R. Baldoni  R. Beraldi  L. Querzoni  A. Virgillito

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

e-mail: {baldoni, beraldi, querzoni, virgi}@dis.uniroma1.it

Background

In content-based publish/subscribe systems subscriptions and events are defined over a predetermined event schema, composed by a set of n attributes. The event schema can be graphically represented as an n-dimensional space where events are embodied by points and subscriptions by subspaces; an event matches a subscription if the corresponding point falls inside the subspace defined by the subscription.

A distributed event notification service is composed by a set of processes, namely event brokers, interconnected through transport-level links which form an acyclic topology. Clients can issue subscriptions and publish events. Events are diffused through the system from publisher to interested subscribers using routing tables whose content is based on previously issued subscriptions.

The problem

If a broker is not directly interested in the event it acts as a simple router: it can publish a “pure forwarder”. A pure forwarder introduces one useless TCP hop on the event diffusion path and requires two useless event matching operations.

The idea

The basic idea of our approach is to rearrange the topology of the brokers’ network in order to directly connect brokers hosting similar subscriptions without affecting network-level metrics. This network rearrangement should lead to a reduction of the average number of “pure forwarders” experienced during event diffusion.

The algorithm

Main goal of the algorithm is to put close brokers hosting similar subscriptions inside the application-level network. To measure the similarity between two subscriptions we use an heuristic, namely associativity, based on the geometrical intersection of the respective areas (hyper-rectangles shapes) occupied in the event space. Each broker B tries to rearrange the network in order to obtain an increment of its associativity AS(B) while not decreasing the global associativity of the system AS. Self-organization takes place only if it leads to an increase of AS increasing the associativity of the whole publish/subscribe system, intuitively means increasing the probability that brokers sharing 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 content-based routing tables update.

1 Triggering: 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 links there could be a broker B’ which can increase its associativity AS(B). This can be done by the broker simply looking at its content-based routing table.

2 Tear-Up Link Discovery: a discovery request message DREQ is sent along the link. The message is forwarded by a broker if there is the possibility that a higher associativity can be obtained with a broker behind one or more of its links. This phase returns to B (via DREP messages) the address of the broker that can assure the highest associativity.

3 Tear-Down Link Selection: the aim of this phase is to select the link that has to be torn 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 than the associativity of the chosen link the algorithm passes to the last phase.

4 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 torn down. After this topology change, data structures at brokers along the path have to be updated. In order to avoid network partitioning, due to concurrent self-organizations, the path connecting B and B’ remains locked during all this phases.

Results

A prototype of the system has been implemented and exploited for a simulation study of the self-organization algorithm, which confirms that, when limiting the scope of neighbor discovery to a reasonable value (in terms of network latency), the self organization algorithm achieves significant improvement on the TCP-hop metric with respect to a pure content-based routing algorithm, without affecting notifications’ average network latency. Moreover, the overhead introduced by the self-organization algorithm can be amortized, with respect to the usage of the content-based routing algorithm alone, only if there are at least, on the average, more than 20 notifications for each subscription (this is actually a day-life working condition for a publish/subscribe system).

For further details on the algorithm and the simulation study please refer to: