Stochastic Analysis of Hierarchical Publish/Subscribe Systems

Gero Mühl[1], Arnd Schröter[1], Helge Parzyjegla[1], Samuel Kounev[2], and Jan Richling[1]

[1] Communication and Operating Systems Group, TU Berlin, Germany
[2] Software Design and Quality Group, University of Karlsruhe (TH), Germany
{g_muehl,arnd.schroeter,parzyjegla,skounev}@acm.org,
richling@cs.tu-berlin.de

Abstract. With the gradual adoption of publish/subscribe systems in mission critical areas, it is essential that systems are subjected to rigorous performance analysis before they are put into production. However, existing approaches to performance modeling and analysis of publish/subscribe systems suffer from many limitations that seriously constrain their practical applicability. In this paper, we present a generalized method for stochastic analysis of publish/subscribe systems employing identity-based hierarchical routing. The method is based on an analytical model that addresses the major limitations of existing work in this area. In particular, it supports arbitrary broker overlay topologies and allows to set workload parameters, e.g., publication rates and subscription lifetimes, individually for each broker. The analysis is illustrated by a running example that helps to gain better understanding of the derived mathematical relationships.

Keywords: Publish/Subscribe, Performance Analysis.

1 Introduction

Publish/subscribe systems were originally motivated by the need for loosely-coupled and asynchronous dissemination of information in distributed event-based applications. With the advent of ambient intelligence and ubiquitous computing, many new applications of publish/subscribe systems have emerged. The main advantage of publish/subscribe is that it makes it possible to decouple communicating parties in time, space, and flow [2].

In a publish/subscribe system, clients can take over the roles of publishers or subscribers depending on whether they act as producers or consumers of information. *Publishers* publish information in the form of *notifications*, while *subscribers* express their interest in specific notifications by issuing subscriptions. Subscriptions are usually defined as a set of constraints on the type and content of notifications and are often referred to as *filters*. A *notification service* delivers published notifications to all subscribers that have issued matching subscriptions. In many cases, the notification service is implemented by a set of *brokers* each

H. Sips, D. Epema, and H.-X. Lin (Eds.): Euro-Par 2009, LNCS 5704, pp. 97–109, 2009.

managing a set of *local clients*. The brokers are connected by *overlay links* and a published notification is routed stepwise from the publisher hosting broker over intermediate brokers to all brokers that host subscribers with matching subscriptions. To achieve this, each broker manages a *routing table* that is used to forward incoming notifications to neighbor brokers and local clients. The routing tables are updated according to a *routing algorithm* (e.g., identity-based or covering-based routing [3]) by propagating information about subscriptions in the broker network in form of *control messages*.

With the increasing popularity of publish/subscribe and its gradual adoption in mission critical areas, performance issues are becoming a major concern. To avoid the pitfalls of inadequate Quality of Service (QoS), it is essential that publish/subscribe systems are subjected to rigorous performance analysis before they are put into production. Existing approaches to performance analysis of publish/subscribe systems suffer from some significant limitations. Most research in this area is focused on specific system configurations that are evaluated by means of time- and resource-intensive simulations. Such evaluations are expensive especially when large-scale systems with multiple alternative configurations and workloads have to be considered. We discuss related work in detail in Sect. 5.

In [4], we derived closed form equations for the routing table sizes and the message rate in publish/subscribe systems based on hierarchical identity-based routing. However, this approach required several simplifications and restrictive assumptions seriously limiting its practical value. For example, the broker topology was assumed to be a complete n-ary tree and publishers were only allowed to be connected to leaf brokers. Furthermore, subscriptions were assumed to be equally distributed among filter classes and brokers.

In this paper, we build on previous work and propose a generalized method for stochastic analysis of publish/subscribe systems that apply identity-based hierarchical routing. The method is based on an analytical model that allows to freely set many important system parameters. In particular, arbitrary tree-based broker overlay topologies are supported and publishers as well as subscribers can connect to any broker in the system. Finally, the subscription arrival rates and their lifetimes, as well as the notification publication rates, can be specified for each broker and filter class individually to model locality. The proposed analytical model can be used to predict the routing table sizes as well as the message rate for a given workload and configuration scenario. This allows to evaluate trade-offs across multiple scenarios with minimal overhead. While presenting our method, we consider an exemplary setting that we use as a running example in order to demonstrate the applicability of our approach and obtain some quantitative results that are used to provide insight into the mathematical relationships. In addition, all analytical results presented for the exemplary setting which comprise over 1400 concrete workload and configuration scenarios, have been compared against simulation of the system to confirm the validity of our method.

