Modeling the Communication Costs of Content-based Routing: The Case of Subscription Forwarding

Stefano Castelli
Dept. of Information and Communication Technology
University of Trento, Italy
castelli.stefano@gmail.com

Paolo Costa
Dept. of Computer Science
Vrije Universiteit
Amsterdam, The Netherlands
costa@cs.vu.nl

Gian Pietro Picco
Dept. of Information and Communication Technology
University of Trento, Italy
picco@dit.unitn.it

ABSTRACT

Content-based routing (CBR) provides the core distribution support of several middleware paradigms, most notably content-based publish-subscribe. Despite its popularity, however, the performance of CBR protocols is typically evaluated through simulation, and analytical models are extremely rare in the literature. Analytical models capture formally the characteristic of the analyzed system, and are therefore worth pursuing on their own. However, they also provide very practical advantages in that they allow one to evaluate tradeoffs extensively (i.e., across many parameter combinations and across all the interesting values) without the lengthy computation times required by simulations. These benefits are particularly welcome when large-scale networks are considered.

In this paper, we provide an analytical model for *subscription forwarding* [4], arguably the most common CBR protocol in use today and one that is often used as a baseline against which to compare new approaches. We provide closed analytical expressions for the overall network traffic required to disseminate subscriptions and propagate notifications, as well as for the message forwarding load on individual nodes. The analytical model we present is validated through simulation for networks with more than 100,000 nodes and against several combinations of the relevant parameters. Results show that our model remains within 3% of the simulated traffic (and in most scenarios well below 1%), therefore indicating that our model can effectively replace simulations. The paper is completed by some examples of how our analytical model can be used in practice, including a precise characterization of the tradeoffs between subscription forwarding and event forwarding.

Categories and Subject Descriptors

C.2.4 [**Computer Communication Networks**]: Distributed Systems

General Terms

Algorithms

DEBS '07, June 20-22, 2007 Toronto, Ontario, Canada

Keywords

Subscription Forwarding, Analytical model

1. INTRODUCTION

Content-based routing is becoming an increasingly popular building block for distributed applications, and is notably at the core of content-based publish-subscribe systems. Content-based routing differs from classical routing paradigms as messages are routed based on their content rather than their destination address, and therefore is usually implemented using dedicated protocols.

Content-based routing protocols. Most of existing CBR protocols assume the existence of a tree-shaped overlay network interconnecting the *brokers*, the application-level routers in charge of disseminating the subscriptions containing the event-filtering patterns and of routing events. The two classic CBR strategies are:

- *Event forwarding:* All events are flooded throughout the tree, to all the brokers. No in-network event filtering is performed: events are matched against subscriptions only at the brokers.
- *Subscription forwarding:* Subscriptions are flooded to the whole tree, and in doing so they setup routes towards subscribers from each node. A published event matching a subscription is routed along the reverse path towards subscribers, using the aforementioned routes. Subscriptions for the same filter need not propagate to the whole tree, as they can often leverage off the routes already setup by the previous subscriptions. This is arguably the most common CBR strategy today, and is typically used in the literature as a baseline for comparison.

A number of variations over these schemes exist [10]. For instance, some systems (e.g., [7, 11]) use a form of *hierarchical forwarding* where the root works as a sort of meeting point for subscriptions and events. Advertisements have been proposed [4] to reduce the subscription traffic, by sending subscriptions only to those nodes that expressed the intent to publish a given class of events. Also, approaches that do not rely on a tree overlay exist (e.g., [5, 6, 8]).

Motivation. Despite the increasing number of proposals, the predominant means of analyzing, comparing, and validating CBR protocols is simulation. There is almost *no* study that, in contrast, provides an analytical characterization of a CBR protocol. The tradeoffs between simulation and analytical models have been stated in the context of CBR in a recent paper [2] by Carzaniga and Hall:

> Simulation analyses are difficult to read because of the many configuration parameters that define each exper-

iment. Even if one understands the performance trend observed through simulation in a section of the parameter space, it is by no means clear whether and how such trends can tell anything about other parts of the parameter space. Also, simulations may not capture essential, implementation-independent properties of a protocol. Having a formal model of networks and protocols would allow us to formulate more general statements about the complexity of protocols.

