset C . In Figure 6, we present

the pseudo code of our algorithm.

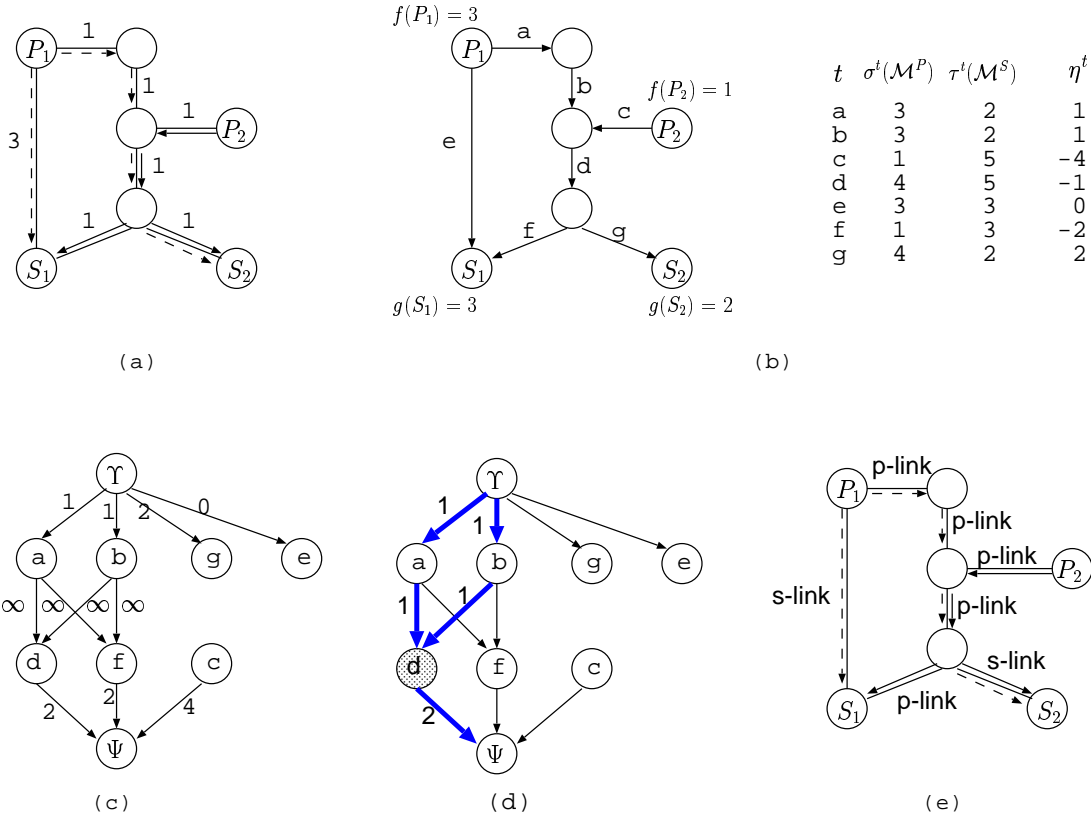


Figure 7: An example illustrating the algorithm for the Min-Cost Matchmaker problem in per-source based multicast tree

An example will help us walk through the algorithm. Figure 7 (a) shows a pub/sub system consisting of two publishers (P_1, P_2) and two subscribers (S_1, S_2). Each link's delay (distance) is marked on the link. The multicast trees for P_1 and P_2 are the single-source shortest paths toward all subscribers, as indicated by the directed dashed lines and directed solid lines respectively. Assume P_1 publishes 3 CDs, P_2 publishes 1 CD, S_1 subscribes to 3 CDs and S_2 subscribes to 2 CDs. In Figure 7 (b), the cost of each links where all links are marked as p-links ($\sigma^t(\mathcal{M}^P)$) and where all links are s-links ($\tau^t(\mathcal{M}^S)$) as well as their difference (η^t) are listed. We now construct a Max-flow problem as shown in Figure 7 (c). Nodes corresponding to links with non-negative η^t — a, b, e, g are placed on the top half of the graph and nodes corresponding to negative η^t — c, d, f are on the bottom. Edges with infinite capacity — $(a, d), (b, d), (a, f)$ and (b, f) are added, capturing their upstream/downstream relations. A single source, Υ , is added with edges connecting to each node on the top. The capacities of these edges are bounded by the corresponding η values of the end nodes. For example, capacity of $(\Upsilon, a) = \eta^a = 1$. Similarly, a single sink, Ψ , is added with edges connecting from each node on the bottom. Then we solve the constructed flow problem. The maximum flow is 2 as shown in Figure 7(d). Since only one node d is saturated, follow the steps from line 8 to line 18 in Figure 6, we can see that only e and g should be marked as s-links. This gives the optimal edge marking solution as shown in