The remainder of the paper is organized as follows: In Sect. 2, we describe how routing is done in the type of systems we consider. Section 3 introduces the foundational system model and the exemplary setting. Sect. 4 describes our

analysis method in detail. In Sect. 5 related work is reviewed. Finally, the paper is wrapped up in Sect. 6.

2 Publish/Subscribe Routing

Routing in publish/subscribe systems takes place on two levels. On the upper level, the *overlay network*, messages traverse the broker overlay topology based on their content and the currently active subscriptions. On the lower level, messages are routed through the *physical network* between neighboring brokers in the overlay topology. The routing of notifications in the broker overlay network is done according to a publish/subscribe routing algorithm. In this paper, we consider publish/subscribe systems that use *hierarchical routing* [5,3] to route notifications from publishers to subscribers. With this approach, brokers are arranged in a tree-based overlay topology, where one of the brokers is designated as *root broker*. Subscriptions are propagated only upwards in the broker tree from the subscriber hosting brokers towards the root broker and establish routing entries on their way forming the reverse delivery path for matching notifications.

The propagation of a subscription can be suppressed if the delivery of all notifications matching the subscription is already guaranteed by another active subscription propagated previously. With *identity-based* hierarchical routing, which we consider here, this is the case if a subscription matching the same set of notifications has already been forwarded to the parent broker before. Notifications, on the other hand, are always propagated all the way up to the root broker. In addition, notifications are propagated downwards towards subscribers whenever they meet a matching routing entry on their way to the root broker.

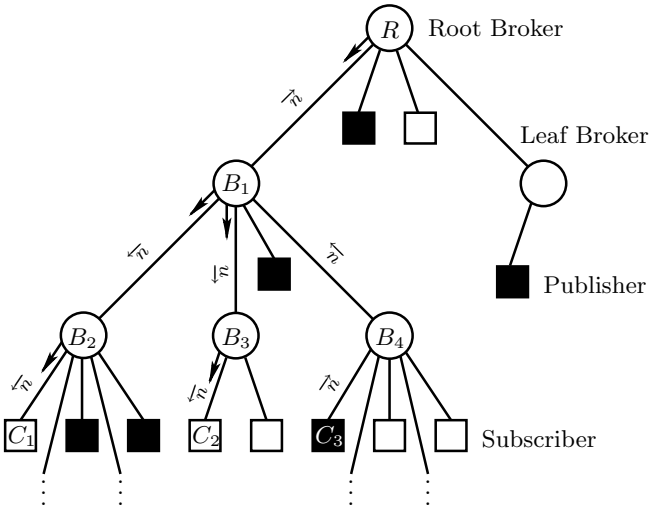

Fig. 1. Hierarchical identity-based routing

The routing algorithm is illustrated in Fig. 1. Client C_1 subscribes to a filter class and the subscription is propagated upwards installing corresponding routing entries in the routing tables of brokers B_2, B_1 and R (depicted as solid arrows pointing downwards). After that, client C_2 subscribes to the same filter class and the subscription is propagated upwards to B_1 where it meets an already installed routing entry for the respective filter class. Thus, the subscription is not forwarded further up the tree. Then, client C_3 publishes a notification n matching the filter class that the other clients have subscribed to. The notification is propagated from B_4 to its parent broker B_1 which forwards it to the root broker R. Due to the routing entries at B_1, the notification is additionally forwarded to B_2 and B_3 which deliver it to clients C_1 and C_2, respectively.

3 Foundational Model and Exemplary Setting

Before we start presenting our analysis method, we first describe the underlying system model as well as an exemplary setting for our running example.