For instance, the general wisdom about the two aforementioned classic strategies is that event forwarding is convenient when the number of subscriptions is much greater than the number of events, while subscription forwarding is preferable when the reverse holds. However, what does it mean "much greater"? Where is the break-even point between the two approaches? How does it change according to different combination of parameters, such as network size and shape as well as application profile (e.g., percentage of subscribers and receivers, message load)? *Precise* and *general* answers to these questions are difficult to derive through simulation, and are best obtained through analytical models.

Contribution. In this paper, we make a step towards the goal of providing analytical studies of CBR protocols, by providing a characterization of the subscription forwarding strategy. We chose this protocol because it is well-known, is deployed in several systems, and is often used as a baseline for comparison against newer approaches. Therefore, although more efficient protocols have been proposed, its analytical characterization is likely to have a stronger impact on common practice, fostering a deeper understanding of the properties of this basic strategy.

The focus of our analysis is on communication costs. Our model provides closed analytical expressions for the overall network traffic required to disseminate subscriptions and propagate notifications, as well as for the message forwarding load experienced by a single node. Both are concerned with *scalability*: the first quantity characterizes it from the point of the view of the traffic globally generated within the network, while the second is helpful in highlighting the potential of bottlenecks on some nodes.

Clearly, a model is useful only if it accurately represent reality. Given that our concern is scalability, a validation based on real-world experiments is difficult. For this reason, we chose to validate our model using the PEERSIM [1] simulator, a popular choice to test application-level protocols for large-scale scenarios. This choice allowed us to validate our model in networks with more than 100,000 nodes and with several combinations of the relevant parameters. Results show that our model remains within 3% of the simulated traffic, and in most scenarios well below 1%. These results not only support the validity of our model, but also indicate that it can effectively replace simulation as the main investigation tool without loosing accuracy, therefore enabling a more extensive analysis at a lower costs in terms of computation time.

Finally, we show how our model can be useful in practice by analyzing three examples. First, we provide a precise characterization of the tradeoffs in terms of message traffic between the two classic CBR protocols, subscription forwarding and event forwarding. Second, we show how our model of the message forwarding load can be used to identify bottlenecks in the deployment of systems based on the subscription forwarding protocol. Finally, we show how the tradeoffs between traffic and forwarding load are affected by the shape of the tree topology. In all these cases, we show how the informal rules of thumb determining the tradeoffs can be stated precisely, given some key parameters of the target scenario.

Related work. As we mentioned, CBR protocols are almost always evaluated using simulation. Therefore, there is almost no directly[1] related work to compare against. A notable exception is the work described in [9], where hierarchical forwarding and event forwarding (respectively, identity-based routing and flooding in [9]) are compared. However, such work has two significant limitations:

1. subscribers are assumed to be placed always on leaves, while publishers are assumed to be either on leaves or on the root;
2. the model is not validated in any way, therefore its accuracy w.r.t. simulated or real scenarios remains unknown.

In comparison, the only assumption we make in this paper about the placement of publishers or subscribers is that they are uniformly spread in the system, an assumption typically made also by the vast majority of simulation studies. Moreover, we do validate our model through simulation, therefore giving a quantitative measure of the difference (very small, as mentioned above) between the two evaluation methods. An additional difference between our work and [9] is in the routing protocols considered, which makes the two works not directly comparable quantitatively. Nevertheless, we believe that the two routing strategies are similar enough that the techniques we used in this paper for modeling subscription forwarding can be adapted towards hierarchical forwarding: such adaptation is in our immediate research agenda. Similarly, our model can also be adapted to model CBR protocols that use advertisements, since these are effectively propagated much like subscriptions in subscription forwarding.

The only other directly related work we were able to find is an unpublished technical report by Carzaniga et al. [3]. The authors define a theoretical framework for CBR schemes and use it to derive memory requirements for the matching process, along with general correctness properties of the protocols. As such, their focus is rather different from ours, in that we aim at deriving expressions for characterizing quantitatively subscription forwarding from a communication standpoint, and we are not interested in proving the correctness of the protocol itself. Moreover, as in [9], the authors of [3] do not provide any validation of their model.

Roadmap. Section 2 defines the characteristics of the systems we consider, and the assumptions we make in deriving our model. Section 3 presents our analytical model of the overall message traffic and of the node forwarding load generated by subscription forwarding. Expressions for the much simpler event forwarding protocol are also derived along the way. Section 4 reports about an extensive validation of our model through simulation. Section 5 illustrates the versatility and practical relevance of our model through examples. Finally, Section 6 ends the paper with brief concluding remarks.