Figure 7(e).

Since the proof of the correctness of this algorithm is long and not our primary focus, we do not include it here. Interested readers can find the proof in [8].

Now we have seen that the Min-Cost Matchmaker problem for per-source multicast tree can be reduced to a Max-flow problem. The good news is that there has been proposed many algorithms that can solve the Max-flow problem efficiently: the fastest algorithm to date is due to Goldberg and Tarjan [10], which achieves a runtime of $O(VE \log(V^2/E))$; the most commonly used algorithm is Ford-Fulkerson method [12] using breadth-first search in finding augmenting path. Edmonds and Karp [7] proved that this strategy yields a polynomial-time algorithm (in $O(VE^2)$). All these algorithms can be used for step 6 of Figure 6, which dominates the complexity of our algorithm. Here, V is the number of nodes and E is the number of edges in the Max-flow problem. We should mention that these bounds are for the worst case performance. As we can see in the example (Figure 7(c)), the flow problem generated usually contains lots of separated subgraphs, which makes the max-flow problem relatively easy to solve. In our simulation studies, as we describe in next section, we experience that solving the Min-Cost Matchmaker problem with realistic network configurations is not computationally expensive.

4 Simulations

In this section, we present our simulations developed to explore the advantages of exploiting edge marking schemes to prune publish announcements and subscription requests in a pub/sub system. Our purpose is to evaluate the benefit of finding a “min-cost matchmaker” as oppose to broadcasting publish announcements or subscription requests in realistic network configurations.

4.1 Simulation Setting

In our simulations, we assume a hierarchical network topology represented by a transit-stub structure. We use the topology generator developed by Georgia Tech [9] to generate such a network topology. The network consists of 100 nodes, 4 of which are transit nodes and the remainder stub nodes. We assume that all the transit nodes and the stub nodes are active and, hence, are able to maintain publication and subscription states and process interest matching. In the simulations, we randomly pick publishers and subscribers from the stub nodes and we do not exclude the possibility that a publisher and a subscriber collocate on the same node. For each publisher or subscriber, we randomly assign a number of CDs which it publishes or subscribes to. The number of CDs is drawn from a *bounded Pareto* distribution given by

$$f(n) = \frac{\alpha k^\alpha n^{-\alpha-1}}{1 - (k/p)^\alpha} \quad k \leq n \leq p$$

with $k = 1$, $p = 100$ and $\alpha = 1.0$. In each simulation, we measure the total number of CDs being propagated and maintained throughout the system. We consider three different schemes: (1) broadcast subscriptions: subscription messages are distributed and stored everywhere (SubE), (2) broadcast publications: publication messages are distributed and stored everywhere (PubE), (3) the optimal configuration derived by solving the

Min-Cost Matchmaker problem (MCM). In the next two subsections, we present our results for shared multicast trees and per-source multicast trees respectively.

4.2 Simulation Results for Shared Multicast Tree

To build a shared multicast data dissemination topology, we choose one of the transit nodes as the “core” and use CBT as the underlying routing protocol.

