Generating Wide-Area Content-Based Publish/Subscribe Workloads

Albert Yu, Pankaj K. Agarwal, Jun Yang
Department of Computer Science, Duke University
{syu, pankaj, junyang}@cs.duke.edu

***Abstract*— Content-based publish/subscribe systems allow events to be selectively and aperiodically pushed to subscribers according to their interests, expressed as predicates in a high-dimensional event space. Making such systems scalable in a wide-area network requires considering multiple factors, e.g., distribution of events, similarity of subscriber interests in the event space, and proximity of subscriber locations in the network. A major obstacle for this research is the lack of publicly available, realistic workloads, because of concerns of privacy and commercial interests in releasing user information. This paper describes a workload generator for wide-area content-based publish/subscribe systems, which extrapolates the limited amount of public data in order to provide a reasonable guess for a large, realistic workload. Our hope is that this tool will help publish/subscribe researchers evaluate their research.**

I. Introduction

Publish/subscribe is a model of data dissemination, where *publishers* (data providers) selectively and aperiodically push *events* to *subscribers* (data consumers) based on their specified interests. Publish/subscribe systems typically employ a network of *brokers* which serve as the middleware between the publishers and the subscribers. Traditionally, publish/subscribe systems are *topic-based*, where subscribers can subscribe only to a set of predefined topics. Recent years, however, have seen growing interests in *content-based* publish/subscribe systems, where subscribers can specify their interests using arbitrary predicates in a multi-dimensional event space.

We have been working on a system called *ProSem* [1, 2] for efficiently supporting powerful content-based subscription functionalities. A main feature of ProSem is joint consideration of *subscription processing* and *notification dissemination*. Traditionally, these problems are considered separately: the database community has mostly focused on efficient subscription processing, while the network community has mainly focused on efficient event dissemination. ProSem is along the recent line of work, including *ONYX* [4], *SemCast* [7], *Net-χ* [8], etc., aimed at bridging this gap. Our goal is to develop an end-to-end solution consisting of not only techniques for subscription processing and indexing but also dissemination network design.

A particular problem of interest is how to assign subscriptions to brokers. Brokers are intermediaries in delivering events from publishers to subscribers. Intuitively, it is beneficial to assign subscriptions with similar interests to the same broker, because events delivered to the broker serve multiple subscriptions, potentially saving communication. On the other hand, we need to be careful in letting one broker handle subscribers that are far away in terms of network distance, because doing so may violate delivery latency requirements and increase communication costs. Balancing the two considerations—similarity of interests in the event space and proximity of locations in the network space—is a tricky optimization. The optimal trade-off between the two also depends on the volumes of events matching shared versus disjoint interests. Therefore, the best solution for a given system must take into account subscription interests and locations as well as event distributions.

A major obstacle for this research is the lack of publicly available, realistic workloads for content-based publish/subscribe. The industry rarely discloses data on subscriptions (user interests and locations) because of privacy concerns and commercial interests. Lack of widely deployed systems supporting powerful content-based subscriptions also contributes to the difficulty in getting large and real workloads.

To tackle this deficiency, we are developing a workload generator for wide-area content-based publish/subscribe systems. We use publicly available data extracted from Google Groups to extrapolate subscribers' interests and geographical locations. We then generate actual network locations of subscribers and brokers based on PlanetLab measurements. Although the raw data is not from content-based publish/subscribe systems, we believe our extrapolation provides a reasonable guess of what a large, realistic content-based publish/subscribe workload looks like. Users can control some aspects of the workload generation to fine-tune the process and get workloads with varying sizes and characteristics. We plan to make the generator publicly available so that publish/subscribe researchers can use it to evaluate their research.

II. Overview

In general, events in our system are drawn from a d-dimensional *event space* $\mathbb{R}^d$. Each event specifies one value for each dimension, and corresponds to a point in the event space. Each subscription defines a region of interest within this space; every event that falls within this region will need to be delivered to the subscriber. Currently, for our workload generator, the event space has two dimensions, and each subscription's region of interest is a rectangle in $\mathbb{R}^d$ (i.e., a conjunction of two range predicates, one in each dimension). Although it is straightforward to extend our extrapolation techniques to higher-dimensional event space, raw data beyond

two dimensions is difficult to find, partly because of lack of widely deployed systems supporting higher-dimensional subscription predicates.