2. SYSTEM MODEL AND ASSUMPTIONS

Our analytical model of subscription forwarding assumes an overlay network containing n broker nodes organized in a unrooted tree, hereafter called *broker tree*. We do not consider the clients attached to brokers, since they do not influence the routing cost. Clients, however, determine the role of a broker in the broker tree. We say that a broker is a *subscriber* for pattern p (or a *publisher* of event e) if at least one of its client is. Similarly, a broker is a *receiver* for event e if one of its clients is a subscriber for a pattern matching e. We assume that subscribers, publishers, and receivers are uniformly spread throughout the broker tree.

Although we assume that, for what concerns routing, the broker tree is unrooted, for our modeling it is useful to assume that the

[1]Many analytical models of routing are available in the networking community. However, these models are not directly comparable, let apart reusable, due to the peculiarities of content-based addressing.

broker tree has a height of h levels, with the root at level $l = 0$. Moreover, we assume that the tree is *full*, i.e., all the leaves are at level $l = h$ *and* all internal nodes have exactly f children. The total number of nodes is then easily computed as

$$n = \sum_{l=0}^{h} f^l$$

This assumption, however, is mostly for illustration purposes. In practice, arbitrary tree topologies are encompassed in our model. Assuming the average degree $\bar{f}$ is known, all the formulae of the analytical model of the overall message traffic described in Section 3.1 still hold when replacing f with $\bar{f}$. On the other hand, the model of the forwarding load on a node, presented in Section 3.2, is more sensitive to variations in the topology. In this case, although replacing f with $\bar{f}$ is still a reasonable option in many cases, some additional parameters of the topology are required to enable an accurate estimation. Section 4 reports about the validation of our model by considering both full and arbitrary trees.

Our model is based on a few parameters that characterize the application profile. We define $\sigma(p)$ as the probability of a node to be subscribed to a given pattern p. In the case of a uniform distribution for patterns, (i.e., each pattern p occurs with the same probability), $\sigma(p) = k_p$, where $0 \leq k_p \leq 1$ is some constant. In general, however, different patterns have a different probability. For instance, in the case of the commonly used Zipf distribution:

$$\sigma(p) = P\frac{c}{(p^\alpha)}$$

being P the average number of subscribed patterns per node, α the exponent characterizing the Zipf distribution, and c a parameter such that $\sum_{p=1}^{F_p} \frac{c}{p^\alpha} = 1$. Moreover, in this paper we do not consider optimizations involving covering relationships among subscriptions [10]. These can be captured by an additional parameter $\gamma(p)$, the probability that a pattern p is already covered by another, and by reusing the techniques we describe in this paper. However, this would make the treatment significantly longer and more cumbersome, and it is therefore omitted here.

In a content-based system, an event may match multiple patterns. We define $\mu(e)$ as the probability of a node to be a receiver for event e. Clearly, a receiver is also a subscriber for at least one of the patterns matching e. As with patterns, the probability $\mu(e)$ varies according to the event e, unless a uniform distribution of events $\mu(e) = k_e$, $0 \leq k_e \leq 1$ is assumed. Note that when both events and patterns follow a uniform distribution, the constraint $k_p \leq k_e$ holds. Otherwise, given a single pattern p and a matching event e, we would allow the nonsensical case where the receivers for e are less than the subscribers for p. Hereafter, to improve readability we drop the indexes p and e, wherever not ambiguous.

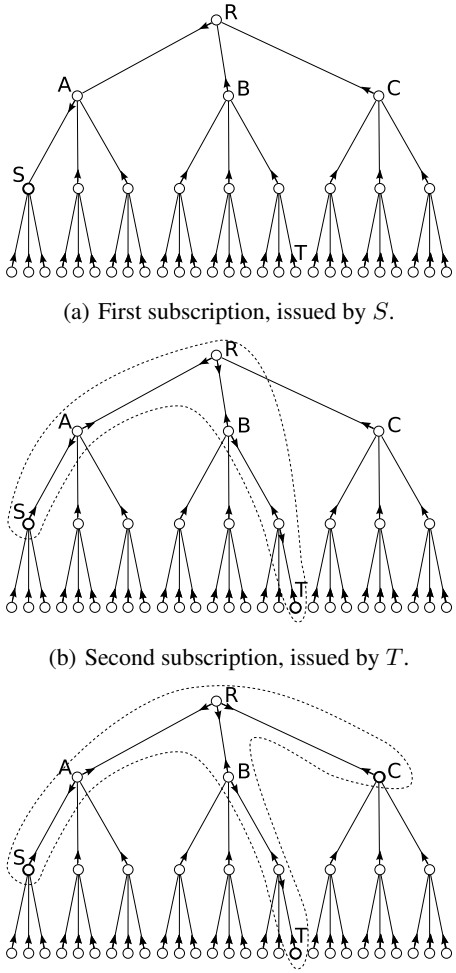
Finally, message traffic and forwarding load are affected by two parameters, again characterizing the application profile: the frequency F_p of distinct patterns and the frequency F_e of distinct events, both relative to a given observation time interval.