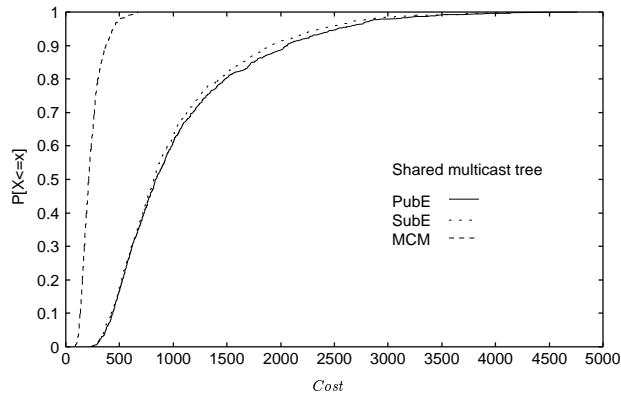


Figure 8: The cumulative distribution function of $Cost$ in shared multicast trees

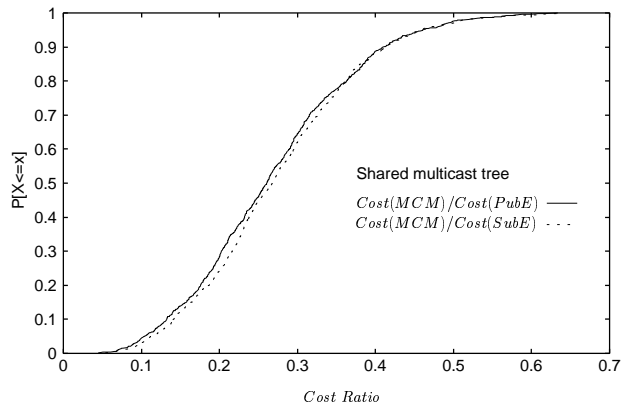


Figure 9: The cumulative distribution function of performance ratios between MCM and PubE/SubE schemes for shared multicast trees

We first fix the number of publishers and the number of subscribers to 10 ($|P| = |S| = 10$) and generate 1000 different samples. Figure 8 plots the cumulative distribution function of the $Cost$ corresponding to PubE, SubE, and MCM respectively. The median value of the $Cost$ of MCM is approximately 250, whereas the medians for PubE and SubE are approximately 800 each. Furthermore, both PubE and SubE have 90-th percentiles of the $Cost$ greater than 2000, whereas, the 90-th percentile of the $Cost$ for MCM is only around 400. Figure 9 shows the cumulative distribution function of the ratio of the $Cost$ for MCM over that for

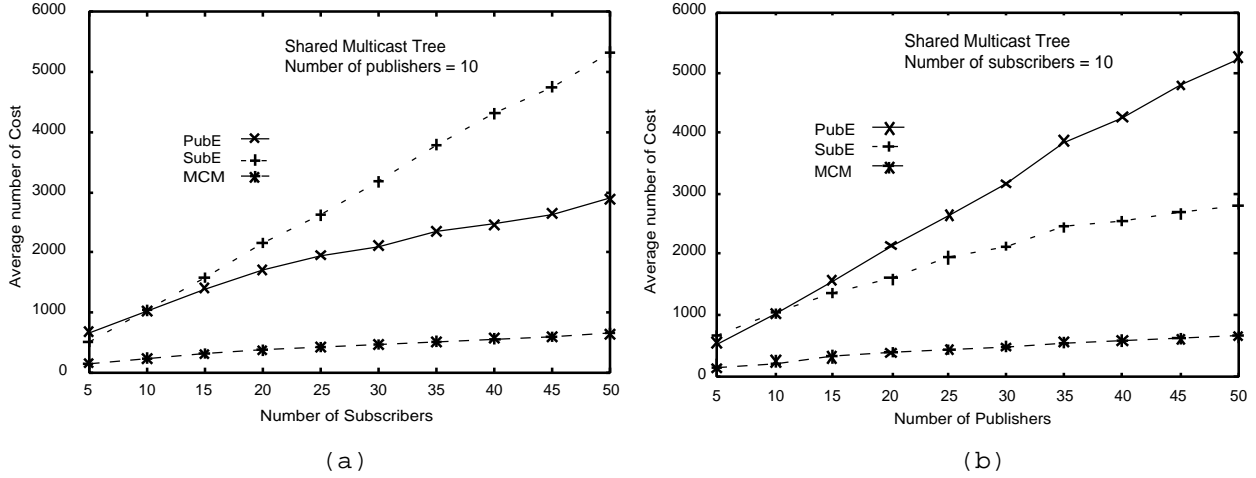


Figure 10: (a) the average value of $Cost$ in shared multicast trees, varying the number of subscribers, setting the number of publishers as 10, and (b) the average value of $Cost$ in shared multicast trees, varying the number of publishers, setting the number of subscribers as 10