3.1 Foundational System Model

Instead of dealing with clients directly, we assume independent arrivals of new subscriptions at the brokers for each filter class. The subscription inter-arrival times are modeled using exponential distributions. The subscription lifetimes, on the other hand, can have arbitrary distributions. We denote with $\lambda^f(B)^{-1}$ the mean *inter-arrival time* and with $\mu^f(B)^{-1}$ the mean *lifetime* of subscriptions at broker B for filter class f. We denote with $\omega^f(B)$ the *publication rate* of broker B for filter class f. Let $\mathcal{B}$ be the set of all brokers and $\mathcal{F}$ be the set of filter classes. The overall publication rate ω^f of a filter class within the whole system is then $\omega^f = \sum_{B \in \mathcal{B}} \omega^f(B)$ and the overall publication rate over all filters classes is $\omega = \sum_{f \in \mathcal{F}} \omega^f$.

To ease the presentation of our method, we assume that each notification belongs to exactly one filter class and that subscriptions are placed for individual filter classes only. However, the presented approach can be extended from the pure channel-based approach to general identity-based routing. In this case, each subscription belongs to a filter class as before but poses an additional constraint on the notification content. Essentially, this extension effects the probability that two subscriptions are identical and the set of notifications matched by a subscription. Both effects can be addressed by introducing two variable factors depending on the number of subscriptions.

3.2 Exemplary Setting

Next, we present an exemplary setting that we use as a running example in the rest of the paper in order to demonstrate the applicability of our approach and obtain some quantitative results that illustrate the steps of our analysis method and provide insight into the mathematical relationships. To this end, a scenario was chosen that illustrates trade-offs in using different broker topologies and

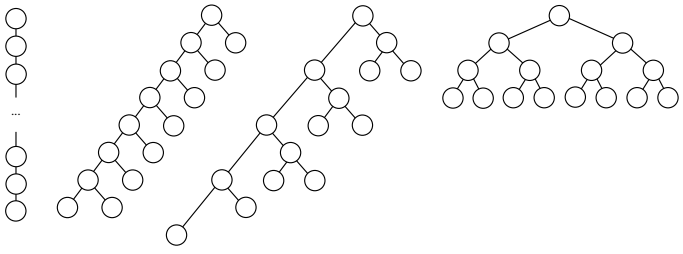

Fig. 2. Resulting topologies for 15 brokers

provides some quantitative results that can be easily interpreted by the reader. Note that all analytical results presented for the exemplary setting (covering in total over 1400 concrete workload and configuration scenarios) have been compared against results obtained through simulation of the system. In all cases, the analytical results were identical to the simulation results.

We consider 7 exemplary topologies consisting of 127 brokers arranged in a binary tree. The trees are varied from a balanced tree with seven levels to a linear arrangement of the brokers with 127 levels by restricting the maximum number of brokers at each level of the tree to $1, 2, 4, 8, 16, 32, 64$, respectively. We denote the topologies by T_1, T_2, T_4, T_8, T_{16}, T_{32} and T_{64}. Fig. 2 shows the four topologies that would result for 15 brokers.

Besides using different topologies, most experiments we present in the context of the example vary the mean overall number of subscriptions $\mathcal{N}_s$ in the system from 10 to 700 000 (equally distributed among brokers and filter classes). Please note that as mentioned in Sect. 1, our analysis method allows arbitrary distributions of the subscriptions and the notifications published among brokers and filter classes. The exemplary setting uses a uniform distribution in order to come up with simple scenarios that illustrate the results. For the rest of this paper, unless stated otherwise, the following parameters are fixed:

- $\mid \mathcal{F} \mid = 1000$, where $\mathcal{F}$ is the set of all filter classes.
- Mean lifetime of subscriptions $\mu^f(B)^{-1}$ is $60s$ for all brokers and filter classes.
- The overall publication rate ω is set to $1000s^{-1}$, i.e., 1000 notifications are published per second (equally distributed among brokers and filter classes).

4 Analysis Method

We now present our analysis method and derive the *routing table sizes* (Sect. 4.1) which are a measure for the matching overhead, as well as the *message rate* (Sect. 4.2) which can be used to estimate the overall communication costs.