3. ANALYTICAL MODEL

We characterize analytically subscription forwarding in terms of the overall message overhead generated by subscription and event messages, as well as of the forwarding load experienced by a node.

3.1 Overall Message Traffic

Here we derive closed expressions for the overall message traffic generated by subscription forwarding. In comparison, the characterization of event forwarding is trivial. Indeed, since subscriptions



(a) First subscription, issued by S.

(b) Second subscription, issued by T.

(c) Third subscription, issued by C.

Figure 1: Configuration after the first, second, and third subscription for a pattern p have been propagated. Arrows represent routing entries in the subscription tables. The dashed line shows the *pattern tree*.

are never propagated, the only contribution comes from events, which are forwarded to all links. Therefore, the traffic generated by event forwarding can be easily computed as:

$$T^{EF} = F_e(n-1) \tag{1}$$

3.1.1 Cost of Routing a Single Pattern p

We begin by modeling the traffic generated by the dissemination of a generic pattern p.

If no previous subscription for a pattern p already exists in the system, subscription forwarding propagates the subscription to all the nodes. After propagation of this first subscription for p completes, a new directed tree rooted at the subscriber has been created on top of the broker tree, as depicted in Figure 1(a). An event matching the pattern p will be propagated from broker i to broker j if and only if a route from i to j exists in the broker tree.

When a second subscription for p is issued, it is not resent along links where it was already propagated by the first subscriber, and therefore is effectively routed only towards it. As can be observed

in Figure 1(b), the links belonging to the path traversed by the second subscription become "bidirectional", i.e. both the link endpoints have an entry in their routing tables. The set of all these bidirectional links form a spanning tree that connects all the nodes subscribed to p, called the *pattern tree* for p. When new subscribers appear, additional routes must be similarly created to connect the subscribers with the pattern tree. For instance, in Figure 1(c) a new subscription from C requires one additional route between this node and the first one encountered on the pattern tree, the root R in the example. The other routes in the tree are already correctly forwarding events towards C.

Therefore, the traffic $T(p)$ due to subscriptions for p is equal to the number of subscription routes (represented by arrows in figure) present on the tree. Since each link can be part of at most two routes, one towards the root and the other towards the leaves, we denote with $\rho_l(p)$ and $\lambda_l(p)$ the probability of a link on level l to be part of a route towards the root and towards the leaves, respectively. Since a generic level l contains f^l links, on average there are $(\rho_l(p) + \lambda_l(p))f^l$ routes per level. We can therefore obtain the analytic expression for $T(p)$ by summing the number of upstream and downstream routes on each level:

$$T(p) = \sum_{l=1}^{h} (\rho_l(p) + \lambda_l(p))\, f^l \tag{2}$$

We can derive the value of $\rho_l(p)$ and $\lambda_l(p)$ by observing that a generic link on the tree contains a subscription route between a father u and its child v if and only if the set $\mathcal{T}_l$ of nodes belonging to the sub-tree rooted at v contains *at least* one subscriber. Similarly, a subscription route from the child v to the father u exists if and only if the the set $\mathcal{N} \setminus \mathcal{T}_l$ of nodes *not* belonging to the sub-tree rooted at v contains *at least* one subscriber, being $\mathcal{N}$ the set of all nodes in the network. For instance, in Figure 1(c), the link between R and B is bidirectional because there is a subscriber (i.e., T) in the sub-tree rooted at B and a subscriber (i.e., S) outside that sub-tree.

