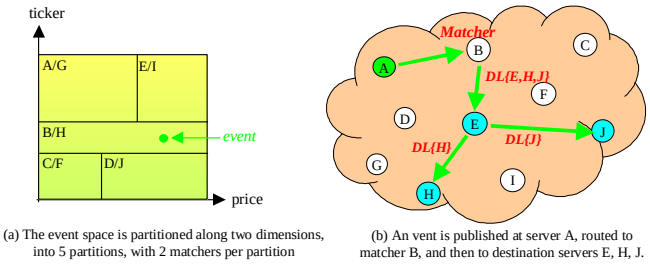
MEDYM: An Architecture for Content-based Publish-Subscribe Networks

Fengyun Cao Jaswinder Pal Singh
Princeton University
{fcao, jps} @cs.Princeton.edu

Content-based publish-subscribe (*pub-sub*) is an important paradigm for asynchronous communication between entities in a distributed network. Users *subscribe* to conditions of interest on future *events*, and are notified when events satisfying those conditions are *published* to the system. Subscriptions can be very expressive, specifying complex filtering criteria. Such timely, highly customized information delivery is valuable to many applications, such as personalized information dissemination, distributed system monitoring, alerting and notification, and application integration.

An Internet-scale pub-sub network consists of a set of pub-sub servers distributed over the Internet. Servers accept events as well as subscriptions from clients of the pub-sub network. We focus on the problem of efficient event delivery from the server at which the event is published to all servers with matching subscriptions. This is a challenging problem, as traditional group-based multicast techniques are not readily applicable. First, pub-sub communication is guided by content rather than network addresses. An event has to be matched with subscriptions to understand *where* it should be sent. Second, it is not clear *how* to route the event, even if the destinations are known. Due to the diversity of content-based subscriptions, different events can match subscriptions from different sets of servers. In the worst case, the total number of such sets can be $2^{\#servers}$, and it is impractical to maintain a multicast group for each such possible set.



(a) The event space is partitioned along two dimensions, into 5 partitions, with 2 matchers per partition

(b) An vent is published at server A, routed to matcher B, and then to destination servers E, H, J.

Figure 1. An example MEDYM network.

Based on the properties of content-based pub-sub communication, we propose a novel pub-sub network architecture called *MEDYM*: *Match Early with DYnamic Multicast*. Fig. 1 shows an example MEDYM network. The event space is partitioned into non-overlapping partitions with balanced load (see (a)). The partitioning algorithm can be data-type dependent (e.g. [5]) and can be combined with considerations of geographic locality, matching capability, etc. Every pub-sub server acts as a *matcher* for one or more partitions. Every partition has one or more matchers, for redundancy and sharing of matching load. Subscriptions are routed to all matchers that are responsible for partitions with which they overlap, so that a given event has to be matched at only one server, and early on so unncessary event propagation is stemmed. A matcher needs to store only those subscriptions that overlap with its event partition(s).

As shown in (b), a published event is first forwarded to a nearby matcher whose partition covers the event. The matcher generates a *destination list* (*DL*) of all *destination servers* with matching subscriptions. At this point, no more matching needs to be done, and the problem is converted to one of routing. In MEDYM, every event message carries a DL, a list of IDs of servers to which the receiver of the message is responsible for event delivery. Based on the DL and knowledge of network topology, a server receiving an event runs a *dynamic multicast routing algorithm*, which computes next-hop servers for the event and partitions the incoming DL into smaller DLs for each next-hop server. The event, with the corresponding new DLs, is then forwarded to the next-hop servers. In this way, a transient, stateless *dynamic multicast tree* grows to span all destination servers. Such distributed decision-making is highly flexible and robust, as servers can optimize routing decisions based on various routing policies and adapt to network environment changes and node or link failures in real time. Since the DL is partitioned at every forwarding step, the average DL size per message in a dynamic multicast tree is only O(log *#destination servers*).

Another major advantage of MEDYM is that events are routed through only the minimum set of servers that actually have matching subscriptions (except for the matcher), thus minimizing total event traffic load in the network, and distributing the load consistent with server self-interests. To the best of our knowledge, no other pub-sub network architecture has achieved this property.

We compared MEDYM with two major existing approaches, the Content-based Filtering (CBF) approach (simulated as in [1][1]) and the Channelization approach (simulated as in [4]). Detailed results can be found in our poster, at http://www.cs.princeton.edu/~fcao/poster.pdf.

Simulation results show that event routing in MEDYM is highly efficient. For example, an event that match subscriptions from 1% servers is routed through 6% servers in CBF, and 28% in Channelization, while only through the 1% matched servers in MEDYM. The maximum server bandwidth consumption of CBF is 58 times that in MEDYM, and of Channelization 37 times; the maximum link stress of CBF is 6 times that in MEDYM, and of Channelization 13 times. Content space partitioning also keeps subscription maintenance cost low, as subscriptions are only replicated and updated at matchers with overlapping partitions. The advantages of MEDYM are greatest for large networks and high subscription selectivity, exactly the scenarios where efficient content-based pub-sub is most valuable.

The overheads MEDYM introduces, such as the DL overhead and dynamic routing computation cost, turn out to be acceptable and more than outweighed by its benefits. For example, to route an event to 1,000 destinations, the average DL has only 18 server IDs and the routing algorithm we developed runs in under 0.5ms at the first server and decreases quickly thereafter. Parallelism can help even further.

We have implemented a prototype of MEDYM on PlanetLab and plan to deploy a publicly available pub-sub service using it. The current MEDYM architecture is expected to scale to at least a few thousand servers, which is more than adequate for most foreseeable applications. We are exploring the use of hierarchical structures or DHT techniques for further scalability. More information about our work can be found at http://www.cs.princeton.edu/DADI/.

REFERENCES

[1] A. Carzaniga, D. Rosenblum, and A. Wolf, "Design and evaluation of a wide-area event notification service," In *ACM TOCS.*, 2001.
[2] A. Carzaniga, A.L. Wolf, "Forwarding in a Content-Based Network". In Proc. of *ACM SIGCOMM* 2003.
[3] A. Carzaniga, M.J. Rutherford, and A.L. Wolf, "A Routing Scheme for Content-Based Networking". In Proc. of *IEEE INFOCOM* 2004.
[4] A. Riabov, Z. Liu, J. Wolf, P. Yu and L. Zhang, "Clustering Algorithms for content-based publication-subscription systems," In Proc. of *ICDCS* 2002.
[5] Y. Wang, L. Qiu, et. al, "Subscription Partitioning and Routing in Content-based Publish/Subscribe Networks". In Proc. of *DISC*'02.

[1] We observe that routing information described in [2,3] can result in redundant event routing, which was not addressed in [2,3]. Private correspondence with author A. Carzaniga confirms this observation, and we are working with him to understand the tradeoff between redundant routing and state needed, which is unclear at this stage. Therefore, we simulated [1] as representative for the CBF approach.