4.1 Routing Table Sizes

The routing table of a broker consists of *local routing entries* used to deliver notifications to local subscribers and *remote routing entries* used to forward

notifications to child brokers along delivery paths. The remote routing entries of a broker B depend on the local and remote routing entries of its child brokers. We first introduce $p_0^f(B)$ as the probability that broker B has no *local* subscription for a filter class f. Modeling the subscriptions at the broker with a $M/G/\infty$ queuing system, the probability $p_0^f(B)$ can be determined based on the arrival rate $\lambda^f(B)$ of subscriptions at broker B and their mean lifetime $\mu^f(B)^{-1}$ [6]:

$$p_0^f(B) = e^{-\lambda^f(B)/\mu^f(B)} \tag{1}$$

The expected number of local routing entries for a filter class f is then:

$$x_l^f(B) = \lambda^f(B) \cdot \mu^f(B)^{-1} \tag{2}$$

We now introduce $P_0^f(B)$ as the probability that B has neither a local subscription nor a remote routing entry for filter class f. We denote with $\mathcal{C}(B)$ the set of all child brokers of broker B. $P_0^f(B)$ depends directly on $p_0^f(B)$ and on the value of P_0^f for all child brokers of B (if B is not a leaf broker)[1]:

$$P_0^f(B) = p_0^f(B) \cdot \prod_{C \in \mathcal{C}(B)} P_0^f(C) \tag{3}$$

Given that client communication is local, the local routing entries are of minor interest as they do not cause network traffic. We therefore concentrate on the remote routing entries. A broker B has no remote routing entry for a filter class f and a child broker C if C has neither a local subscriber nor a remote routing entry for f. The probability that this is the case is $P_0^f(C)$. The expected number of remote routing entries broker B has for filter class f can then be determined as

$$x_r^f(B) = \sum_{C \in \mathcal{C}(B)} (1 - P_0^f(C)) \tag{4}$$

Thus, the sum of all remote routing entries for filter class f is given by

$$x_r^f = \sum_{B \in \mathcal{B}} x_r^f(B) \tag{5}$$

We denote with $x_r = \sum_{f \in \mathcal{F}} x_r^f$ the expected overall number of remote routing entries in the system. When the number of subscriptions grows infinitely, x_r converges to the product of the number of overlay links and the number of filter classes, i.e., 126 000 in our example.

Figure 3 shows the expected overall number of remote routing entries x_r for the exemplary seven topologies plotted against the mean number of subscriptions in the system. As the number of subscriptions grows, x_r increases strictly monotonically with the gradient[2] continuously decreasing. As expected, x_r eventually

[1] In case of a leaf broker the (empty) product equals 1.

[2] Please note that all figures use a logarithmic scale for the number of subscriptions.

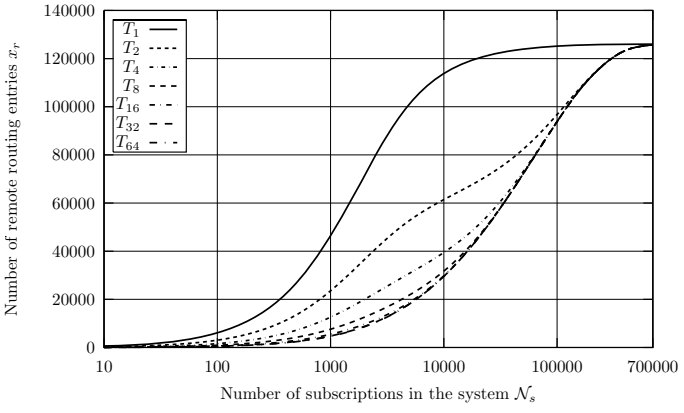

Fig. 3. Routing table sizes

converges to 126 000 for all topologies. The gradient at which the individual plot of a certain topology starts in the origin depends on the average path length of the topology. The longer the average path length, the steeper the plot is for lower number of subscriptions and the flatter it is for higher number of subscriptions.

4.2 Message Rate

The message rate is the sum of the rate used for notification messages and the rate used for control messages.