Therefore, $\rho_l(p)$ is simply the probability that a subscriber exists among the $|\mathcal{T}_l|$ nodes contained in the sub-tree rooted at a node at level l. The value of $|\mathcal{T}_l|$ is easily computed as

$$|\mathcal{T}_l| = \sum_{k=0}^{h-l} f^k \tag{3}$$

based on the topological properties of the broker tree. The probability to find a subscriber in $\mathcal{T}_l$ can be computed by recalling, from Section 2, that $\sigma(p)$ is the probability of a node to be subscribed to pattern p. Therefore, we can compute $\bar{\sigma}^{|\mathcal{T}_l|}$ as the probability that *no subscriber exists* in $\mathcal{T}_l$, using the complementary probability of σ, $\bar{\sigma} = 1 - \sigma$. The probability $\rho_l(p)$ that *at least one subscriber exists* in $\mathcal{T}_l$ is then the complementary probability of $\bar{\sigma}^{|\mathcal{T}_l|}$:

$$\rho_l(p) = 1 - \bar{\sigma}^{|\mathcal{T}_l|} = 1 - \bar{\sigma}^{\sum_{k=0}^{h-l} f^k} \tag{4}$$

Along the same lines, we can derive the expression for $\lambda_l(p)$ as the probability that at least one subscriber exists among the $n - |\mathcal{T}_l|$ nodes outside $\mathcal{T}_l$:

$$\lambda_l(p) = 1 - \bar{\sigma}^{n-|\mathcal{T}_l|} = 1 - \bar{\sigma}^{n-\sum_{k=0}^{h-l} f^k} \tag{5}$$

We can now compute the total cost of disseminating *all* subscriptions for a pattern p. Recalling Equations (2)-(5), the complete formula for $T(p)$ is:

$$\begin{aligned} T(p) &= \sum_{l=1}^{h} (\rho_l(p) + \lambda_l(p))\, f^l \\ &= \sum_{l=1}^{h} \left(2 - \bar{\sigma}^{\sum_{k=0}^{h-l} f^k} - \bar{\sigma}^{n-\sum_{k=0}^{h-l} f^k}\right) f^l \end{aligned} \tag{6}$$

Equation (6) is easily verified for $\sigma(p) = 1$, i.e., all nodes are subscribed to p. In this case, $\bar{\sigma} = 0$, and therefore

$$T(p) = \sum_{l=1}^{h} 2f^l = 2(n-1)$$

i.e., all links are bidirectional and have two subscriptions.

3.1.2 Cost of Routing a Single Event e

We observe that a link between a node u (father) and a node v (child) is traversed by an event e if and only if the sub-tree $\mathcal{T}_l$ rooted at v contains at least one receiver for e and the publisher of e is not in $\mathcal{T}_l$ or, vice versa, e has been published in $\mathcal{T}_l$ and there exists at least one receiver outside $\mathcal{T}_l$.

We define the probability $\pi_l = \frac{|\mathcal{T}_l|}{n}$ as the probability that the publisher lies in $\mathcal{T}_l$, and $\bar{\pi}_l = 1 - \pi_l$ as the probability that the publisher lies outside $\mathcal{T}_l$. Therefore, the probability $\psi_l(e)$ that a link at level l is traversed by an event e is

$$\psi_l(e) = \pi_l(1 - \bar{\mu}^{n-|\mathcal{T}_l|}) + \bar{\pi}_l(1 - \bar{\mu}^{|\mathcal{T}_l|}) \tag{7}$$

being μ the probability of a node to be a receiver for e, as we defined in Section 2.

The traffic $T(e)$ due to event e can then be expressed as

$$\begin{aligned} T(e) &= \sum_{l=1}^{h} \psi_l(e) f^l \\ &= \sum_{l=1}^{h} \left(\pi_l(1 - \bar{\mu}^{n-|\mathcal{T}_l|}) + \bar{\pi}_l(1 - \bar{\mu}^{|\mathcal{T}_l|})\right) f^l \end{aligned} \tag{8}$$

Equation (8) confirms the intuition for $\mu = 1$, i.e. when all nodes are supposed to receive event e. In this case the event is forwarded on all links and, therefore, $T(e) = n - 1$.

3.1.3 Total Message Traffic

We can estimate the total message traffic generated in a given interval by recalling, from Section 2, the frequencies F_p and F_e for distinct patterns and events.

If patterns and events follow a uniform distribution, the overall message traffic for subscription forwarding is simply

$$T^{SF} = F_p T(p) + F_e T(e) \tag{9}$$

However, in general σ and μ may change for each pattern and event. The general expression of the overall message traffic is then:

$$T^{SF} = \sum_{p=1}^{F_p} T(p) + \sum_{e=1}^{F_e} T(e) \tag{10}$$

3.2 Forwarding Load on a Node

Thus far, we restricted our analysis to the network overhead introduced by the subscription and event messages exchanged in the network. This quantity accounts for scalability in terms of the *overall* load imposed on the system at large. Here, instead, we turn our attention to the load experienced by a node, in terms of the messages it must forward. This analysis complements the previous one,

in that it still provides insights about scalability, but from the point of view of an individual node.