PubE and that for SubE. We find that in more than 50% of the samples, MCM provides solutions with the $Cost$ less than 30% of that of either PubE or SubE, and in more than 95% of the samples, MCM can reduce the $Cost$ by at least half comparing to that of either PubE or SubE.

We also investigate scenarios with an asymmetric number of publishers and subscribers. In Figure 10(a), we fix the number of publishers to 10 and vary the number of subscribers from 5 to 50, and in Figure 10(b), we fix the number of subscribers to 10 and vary the number of publishers from 5 to 50. Each data point corresponds to the $Cost$ average of 1000 randomly generated samples. The $Cost$ associated with PubE in Figure 10(a) increases sublinearly when the number of subscribers increases, indicating that PubE is more capable of handling large number of subscribers than SubE. On the other hand, Figure 10(b) indicates that SubE is more scalable with a large number of publishers than PubE. In both figures, MCM exhibits very good scalability both with the number of publishers and the number of subscribers.

4.3 Simulation Results for Per-source Multicast Trees

We conducted the same set of experiments for per-source multicast trees where the data dissemination tree of a publisher is the reverse of the shortest paths from each of the subscribers to the publishers. Figure 11 to Figure 13 present the results. We get the same observations as in shared multicast dissemination tree: using MCM can greatly reduce the total $Cost$ in the system comparing to PubE and SubE, and MCM scales well with both the number of publishers and the number of subscribers.

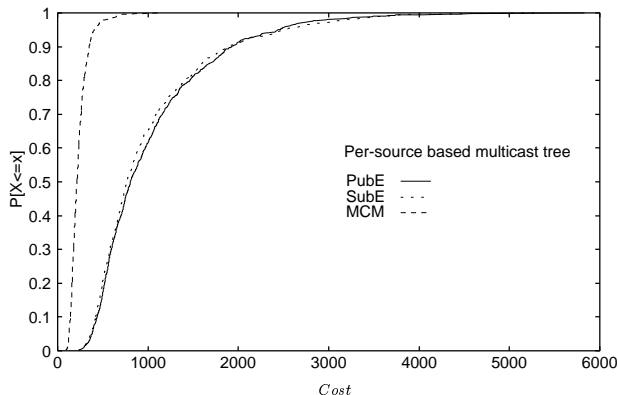


Figure 11: The cumulative distribution function of $Cost$ in per-source trees

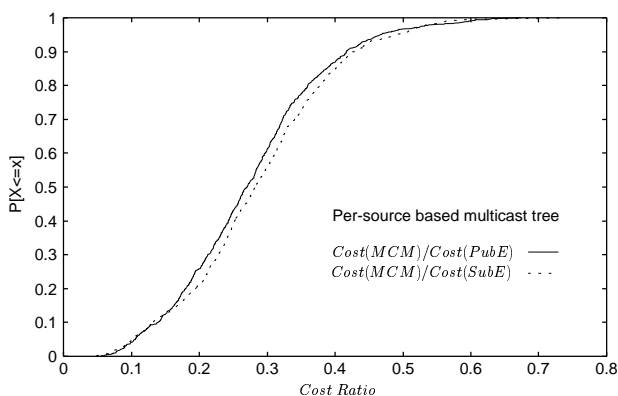


Figure 12: The cumulative distribution function of performance ratios between MCM and PubE/SubE schemes for per-source trees

5 Implementation Considerations

In this section, we address some considerations about implementing the min-cost matchmaker and about integrating it into pub/sub paradigms.