Notification Rate. In order to derive the notification rate, we first look at the case where all publishers are connected to the root broker. Consider the publication of a notification of filter class f. This notification causes one message for each remote routing entry for filter class f in the system. Thus, the number of notifications b_n sent in the system per second is:

$$b_n = \sum_{f \in \mathcal{F}} x_r^f \cdot \omega^f \tag{6}$$

If publishers can connect to arbitrary brokers, we have to add the number of additional messages sent due to hierarchical routing. A notification forwarded by broker B to its parent broker B' is *additional* if no subscriber for the respective filter class exists in the sub-tree rooted at B since the notification would not have been forwarded over this link from B' to B if it would have been published at the root broker R instead. We will use the notation $\hat{B}$ to denote the parent broker of broker B assuming that $B \neq R$. We introduce $\mathcal{P}(B)$ as the set of brokers on the path from B to R, including B and excluding R, i.e., $\mathcal{P}(B) = \{B\} \cup \mathcal{P}(\hat{B})$ if $B \neq R$ and $\mathcal{P}(B) = \emptyset$ otherwise. The mean number of additional messages caused by a notification of filter class f that is published at a broker B is $\sum_{B' \in \mathcal{P}(B)} P_0^f(B')$ since $P_0^f(B')$ is the probability that there is

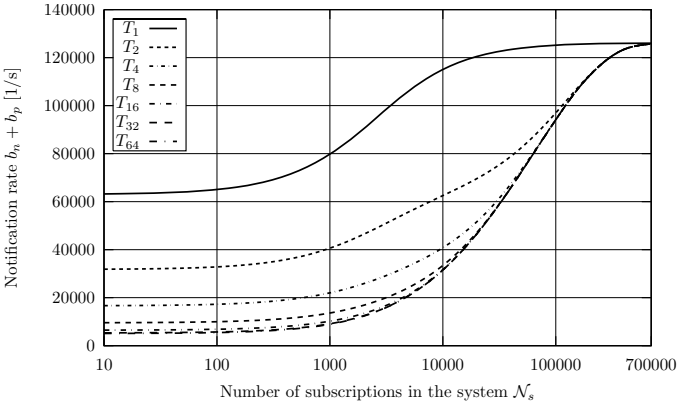

Fig. 4. Overall notification rate

no subscription for filter class f in the sub-tree rooted at B'. Thus, the mean number of additional messages published per second is:

$$b_p = \sum_{f \in \mathcal{F}} \sum_{B \in \mathcal{B} \setminus \{R\}} \left(\omega^f(B) \cdot \sum_{B' \in \mathcal{P}(B)} P_0^f(B') \right) \tag{7}$$

Figure 4 shows the overall notification rate $b_n + b_p$ for our exemplary setting. The rate monotonically increases until it converges to 126 000 notifications per second. For smaller numbers of subscriptions, the rate is dominated by b_p, while for larger numbers of subscriptions it is dominated by b_n. Additionally, the more balanced the topology is, the less important b_p is.

Control Message Rate. The control message rate b_c consists of all messages sent in the system to keep the broker routing tables up-to-date. The key to determine this rate is the toggling of brokers: We say that a broker B is in state 0 for filter class f if it has neither a local nor a remote subscription (i.e., a routing entry installed for one of its child brokers) for this filter class. Otherwise, the broker is said to be in state 1 for this filter class. Since we are using hierarchical identity-based routing (cf. Sect. 2), each time a broker toggles from state 0 to state 1 or vice versa it sends a *toggle message* to its parent broker. The control message rate is then given by the sum of the toggle rates of all brokers, where the toggle rate of an individual broker depends on its local clients and on the toggle rates of its child brokers.

To derive the toggle rate of an inner broker for a given filter class, we need to determine the rate at which subscriptions for the respective filter class arrive in the sub-tree rooted at the broker. We will refer to this rate as the *accumulated* subscription arrival rate. The latter depends on the local subscription arrival rate at the broker, given by $\lambda^f(B)$, and the accumulated arrival rates of its child brokers. The subscription arrivals at each broker form a Poisson process and the