Once more, in our analysis we consider only subscription forwarding. Indeed, event forwarding can be easily characterized under the assumptions stated in Section 2 and in particular the uniform placement of publishers and subscribers. Assuming f is the *actual* degree of a node, to account for arbitrary non-full trees, the load experienced by a generic internal node i for a given event e depends on whether the node is the publisher of e or not. In the first case, the event is propagated to all $f+1$ neighbors, otherwise only to the f neighbors different from the one that forwarded the event. The load for a generic internal node is then:

$$L_i^{EF} = \frac{1}{n}(f+1)F_e + (1-\frac{1}{n})fF_e = (\frac{1}{n}+f)F_e$$

where $\frac{1}{n}$ is the probability of a node to be a publisher. A similar expression holds for the root, which has however only f neighbors:

$$L_r^{EF} = \frac{1}{n}fF_e + (1-\frac{1}{n})(f-1)F_e = (\frac{1}{n}+f-1)F_e$$

Finally, a leaf forwards an event only when publishing it:

$$L_l^{EF} = F_e\frac{1}{n}$$

In the following, we similarly discuss separately the load experienced by the root, by an internal node, and by a leaf.

3.2.1 Subscriptions

Root. The forwarding load on the root can be estimated by observing that the first subscription received by this node is propagated to all of its children, except the one from which the subscription has been received (Figure 2(a)). No other subscription will come from this link, since a subscription is never propagated twice on the same link. Conversely, a subscription coming from any other link is propagated to the sub-tree containing the first subscriber (Figure 2(b)). This, however, happens only the first time such a subscription is received. Any additional subscription, regardless from its provenience, does not require any forwarding since all the routes have been already built, i.e., all the outgoing links have been traversed by a subscription (Figure 2(c)).

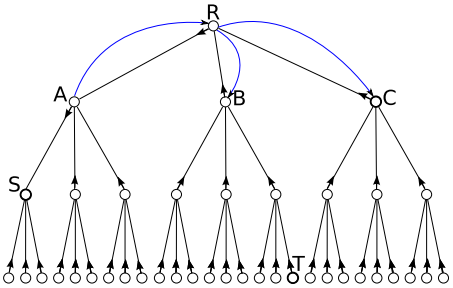
Based on these observations, the expected forwarding load on the root due to the first subscription can be expressed as the product between the probability that at least one subscriber exists in the rest of the network (i.e., $1-\bar{\sigma}^{n-1}$) times the $f-1$ children the subscription must be forwarded to. This number is derived by recalling that we exclude the root child from which the subscription is received. Let us call this node (A in Figure 2(a)) the *first subscribed child*. The contribution to the load given by the first subscription is:

$$L_{r,1}(p) = (f-1)(1-\bar{\sigma}^{n-1})$$

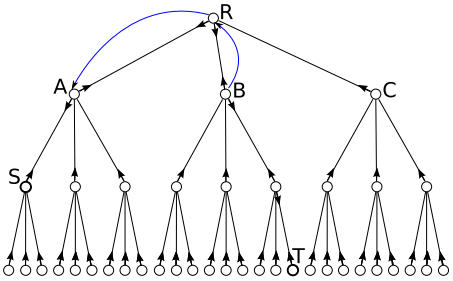
As for subsequent subscriptions, the only one that matters is the first coming either from one of the root's clients or through a link different from the one towards the first subscribed child, e.g., the one coming through B instead of A in Figure 2(b). This subscription is propagated only towards the first subscribed child. Therefore, the forwarding load is simply equal to the probability to find at least one subscriber among the $n-\frac{n-1}{f}$ nodes which do not belong to the sub-tree of the first subscribed child. The load due to subsequent subscriptions forwarded by the root is:

$$L_{r,*}(p) = (f-1)(1-\bar{\sigma}^{\frac{n(f-1)+1}{f}})$$

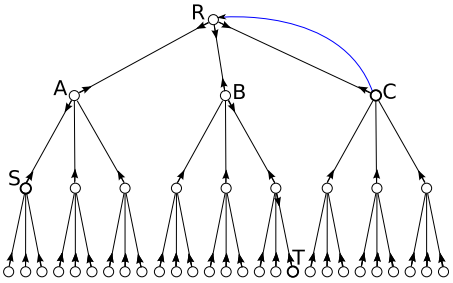
In addition, we must consider that the first subscriber might be the root itself. In this case, the subscription is forwarded to all the f children and no subsequent subscription is propagated.