We have already seen how a min-cost matchmaker system relies on a proper marking of the links. However, a link is merely a transmission media with interfaces attached to the routers and hosts. All the states should be maintained at the ends (routers or hosts) of a link. The marking decision (either as p-link or as s-link) of a link thus should be maintained at both ends of the link and should be consistent. In the case where inconsistency happens, a priority should be used to resolve it. For example, one can let the router or host at the upstream end of the data flow determine the correct marking.

When considering a set of edge markings, one should first ensure that the edge markings retain the functionality of the matchmaker system (the validity concern) before considering reducing the number of content

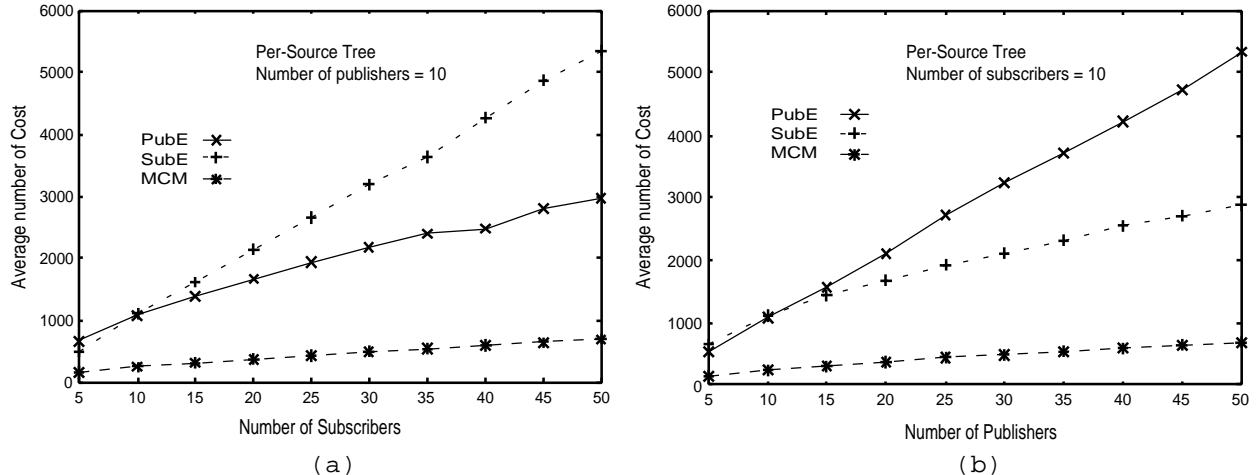


Figure 13: **(a)** the average value of $Cost$ in per-source multicast trees, varying the number of subscribers, setting the number of publishers as 10, and **(b)** the average value of $Cost$ in per-source multicast trees, varying the number of publishers, setting the number of subscribers as 10

descriptors (CDs) being propagated and maintained (the optimality concern). Therefore, when a publisher joins the system, or a node previously existing in the system starts to announce publication messages, all the links grafting this publisher to the rest of the tree should first be marked as p-links. Similarly, all the links grafting a newly joined subscriber should first be marked as s-links. Such procedure preserves the validity of existing markings.

Knowing accurate values of $\sigma^t(\mathcal{M}^P)$ and $\tau^t(\mathcal{M}^S)$ for each link is crucial in determining the markings for both shared multicast trees and source based multicast trees. This requires a hop by hop signaling process to disseminate the information regarding the number of CDs from all the upstream publishers and the number of CDs that are interested by the downstream subscribers to all nodes on the trees. However this is of relatively low overhead in that only the *numbers* rather than the specifications of the CDs are needed. Furthermore, when the specifications of the CDs are updated, while the total number of the CDs remains the same (e.g., in distributed simulation, one unit moving from one location in the virtual world to another updates its previous subscription CD), no additional effort regarding to computing proper markings is needed.