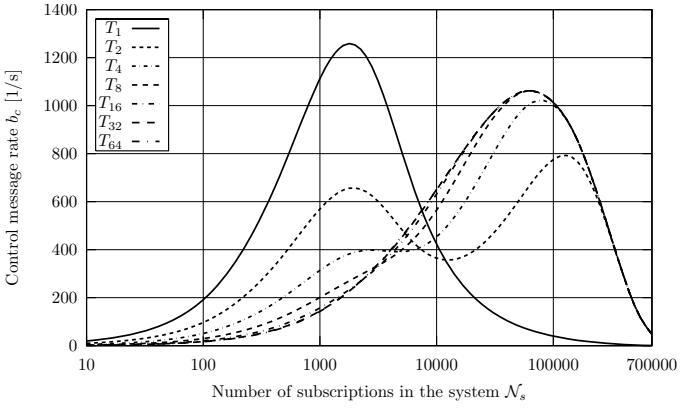

Fig. 5. Control message rate

superposition of multiple independent Poisson processes is also a Poisson process with arrival rate equal to the sum of the arrival rates of the individual Poisson processes. Thus, we obtain the following recursive formula for the accumulated subscription arrival rate $\lambda_a^f(B)$ of broker B for filter class f:

$$\lambda_a^f(B) = \lambda^f(B) + \sum_{C \in \mathcal{C}(B)} \lambda_a^f(C) \tag{8}$$

Using $\lambda_a^f(B)$, we can now calculate the toggle rate $M^f(B)$ of each broker in the system for filter class f. Each subscription issued or revoked leads to a control message sent by broker B to its parent broker if there is no other subscription for the same filter class in the sub-tree rooted in B. The probability for this is $P_0^f(B)$. Moreover, the expected values of the accumulated subscription arrival rate and the accumulated subscription death rate are equal because we consider a system in equilibrium. The expected number of toggles per second is thus:

$$M^f(B) = 2 \cdot \lambda_a^f(B) \cdot P_0^f(B) \tag{9}$$

The control message rate (total number of toggle messages) b_c is given by:

$$b_c = \sum_{f \in \mathcal{F}} \sum_{B \in \mathcal{B} \setminus R} M^f(B) \tag{10}$$

Figure 5 shows the control message rate b_c in the publish/subscribe system for the seven exemplary topologies. For T_1, T_8, T_{16}, T_{32} and T_{64}, the control message rate rises from 0 to a global maximum, then starts to drop, and finally converges to 0 as the number of subscriptions is further increased. This is because for small numbers of subscriptions, the routing tables are only lightly filled and therefore when a subscription is issued or revoked, the probability that some toggle messages are generated is high. However, as the routing tables get increasingly filled, a point is reached after which the probability of generating a

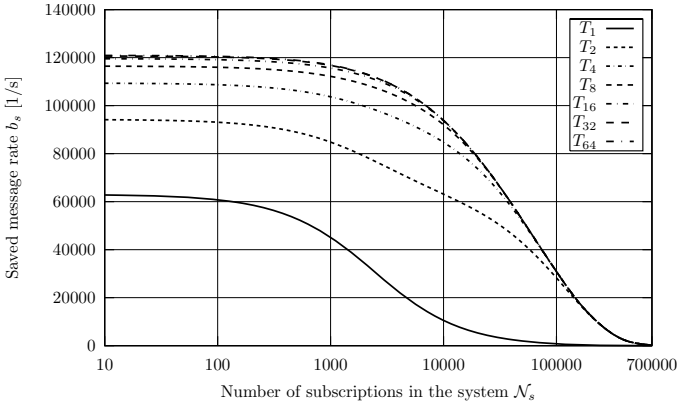

Fig. 6. Saved message rate compared to flooding

toggle message when a subscription is issued or revoked starts to drop. In consequence, the control message rate starts to decrease after reaching its maximum and eventually converges to 0.

For T_1, the control message rate reaches its maximum for a much smaller number of subscriptions than in the other topologies. This is due to the fact that in this topology the routing tables fill up more quickly. The plots of T_2 and T_4 are different in that they additionally have a local maximum caused by the asymmetry of these topologies.