(a) First subscription, from S. The root R receives the subscription through its child A, and forwards it to all the other children.



(b) Subscription from T. The root R receives the subscription through its child B. The subscription must be forwarded only towards A.



(c) Subscription from C. All routes are already in place: no forwarding by the root is required.

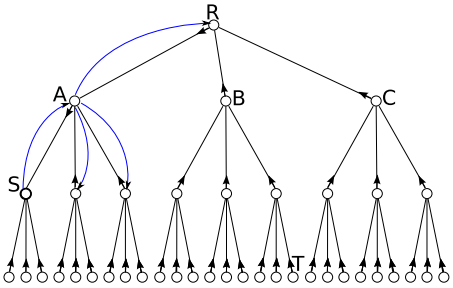
Figure 2: Subscription messages forwarded by the root R. Curvy arrows denote message propagation (only involving the root), straight arrows entries in the routing tables.

The final expression of the forwarding load on the root is:

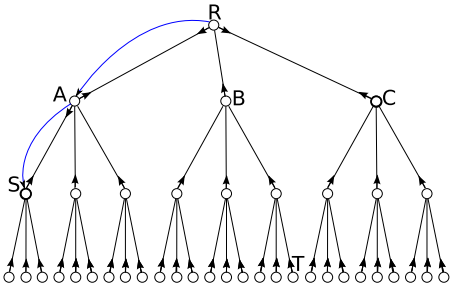
$$L_r(p) = \frac{1}{n}f + \left(1-\frac{1}{n}\right)\left[(f-1)(1-\bar{\sigma}^{n-1}) + (1-\bar{\sigma}^{\frac{n(f-1)+1}{f}})\right] \quad (11)$$

where $\frac{1}{n}$ is the probability that the root is the first subscriber for p.

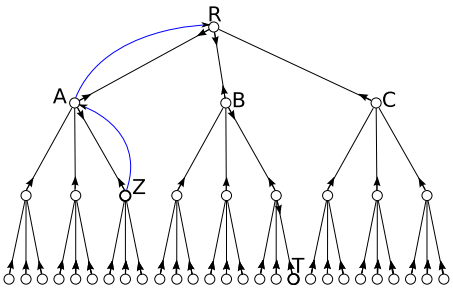
Internal Nodes. The expression for a generic internal node at level l is slightly more complex. Again, we illustrate our technique with the help of a reference example in Figure 3, where we focus on the internal node A. As with the root, the first subscription ever received must be propagated to all neighbors, as shown in Figure 3(a). The contribution of the first subscription is therefore similar to the one of the root, with the only difference that internal nodes have



(a) The root child A receives the first subscription from S and forwards it to its other f neighbors, including the root.



(b) Another subscription is issued by C. A forwards it towards the first subscriber, S. In this case, the first subscription received by A came (in this case directly) through one of the links connecting A to a child.



(c) Here, we assume instead that the first subscription in the system was issued by T, instead of S as in Figure 3(a). In this case, the first subscription received by A came through the link connecting it to the root R. A subsequent subscription from Z is propagated towards T.

Figure 3: Subscription messages dispatched by an internal node, A. Curvy arrows denote message propagation (only involving A), straight arrows entries in the routing tables.

one more neighbor:

$$L_{i,1}(p,l) = f(1-\bar{\sigma}^{n-1})$$

As for subsequent subscriptions, the observation enabling our estimate is similar to the one we made for the root: a subscription is never forwarded again along a link where it already went through. Let us call s the link belonging to the internal node (A in our case), along which the first subscription came, and $\mathcal{T}$ the set of nodes belonging to the sub-tree connected to A through s. Therefore, the first subscriber belongs to $\mathcal{T}$, and if another appears in $\mathcal{T}$ its subscription would never reach A. The contribution to the load can then be computed simply as the probability to find one subscriber outside $\mathcal{T}$, $1-\bar{\sigma}^{n-|\mathcal{T}|}$.