With the knowledge of the correct value of $\sigma^t(\mathcal{M}^P)$ and $\tau^t(\mathcal{M}^S)$ for link t , determining the link marking is easy for shared multicast trees. As indicated by the greedy algorithm shown in Section 3.2, each link can determine the proper marking only based on its local information of $\sigma^t(\mathcal{M}^P)$ and $\tau^t(\mathcal{M}^S)$, and consequently global optimality is reached. This is due to the monotonicity property, which is guaranteed for shared multicast trees. However, for per-source multicast trees, the situation is more complicated. Realizing the algorithm in Section 3.3 requires more than local information. Fortunately, from our simulation, we find that only a small fraction of the randomly generated samples violate the monotonicity property, and within such samples, only a small fraction of nodes are actually involved in such a violation. Thus, once a violation is detected, one of the involved nodes is elected, which gathers all the related information, computes

the min-cost matchmaker configuration and informs other nodes of the result. Such a scheme may involve sophisticated control and complicated computation if many links on different source specific trees are entangled with the violation of the monotonicity property; otherwise the processing and signaling procedure is straightforward.

As the goal of pub/sub infrastructure is to provide interest based data dissemination, the task of a matchmaker system is to conduct matches between publication announcements from the publishers and subscription requests from the subscribers, and to trigger establishing the communication channels. This is achieved by the following steps: (1) compute the edge marking scheme that minimizes the total CDs in the system; (2) propagate and store the publication messages by the publishers and subscription messages by the subscribers according to the edge marking scheme; (3) conduct a match between publish and subscribe CDs and compute the descriptors of matched pub/sub content, which can be different from both publish CDs and subscribe CDs (see the example in Figure 1 (b)); (4) a matchmaker node that detects a match of pub/sub content sends signaling messages, containing the CD for the matched part, both upstream toward the publishers and downstream toward the subscribers. These signaling messages trigger the communication channel setup. In this paper, our focus has been on the first two steps. Meanwhile, many studies has been concerned with designing data structures and algorithms for efficient matching [17, 5]. As for step (4), the communication channel setup, mostly accomplished through content-based routing [18] and data filtering [15, 19], has also been investigated.

We notice that only step (4) involves the *data channel* while other steps are dealing with the *control channel*. However step (4) is only triggered when a match is found in step (3). Thus using a matchmaker structure ensures that the “state installations” for either content-based routing or data filtering happens only when a match of interests between a publisher and a subscriber is detected. Publish announcements that are of no interest to any subscriber and subscription requests that cannot be served have no impact on the *data channel*. This is different from most current designs where the states needed for data forwarding are mixed with the states needed for interest matching. Since data channel is usually considered more loaded than the control channel, being able to save the states associated with the data channel makes the design of matchmaker structure a more attractive approach.

6 Conclusion and Future Work

In this paper, we have presented a distributed *matchmaker* structure for publish/subscribe paradigm. The matchmaker, composed of intermediate nodes and end hosts, resides on the data dissemination path, intercepts both publication announcements and subscription requests so as to match publishers to interested subscribers. While previous studies take for granted that either publish announcements or subscription requests should be broadcasted, our matchmaker system exploiting an edge marking scheme which suppresses both unnecessary publication content descriptors and unnecessary subscription content descriptors in the middle of the network. We have formalized an optimization problem, namely the Min-Cost Matchmaker problem, whose objective is to minimize the number of publication/subscription content descriptors being maintained and transmitted throughout a pub/sub system. Furthermore, we have developed polynomial time algorithms to solve the Min-Cost matchmaker problem under two different scenarios: where the underlying multicast dissemination topology is a shared-tree and where the underlying multicast routing

assumes per-source trees. Our simulation results show that using a min-cost matchmaker can significantly reduce the overall pub/sub content descriptors transmitted and stored, and can scale well with both a large number of publishers and a large number of subscribers.

In our future work, we would like to implement the min-cost matchmaker structure and integrate it into our existing active-network-facilitated large-scale distributed simulation system [19]. We will evaluate the signaling overhead and storage overhead of using the min-cost matchmaker structure, and compare them with the overhead of using broadcast publication and using broadcast subscription schemes.

Acknowledgement

The authors would like to thank Micah Adler for his helpful comments on this work.