Network locations are modeled as a t-dimensional *network space* $\mathbb{R}^t$, using network embedding techniques [6, 3]. Each network location is mapped to a point in the network space, such that the latency between two network locations can be approximated by the Euclidean distance between the corresponding points in this space. Subscriptions and brokers are located all over the world, and their network locations are mapped to points in the network space. Our workload generator allows the number of network space dimensions to be adjusted.

To recap, the workload we generate consists of the following: a) a set of subscriptions, each having a rectangular region of interest in the event space and a point location in the network space; b) an event distribution over the event space, from which a sequence of random events can be drawn; c) a set of brokers, with their point locations in the network space.

Given the limited amount of public data to extrapolate from, we have been conservative in not generating workloads with too many features. In Section V, we discuss several possible extensions. The next two sections cover the two work phases of our generator. Section III discusses how to extract raw data from publicly available sources. Section IV discusses how to generate a workload using the raw data as basis for extrapolation.

III. Raw Data Extraction

Data from Google Groups Google Groups allows people with common interests to form and participate in a discussion group. Google's group directory (`http://groups.google.com/groups/dir`) tags each group with three attributes: topic (e.g., "soccer"), language (e.g., "Spanish"), and geographic region (e.g., "Europe"). For each group, Google provides its number of members. Google also defines a hierarchy over topics (and regions, but not languages). For instance, "recreation" contains "sports" which contains "soccer." We say a topic is *atomic* if it does not contain any subtopics.

We base our subscription workload on data extracted from Google Groups. Two of the three attributes, topic and language, are treated as dimensions of the event space. We make each pair of atomic topic and language a *base interest*. There are 268 atomic topics and 141 language, giving rise to $37{,}788$ potential base interests. Intuitively, each member in a group associated with a base interest counts as a subscription with that interest. Since Google Groups is topic-based rather than content-based, we will need to convert the categorical event space dimensions into continuous ones, and map base interests into rectangles in $\mathbb{R}^2$; Section IV has the details.

The third attribute of a Google group, geographic region, gives an idea of where its members are located in the network space. For each base interest, we divide all Google groups associated with the interest by their geographic region, and count the total number of members for each geographic region.

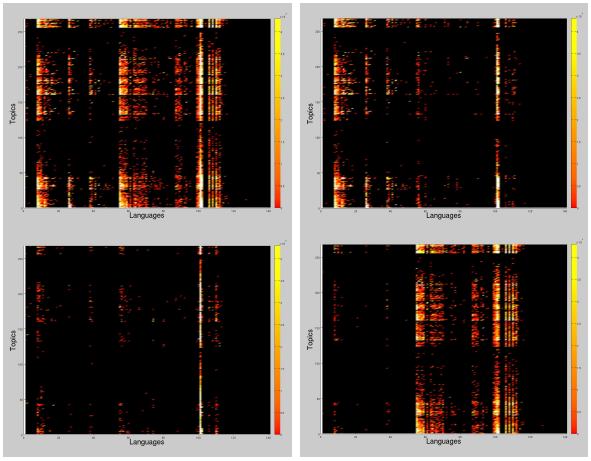

Fig. 1. Correlation between interests and geographic regions. Upper-left: all geographic regions; upper-right: Asia; lower-left: US; lower-right: Europe.

Intuitively, these counts provide a rough indication of the distribution of subscribers over the network.

The correlation between interests and geographic regions is depicted in Figure 1. The upper-left heat map shows the distribution of all subscription interests over the event space. The other maps show the subscription interests within three different geographic regions. We see that Europeans subscribe to many languages, while the US subscribers are only interested in a few languages. On the other hand, both seem to be interested in more topics than Asians. Although subscribers from the three geographic regions share some common interests, correlation between interests and geographic regions is clearly visible.