Comparison with Notification Flooding and Simple Routing. The analysis method presented so far can already be used to compare hierarchical routing to basic notification flooding. With flooding no remote routing entries are used: Each published notification is sent once over every overlay link leading to a notification rate of

$$b_f = \left(\sum_{B \in \mathcal{B}} |\mathcal{C}(B)| \right) \cdot \sum_{f \in \mathcal{F}} \omega^f \tag{11}$$

The notification rate saved by hierarchical routing compared to basic notification flooding is thus given by:

$$b_s = b_f - (b_n + b_p) \tag{12}$$

The overall message rate b saved by applying hierarchical routing compared to using basic notification flooding is then:

$$b = b_s - b_c \tag{13}$$

Figure 6 shows the saved message rate for the seven exemplary topologies. The saved message rate equals 126 000 notifications per second if there are no subscriptions in the system. For larger numbers of subscriptions, the message rate

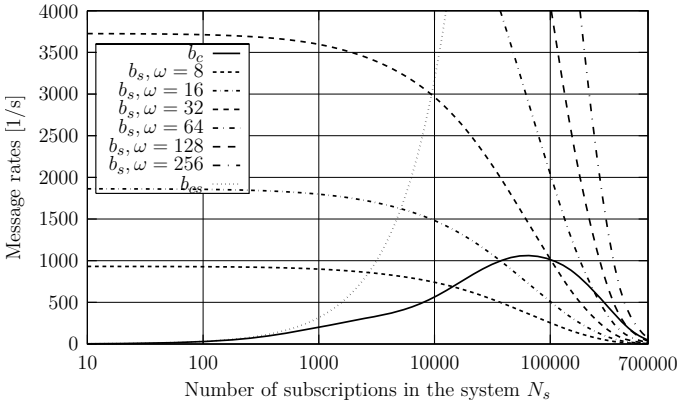

Fig. 7. Control message vs. notification rate (for T_8)

saved continuously decreases and finally converges to 0. The more unbalanced the topology is, the steeper the drop in the saved message rate is.

For our last example, we include a comparison with simple routing, where each subscription is always propagated towards the root broker [3]. In this case, the notification traffic is the same while the control traffic is:

$$b_{cs} = 2 \cdot \sum_{f \in \mathcal{F}} \sum_{B \in \mathcal{B} \setminus \{R\}} \lambda_a^f(B) \tag{14}$$

Figure 7 depicts the control message rates b_c and b_{cs} as well as the saved notification rate b_s for topology T_8 and different publication rates ω. It shows that depending on ω, filtering may perform worse than flooding if the number of subscriptions in the system exceeds a certain threshold. For example, with $\omega = 32s^{-1}$ this is the case for identity-based routing if $\mathcal{N}_s > 100\,000$ and for simple routing if $\mathcal{N}_s > 10\,000$. Identity-based routing outperforms simple routing.

5 Related Work

An analytical model of publish/subscribe systems based on subscription forwarding is presented by Castelli et al. [7]. The authors provide closed form analytical expressions for the overall network traffic required to disseminate subscriptions and propagate notifications, as well as for the message forwarding load on individual system nodes. However, the same restrictive assumptions as in [4] are made about the topology and the distribution of publishers and subscribers among brokers. Thus, this model is not applicable in most practical scenarios.

Bricconi et al. present in [8] a simple model of the JEDI publish/subscribe system. The model is mainly used to calculate the number of notifications received by each broker using a uniform distribution of subscriptions. To model the multicast communication, the authors introduce a spreading coefficient between 0

and 1 which models the probability that a broker at a given distance (in hops) from the publishing broker receives a published notification.

Baldoni et al. [9,10] propose an analytical model of distributed computation based on a publish/subscribe system. The system is abstracted through two delays (subscription/unsubscription delay and diffusion delay) which are assumed to be known. The proposed model is only used to calculate the number of notifications that are missed by subscribers due to high network delays.

A basic high-level cost model of publish/subscribe systems in mobile Grid environments is presented in [11]. This model, however, does not provide much insight into the behavior of the system since it is based on the assumption that the publish/subscribe cost and time delay per notification are known. In [12], probabilistic model checking techniques and stochastic models are used to analyze publish/subscribe systems. The communication infrastructure (i.e., the transmission channels and the publish/subscribe middleware) are modeled by means of probabilistic timed automata. The analysis considers the probability of message loss, the average time taken to complete a task and the optimal message buffer sizes. However, a centralized architecture is assumed.