References

- [1] Andrew Adams, Jonathan Nicholas, and William Siadak. Protocol independent multicast - dense mode (PIM-DM): Protocol specification (revised), February 2002. INTERNET DRAFT, <http://netweb.usc.edu/pim/internet-drafts/draft-ietf-pim-dm-new-v2-01.txt>.
- [2] T. Ballardie, P. Francis, and J. Crowcroft. Core based trees (CBT), an architecture for scalable inter-domain multicast routing. In *ACM SIGCOMM*, 1993.
- [3] Guruduth Banavar, Tushar Chandra, Bodhi Mukherjee, Jay Nagarajarao, Robert E. Strom, and Daniel C. Sturman. An efficient multicast protocol for content-based publish-subscribe systems. In *Proceedings of the 19th IEEE International Conference on Distributed Computing Systems*, Austin, Texas, June 1999.
- [4] Luis Felipe Cabrera, Michael B. Jones, and Marvin Theimer. Herald: Achieving a global event notification service. In *Proceedings of the Eighth Workshop on Hot Topics in Operating Systems (HotOS-VIII)*, Elmau, Germany, May 2001.
- [5] A. Carzaniga, D. Rosenblum, and A. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19(3), Aug 2001.
- [6] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, and L. Wei. The pim architecture for wide-area multicast routing. *ACM Transactions on Networks*, April 1996.
- [7] Jack Edmonds and Richard M. Karp. Theoretical improvements in the algorithmic efficiency for network flow problems. *Journal of the ACM*, 19:248–264, 1972.
- [8] Zihui Ge, Ping Ji, Jim Kurose, and Don Towsley. Min-cost matchmaker problem in distributed publish/subscribe infrastructures. Technical report, University of Massachusetts at Amherst, 2002.
- [9] Georgia Tech. *Modeling Topology of Large Internetworks*. <http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html>.
- [10] Andrew V. Goldberg and Robert E. Tarjan. A new approach to the maximum flow problem. In *Proceedings of Eighteenth Annual ACM Symposium on Theory of Computing*, pages 136–146, 1986.
- [11] Yang hua Chu, Sanjay G. Rao, and Hui Zhang. A case for end system multicast. In *Proceedings of ACM SIGMETRICS*, Santa Clara, CA, June 2000.
- [12] Jr Lestor R. Ford and D.R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.
- [13] G. Mühl, L. Fiege, and A. Buchmann. Filter similarities in content-based publish/subscribe systems. In *International Conference on Architecture of Computing Systems (ARCS)*, 2002.

- [14] Gero Muhl. Generic constraints for content-based publish/subscribe. In *Proceedings of the 6th International Conference on Cooperative Information Systems (COOPIS)*, 2001.
- [15] Manuel Oliveira, Jon Crowcroft, and Christophe Diot. Router level filtering for receiver interest delivery. In *Second International Workshop on Networked Group Communication (NGC)*, Palo Alto, California, Nov. 2000.
- [16] Lukasz Opyrchal, Mark Astley, Joshua S. Auerbach, Guruduth Banavar, Robert E. Strom, and Daniel C. Sturman. Exploiting IP multicast in content-based publish-subscribe systems. In *Middleware*, pages 185–207, 2000.
- [17] Joao Pereira, Françoise Fabret, François Lirbat, and Dennis Shasha. Efficient matching for web-based publish/subscribe systems. In *Conference on Cooperative Information Systems*, pages 162–173, 2000.
- [18] Antony I. T. Rowstron, Anne-Marie Kermarrec, Miguel Castro, and Peter Druschel. SCRIBE: The design of a large-scale event notification infrastructure. In *Networked Group Communication*, pages 30–43, 2001.
- [19] S. Zabele, M.Dorsch, Z. Ge, P. Ji, M. Keaton, J. Kurose, J. Shapiro, and D. Towsley. Sands: Specialized active networking for distributed simulation. In *DARPA Active Networks Conference and Exposition (DANCE)*, San Francisco, California, USA, May 2002.