These data characteristics provide useful insights into system design. Suppose that each broker summarizes its subscriptions using a "super-interest" containing all of them (a common practice by many systems). As shown in the lower-left heat map, US subscription interests spread sparsely over the event space. If a publish/subscribe system assigns subscriptions to brokers based on network proximity alone, we would end up with large super-interests, causing many events to be delivered to brokers unnecessarily. On the other hand, if we ignore network proximity, an Asian subscriber may be assigned to an European broker since the Asian and European subscribers happen to share some common interests. In this case, latency may become unacceptable.

Google Groups also provides, for each group, the average number of messages posted per month. We use this information to obtain event distribution. The rate of events matching a base interest is approximated by the total number of messages posted to groups associated with this interest.

Data from PlanetLab Although data from Google Groups gives us a rough distribution of subscriptions by geographic region, we still need actual network locations for subscriptions as well as brokers. To this end, we choose a subset of PlanetLab nodes, measure their inter-node latencies, and embed this latency relationship in a low-dimensional Euclidean

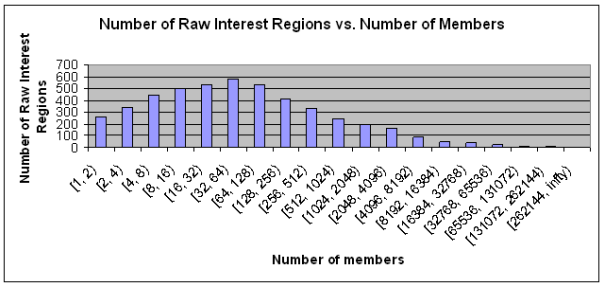

Fig. 2. Popularities of base interests.

space. Many techniques and tools are available for performing this embedding (e.g., [6, 5]). We used [5] with the number of dimensions set to 5; it is easy to replace this default with a command line argument. The resulting coordinates in the network space, together with their geographic region destinations, are then used to generate subscription and broker locations, as described in the next section.

IV. Workload Generation

This section discusses how we generate a wide-area content-based publish/subscribe workload. We subject the raw data extracted in Section III to a series of transformations, some optional, allowing us to extrapolate from fundamentally simpler workloads, remove idiosyncrasies in the raw data, and offer users additional control over the workload being generated.

Hot-Interest Removal The popularity of base interests, measured by the total membership size of their constituent Google groups, is heavily skewed. As shown in Figure 2, a few interests contain a significant fraction of members. The top 3 interests contain at least 2^{18} members each, the next 9 contain 2^{17} to 2^{18} members each, and the next 11 contain 2^{16} to 2^{17} members each. There are roughly 8.1 million members total. By removing these top 24 interests, the number of members reduces to about 4.3 million. Figure 3 illustrates that even as we keep on removing the most popular interests, popularity distribution remains skewed; a small fraction of interests will contain a large faction of members. Removing the most popular interests do have some effect on the subscription distribution across geographic regions. As shown in Figure 4, the percentage of European subscribers gradually increases as more popular interests are removed. The main reason is that a large number of US (and to a lesser extent, Asian) subscribers are members of very popular interests. Nonetheless, distribution of interests within each geographic region and the correlation between interests and geographic regions (figures omitted due to limited space) remain similar to what we observe in Figure 1.

Our workload generator provides an option to remove those most popular interests. It allows users to control the size of the workload, while still retaining most distributional characteristics of the original data.

Interest Smoothing As we have seen, the distribution of interests over the event space is quite jagged. We give an

