Subscription-Driven Self-Organization in Content-Based Publish/Subscribe *

Roberto Baldoni Roberto Beraldi Leonardo Querzoni Antonino Virgillito
Dipartimento di Informatica e Sistemistica
Università di Roma “La Sapienza”
Via Salaria 113, 00198, Roma, Italy
email: {baldoni, beraldi, querzoni, virgi}@dis.uniroma1.it

Abstract

In this paper we outline a novel approach for enhancing content-based routing performance in publish/subscribe systems composed by an application-level network of event brokers. We present an algorithm which aim is to logically place close (in terms of TCP hops) brokers that manage subscribers with similar interests. We show how this algorithm allows the content based routing algorithm to work in a favorable scenario, thus increasing its global performance.

1. Introduction

Content-based publish/subscribe (pub/sub) communication systems are a basic technology for many-to-many event diffusion over large scale networks with a potentially huge number of clients. The high flexibility of the subscription language makes content-based pub/sub not immediately scalable to systems with a high number of producers (publishers), consumers (subscribers) and a high event publication rate. Scalable solutions are obtained considering a network of event brokers linked through transport-level connections and dispatching events from publishers to subscribers.

The content-based routing (CBR) algorithm introduced in [2] enhances scalability by reducing the number of TCP hops experienced by an event during its diffusion toward the intended subscribers. Unfortunately the CBR algorithm performs at its best only when similar subscriptions are hosted at brokers which are directly connected through TCP links.

In this paper we propose a self-organizing algorithm executed by brokers, which dynamically arranges TCP connections between pair of brokers in order to maximize an associativity metric among them. Associativity measures the degree of similarity of the subscriptions managed by a broker with the ones of its neighbors. The algorithm uses this heuristic to put users sharing similar interests as close as possible even though they are connected to distinct brokers. Self-organization allows the CBR algorithm to works close to its best-case scenarios creating thus the conditions for a reduction of TCP hops metric per event diffusion. The heuristic is also network aware: it allows the establishing of a TCP connection between two brokers only if it does not harm network level performance in term of either TCP link latency or IP hops.

2. A Self-Organizing Algorithm

In this section we briefly describe the self-organization algorithm. A detailed description of the algorithm can be found in [1]. We consider a content-based pub/sub system structured as a set of processes, namely event brokers, interconnected through transport-level links which form an acyclic topology. During the diffusion of an event through the brokers’ network, each broker acts as forwarder for the event; if a broker is not directly interested in the event, i.e. it doesn’t contains a subscription matched by the event, it acts as a simple “event router”: we call such a broker a pure forwarder. A pure forwarder introduces one useless TCP hop on the event diffusion path and requires two useless event matching operations, one for the local subscriptions and one for the forwarding step.

The algorithm aims at reducing the number of pure forwards encountered by an event during its diffusion across the brokers’ network, toward the intended recipients. The basic idea of our approach is to rearrange the topology of the network in order to directly connect through a TCP link brokers hosting ‘similar’ subscriptions without affecting network-level metrics. This allows to reduce the TCP hops experienced by an event to reach all its intended destinations. Lowering the number of such TCP hops has a positive impact on the scalability of the system: each broker has to process a lower number of messages, both for the

---

* This work is partially supported by the european project EU-Publi.com funded by the European Community and by the italian projects MAIS and IS-MANET funded by the Italian Ministry of Research.
matching and for the forwarding operations. However, it is important to remark that reducing the TCP hops per notification diffusion through a topology rearrangement could not necessarily lead to an improvement wrt network level metrics such as IP hops, latency and bandwidth. Hence, the self-organization algorithm is also network-aware because it arranges a new connection only when this does not harm network-level routing performance.

Similarity among subscriptions is measured through the associativity metrics. The subscription model used in content-based publish/subscribe is very generic, including operators such as range, prefixes, etc. In general such subscriptions can be represented as geometric hyper-rectangles. The calculus of the associativity between two subscriptions is an heuristic based on the geometrical intersection of the respective hyper-rectangles.

Algorithm Behavior. The algorithm follows this basic simple heuristic: each broker tries to rearrange the network in order to obtain an increment of its associativity $AS(B)$ while not decreasing the total associativity of the system $AS$. Practically this is realized as follows: a self-organization is triggered by a broker $B$ when it detects (through the reading of the CBR tables) that there could be some other broker $B'$, not directly connected to $B$ through a TCP link, that could increase its associativity. The aim of the self-organization is: (i) to connect $B$ to $B'$ and (ii) to tear down a link in the path between $B$ and $B'$ in order to keep the topology of the network acyclic (a requirement of the CBR algorithm). While point (i) leads to an increment of $AS(B)$, point (ii) can at the same time potentially lead to a decrease of $AS$. Therefore the algorithm does its best to select the link in the path between $B$ and $B'$ that has the lowest associativity between the two brokers it connects.

Self-organization takes place only if it leads to an increase of $AS$. Increasing the associativity of the whole pub/sub system, intuitively means increasing the probability that two brokers with common interests get close to each other. The algorithm can be split in four phases: triggering, tear-up link discovery, tear-down link selection, and reconfiguration which includes the CBR routing tables update:

- **The triggering phase.** The algorithm is triggered by a broker $B$ when it detects the possibility of increasing its associativity. Specifically this can happen when $B$ suspects that behind one of its a link there could be a broker $B'$ which can increase its associativity. The triggering phase starts several independent discovery procedures, depending on the number of links which satisfy the previous condition. Each of these procedures returns the broker with the highest associativity that is located behind the link.

- **Tear-Up Link Discovery.** For each discovery procedure a request message is sent along the corresponding link. The message is forwarded by a broker if there is the possibility that a higher associativity can be discovered with a broker behind one or more of its links. This can be determined by the local structures at a broker. Otherwise the broker replies to the message sending backward a reply message. Eventually the reply belonging to the broker $B'$ that can assure the highest associativity is returned to $B$.

- **Tear-Down Link Selection.** The aim of this phase is to select the link that has to be teared down during the reconfiguration phase. This link is selected among those constituting the path between $B$ and $B'$. Specifically the link connecting brokers with the lowest associativity is chosen. If the associativity of the new link that has to be created between $B$ and $B'$ is higher that the associativity of chosen link the algorithm passes to the last phase.

- **Reconfiguration.** During this last phase the two brokers $B$ and $B'$ are connected through the creation of a new link, while the link chosen in the previous phase is teared down. After this topology change, data structures at brokers have to be updated. In order to avoid network partitioning during this phase is critical to avoid concurrent self-organizations involving common links. To this aim the algorithm locks the path connecting $B$ and $B'$ before starting the reconfiguration phase.

3. Conclusions

Providing a pub/sub system with self-organizing capability is a novel approach to the enhancement of the scalability of systems with a low physical locality. In this paper we introduced a self-organizing algorithm that works in synergy with a content-based routing (CBR) protocol. This algorithm is based on a dynamic network of brokers and tries to self-arrange the network so that brokers managing subscriptions with high similarity will be as close as possible in the network graph.

We performed an extensive simulation study, realized by implementing a full content-based pub/sub system enhanced with the self-organization algorithm. Experimental results (reported in [1]) show clearly that (i) increasing the associativity produces a reduction in the TCP hops metric and (ii) the self-organizing algorithm allows to reduce the overall application-level network traffic when the rate of event publications is roughly twenty times higher than subscriptions rate (a common situation for pub/sub systems).

References