In [13], a methodology for workload characterization and performance modeling of distributed event-based systems is presented. A workload model of a generic system is developed and analytical analysis techniques are used to characterize the system traffic and to estimate the mean notification delivery latency. For more accurate performance prediction queuing Petri net models are used. While this technique is applicable to a wide range of systems, it relies on monitoring data obtained from the system and it is therefore only applicable if the system is available for testing. Furthermore, for systems of realistic size and complexity, the queuing Petri net models would not be analytically tractable.

6 Conclusions

In this paper, we presented a generalized method for stochastic analysis of publish/subscribe systems employing identity-based hierarchical routing. The method eliminates the major limitations of existing work in this area and is applicable to a much wider range of systems. We presented results for over 1400 different workload and configuration scenarios for which we have compared the analytical results against simulation confirming the validity of our method.

The analysis method we presented provides an important foundation for performance modeling and evaluation of publish/subscribe systems which is an important contribution in an area where evaluations mostly rely on extensive simulation studies and often neither the simulation code nor the underlying data sets are publicly available.

As a next step, we intend to build on our analysis method and derive performance metrics such as utilization and delay for physical links and overlay links as well as notification and subscription delays. Furthermore, we intend to extend our analytical model to support covering-based and merging-based routing [3], peer-to-peer routing in addition to hierarchical routing, and advertisements.

References

1. Oki, B., Puegl, M., Siegel, A., Skeen, D.: The information bus: an architecture for extensible distributed systems. In: SOSP 1993: Proceedings of the fourteenth ACM symposium on Operating systems principles, pp. 58–68. ACM, New York (1993)
2. Eugster, P., Felber, P., Guerraoui, R., Kermarrec, A.M.: The many faces of publish/subscribe. ACM Computing Surveys 35(2), 114–131 (2003)
3. Mühl, G., Fiege, L., Pietzuch, P.: Distributed Event-Based Systems. Springer, Heidelberg (2006)
4. Jaeger, M.A., Mühl, G.: Stochastic analysis and comparison of self-stabilizing routing algorithms for publish/subscribe systems. In: Proc. of the 13th IEEE/ACM Intl. Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp. 471–479 (2005)
5. Cugola, G., Di Nitto, E., Fuggetta, A.: The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. IEEE Transactions on Software Engineering 27(9), 827–850 (2001)
6. Jain, R.: The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. Wiley Interscience, New York (1991)
7. Castelli, S., Costa, P., Picco, G.P.: Modeling the communication costs of content-based routing: The case of subscription forwarding. In: Proc. of the Inaugural Conference on Distributed Event-Based Systems (DEBS 2007), pp. 38–49 (2007)
8. Bricconi, G., Nitto, E.D., Tracanella, E.: Issues in analyzing the behavior of event dispatching systems. In: Proc. of 10th Intl. Workshop on Software Specification and Design, pp. 95–103 (2000)
9. Baldoni, R., Beraldi, R., Piergiovanni, S.T., Virgillito, A.: Measuring notification loss in publish/subscribe communication systems. In: Proc. of 10th IEEE Pacific Rim International Symposium on Dependable Computing, pp. 84–93 (2004)
10. Baldoni, R., Beraldi, R., Piergiovanni, S.T., Virgillito, A.: On the modelling of publish/subscribe communication systems. Concur. and Comput.: Pract. and Exper. 17(12), 1471–1495 (2005)
11. Oh, S., Pallickara, S.L., Ko, S., Kim, J.-H., Fox, G.C.: Cost model and adaptive scheme for publish/Subscribe systems on mobile grid environments. In: Sunderam, V.S., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2005. LNCS, vol. 3516, pp. 275–278. Springer, Heidelberg (2005)
12. He, F., Baresi, L., Ghezzi, C., Spoletini, P.: Formal analysis of publish-subscribe systems by probabilistic timed automata. In: Derrick, J., Vain, J. (eds.) FORTE 2007. LNCS, vol. 4574, pp. 247–262. Springer, Heidelberg (2007)
13. Kounev, S., Sachs, K., Bacon, J., Buchmann, A.: A methodology for performance modeling of distributed event-based systems. In: Proc. of the 11th IEEE Intl. Symposium on Object/Component/Service-oriented Real-time Distributed Computing (2008)