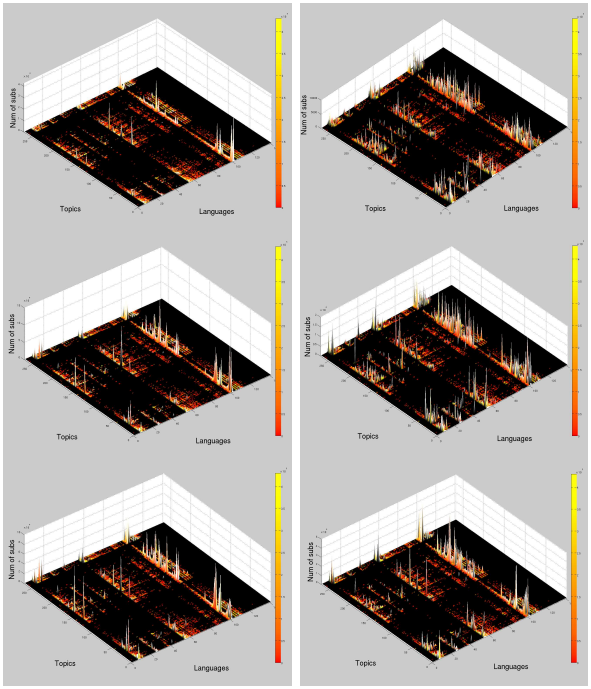

Fig. 3. Relative popularities of base interests, as the most popular ones (at least t members) are removed. Start from the upper-left plot in a counter-clockwise direction: $t = \infty$, $t = 150,000$, $t = 100,000$, $t = 50,000$, $t = 20,000$, $t = 10,000$.

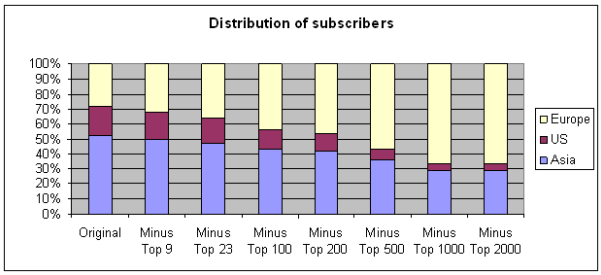

Fig. 4. Distribution of subscribers over geographical regions, as the most popular interests are removed.

option to smooth this distribution. The intuition is that interest in one topic implies possible interest in related topics as well. We capture the notion of "relatedness" by leveraging the topic hierarchy provided by Google Groups. Google Groups does not provide a hierarchy on languages, so we introduce our own by grouping languages into categories according to their origins. For example, "Asian languages" contains "Eastern Asian languages" which contains "Korean."

To perform smoothing, we proceed top-down from the roots of the two hierarchies. Consider the pair (t, l), where t is an non-atomic topic with child topics $t_1, \ldots, t_m$, and l is a language category with child categories or languages $l_1, \ldots, l_n$. Let $C(t, l)$ denote the total subscription count under (t, l), i.e., over all base interests with topics under t and languages under l. We lower the variance among the subscription counts of all (t, l)'s "immediate sub-pairs," i.e., $\{C(t_i, l_j) \mid 1 \leq i \leq m \text{ and } 1 \leq j \leq n\}$, as follows. We

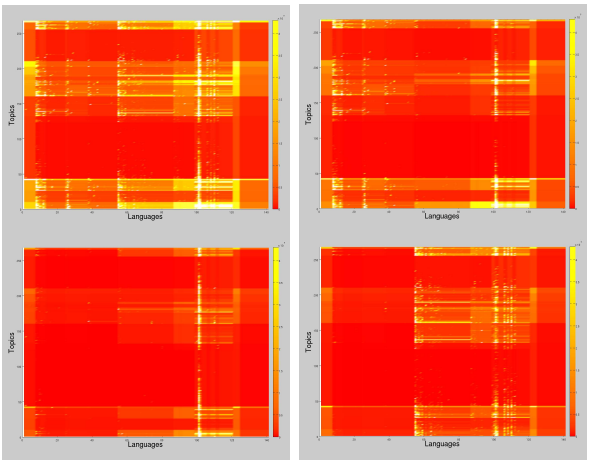

Fig. 5. Distribution of interests over the event space, after smoothing ($p = 0.95$). Upper-left: all geographic regions; upper-right: Asia; lower-left: US; lower-right: Europe.

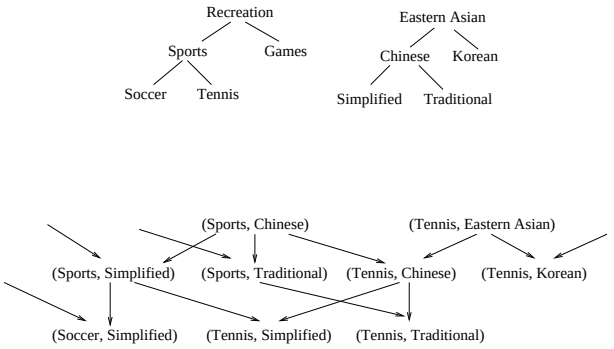

Fig. 6. Part of the Hasse diagram for the poset of interests, constructed from the two hierarchies above.

set each $C(t_i, l_j)$ to $p \cdot C(t_i, l_j) + (1-p) \cdot \frac{C(t,l)}{mn}$, where p is a user-specified smoothing parameter between 0.5 and 1, by adjusting the subscription counts of base interests under (t_i, l_j) proportionally. The same procedure is then recursively applied to each of the sub-pairs.

This smoothing procedure ensures that base interests with more related topics and languages in their respective hierarchies have more influence over each others' subscription counts than those less related. Note that subscription counts for different geographic regions are smoothed separately. Figure 5 shows how subscription interests look like after smoothing is applied.

Interest Generalization In practice, we should expect some subscribers to have broader interests than others; e.g., some may be interested in sports in general, while other may be interested only in soccer. Google Groups indeed allows a group to be tagged with a general, non-atomic topic (or even multiple topics). However, this practice seems rare; general topics have very few (if any) groups of itself when those of its subtopics are excluded.

Therefore, our workload generator obtains more general interests by extrapolating from base interests. We first construct a partially ordered set (poset) of interests from the set of base interests as follows. Let T denote the set of topics in the topic hierarchy (including both atomic and non-atomic ones), and let L denote the set of languages and language categories in the language hierarchy. The set of interests (which includes all base interests) is given by $T \times L$. We define a partial order $\leq$ on the set of interests: Given two interests (t_1, l_1) and (t_2, l_2), we have $(t_1, l_1) \leq (t_2, l_2)$ iff 1) t_2 is an ancestor of (or the same as) t_1 in the topic hierarchy, and 2) l_2 is an ancestor of (or the same as) l_1 in the language hierarchy. Figure 6 illustrates the construction of the poset of interests.

Recall that the data from Google Groups has, for each base interest, subscription counts by geographic region. We provide two options for extrapolating subscription counts for ancestor interests in the poset. The first option assumes that a (user-specifiable) fraction of the subscriptions to a particular interest are actually meant for a parent interest in the poset. If there are multiple parent interests, the fraction is divided equally among them. Starting from the base interests, we propagate subscription counts upwards in (increasing) topological order.

The second option takes a different view that, in reality, subscribing to a general interest implies subscribing to all its child interests in the poset; counts for base interests from Google Groups reflect contributions from ancestor interests. One possible reconstruction of subscriptions counts for ancestor interests works as follows. We again start from the base interests and work our way upwards in topological order. For every non-base interest I, we compute $\delta = \min_{J \text{ is a child of } I} \{\frac{\text{count assigned to } J}{\text{number of } J\text{'s parents}}\}$. We assign to I a (user-specifiable) fraction of δ, and deduct the same quantity from all children of I.

Subscription counts for different geographic regions are propagated separately, so every non-base interest will have a count for each geographic region, just like the base interests.

Subscription Generation After interest generalization, we now have subscription counts by geographic region for all interests. We now discuss how to generate a subscription workload where each subscription has a rectangular region of interest in $\mathbb{R}^2$ and a point location in the network space.

Recall that the domain of interests is $T \times L$, where both T and L are hierarchies. To convert an interest to a rectangle in $\mathbb{R}^2$, we map elements of a hierarchy to intervals in $\mathbb{R}$ such that: 1) If x is y's parent in the hierarchy, then x's interval contains y's interval. 2) For any x in hierarchy, intervals for x's children in the hierarchy all have the same length, and adjacent intervals are separated by gaps of the same length. An example is shown in Figure 7. The gaps between adjacent intervals ensure a desirable property: Given x, y, and z in the hierarchy, if the lowest common ancestor of x and y is a descendant of the lowest common ancestor of x and z (i.e., if y is closer to x than z in the hierarchy), then the gap between x and y's intervals is smaller than the gap between x and z's intervals.

Given a subscription from a geographic region R, to generate its point location in the network space, we use the

Fig. 7. Hierarchy of intervals.

coordinates of PlanetLab nodes obtained in Section III. We provide two options. For the first option, we randomly draw a point from inside the minimum enclosing box (or convex hull) of all PlanetLab nodes from R. For the second option, we first randomly select a PlanetLab node from R; then, we randomly draw a point from the vicinity of the selected node.

For the full subscription workload, we would generate, for each pair of interest I and geographic region R, as many random points in the network space as the subscription count for I in R. The subscription count is rounded to the next smallest integer if it is a fractional value due to the smoothing and interest generalization steps. Each random point becomes a subscription at this network location with the rectangular region of interest converted from R.

For users who want smaller subscription workloads, we provide a sampling tool for generating a subset of the subscriptions. With our highly skewed interest distribution, naive random sampling is inadequate when the sample size is small, since it tends to miss subscriptions of less popular interests. Unrepresentative samples are problematic for system evaluation, because they make subscriptions more homogeneous than they should be. To alleviate this problem, we implement stratified sampling. We group interests into strata by popularity: those in the same stratum have similar subscription counts. We then sample from each stratum, drawing a number of subscriptions proportional to the total subscription count of that stratum.

Event Generation The message count data we extracted from Google Groups in Section III gives the event rate matching each base interest. Our continuous event space, however, has gaps between regions that correspond to base interests. To obtain event rates for these gaps, we use kernel smoothing, which redistributes some of the event density at each location in $\mathbb{R}^2$ to its vicinity. The resulting distribution over the event space allows us to generate a random sequence of events with characteristics similar to those in the extracted raw data.

Broker Generation To generate network locations for a set of brokers, we simply pick them at random from the coordinates of PlanetLab nodes obtained in Section III. By default, the number of brokers we pick from a geographic region is proportional to the number of subscriptions from that region. If there are not enough PlanetLab nodes in a particular geographic region, we can obtain additional coordinates in the region using the techniques discussed above for generating subscription locations.

V. Conclusion and Future Work

In this paper, we have presented a generator for wide-area content-based publish/subscribe workloads. We have shown how to extrapolate the limited amount of public data to obtain large workloads resembling realistic ones. For future work, there are many useful extensions, such as more details on event publishing (e.g., where events originate), changes to event distributions and subscriptions over time, subscriptions beyond multi-dimensional range predicates, etc. We hope this workload generator will become a useful tool to the publish/subscribe research community.

References

[1] Badrish Chandramouli, Junyi Xie, and Jun Yang. On the database/network interface in large-scale publish/subscribe systems. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, pages 587–598, Chicago, Illinois, USA, June 2006.

[2] Badrish Chandramouli, Jun Yang, Pankaj K. Agarwal, Albert Yu, and Ying Zheng. ProSem: Scalable wide-area publish/subscribe. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, Vancouver, Canada, June 2008.

[3] Frank Dabek, Russ Cox, Frans Kaashoek, and Robert Morris. Vivaldi: a decentralized network coordinate system. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 15–26, 2004.

[4] Yanlei Diao, Shariq Rizvi, and Michael J. Franklin. Towards an internet-scale XML dissemination service. In *Proceedings of the 2004 International Conference on Very Large Data Bases*, pages 612–623, Toronto, Canada, September 2004.

[5] Jonathan Ledlie, Peter Pietzuch, and Margo Seltzer. Stable and accurate network coordinates. In *Proceedings of the 2002 International Conference on Distributed Computing Systems*, page 74, Vienna, Austria, July 2002.

[6] T. S. Eugene Ng and Hui Zhang. Predicting internet network distance with coordinates-based approaches. In *Proceedings of the 2002 IEEE International Conference on Computer Communications*, New York, New York, USA, June 2002.

[7] O. Papaemmanouil and U. Cetintemel. SemCast: Semantic multicast for content-based data dissemination. In *Proceedings of the 2005 International Conference on Data Engineering*, Tokyo, Japan, April 2005.

[8] Praveen Rao, Justin Cappos, Varun Khare, Bongki Moon, and Beichuan Zhang. Net-χ: unified data-centric internet services. In *NETB'07: Proceedings of the 3rd USENIX international workshop on Networking meets databases*, pages 1–6, 2007.