We distinguish two cases, $|\mathcal{T}_c|$ and $|\mathcal{T}_f|$, depending on whether s is "below" A (i.e., connecting A to one of its children, S in our example) or "above" A (i.e., connecting A to its father). In addition, similar to the root case, we take into account the forwarding along $f+1$ links occurring if the selected node is the first subscriber. The final formula for the load on a generic node at level l is then:

$$\begin{aligned} L_i(p,l) &= \frac{1}{n}(f+1) + \left(1-\frac{1}{n}\right)\Big[f(1-\bar{\sigma}^{n-1}) + \\ &\quad + \tau_c(1-\bar{\sigma}^{n-|\mathcal{T}_c|}) + \tau_f(1-\bar{\sigma}^{n-|\mathcal{T}_f|})\Big] \end{aligned} \quad (12)$$

where the coefficients

$$\tau_f = \frac{|\mathcal{T}_f|}{n-1} \qquad \tau_c = 1-\tau_f \quad (13)$$

account for the probability that s, the link along which the first subscription came, connects the node to its father or child, respectively. The term $\frac{1}{n}$ is instead the probability that the node is the first subscriber for p.

Equation (12) can be applied to arbitrary trees, provided that the values of $|\mathcal{T}_c|$, $|\mathcal{T}_f|$, and f are known for the node at hand. In the case a full tree, the expression can be simplified further by observing that the following holds:

$$|\mathcal{T}_c| = \sum_{i=0}^{h-l-1} f^i \qquad |\mathcal{T}_f| = n - \sum_{i=0}^{h-l} f^i \quad (14)$$

The formula above can be understood by looking at Figure 3(b), where $|\mathcal{T}_c|$ is the number of nodes in the sub-tree rooted at the first subscribed child S, and Figure 3(c), where $|\mathcal{T}_f|$ is the number of nodes that does not belong to A's sub-tree, including A's itself. In the case of a full tree, Equation (12) is then simplified into:

$$\begin{aligned} L_i(p,l) &= \frac{1}{n}(f+1) + \left(1-\frac{1}{n}\right)\Big[f(1-\bar{\sigma}^{n-1}) + \\ &\quad + \tau_c(1-\bar{\sigma}^{n-\sum_{i=0}^{h-l-1} f^i}) + \tau_f(1-\bar{\sigma}^{\sum_{i=0}^{h-l} f^i})\Big] \end{aligned} \quad (15)$$

Leaves. Leaves have no children and only one neighbor. Therefore, they forward only their own subscriptions:

$$L_l(p) = \sigma$$

3.2.2 *Events*

Root. When an event is published at the root, it is forwarded to a child u only if there is at least one receiver in u's sub-tree. Therefore, the number of messages forwarded by the root in this case can be expressed as the sum of the probability, over its f children, that at least one receiver exists in their sub-tree, $1-\bar{\mu}^{\frac{n-1}{f}}$. Similarly, if the event is received from one of the root's children, it is forwarded to the other $f-1$ children with the same probability above. Recalling that the probability of a node to be the publisher is $\frac{1}{n}$:

$$\begin{aligned} L_r(e) &= \frac{f}{n}(1-\bar{\mu}^{\frac{n-1}{f}}) + \left(1-\frac{1}{n}\right)(f-1)(1-\bar{\mu}^{\frac{n-1}{f}}) \\ &= (\frac{1}{n}+f-1)(1-\bar{\mu}^{\frac{n-1}{f}}) \end{aligned}$$

Internal Nodes. To compute the event load for internal nodes, we follow an approach similar to the one used for subscriptions. However, while earlier we computed the probability that one subscriber exists *outside* $\mathcal{T}$ (i.e., $1-\bar{\sigma}^{n-|\mathcal{T}|}$), here we are interested in the

probability that one receiver exists *inside* $\mathcal{T}$ (i.e., $1-\bar{\mu}^{|\mathcal{T}|}$). Indeed, subscriptions are propagated *from* subscribers whereas events are forwarded *towards* receivers.

Similarly to subscriptions, we distinguish between two cases. If the event is coming from the father, the forwarding load is equal to the number of children f, times the probability that at least one receiver exists in $\mathcal{T}_c$. If, instead, the event comes from one of the children, it is forwarded towards *i)* the other $f-1$ children (or f if the node is itself the publisher), if one receiver exists in $\mathcal{T}_c$; or *ii)* the father, if there is at least one receiver in $\mathcal{T}_f$.

Therefore, using the probabilities τ_f and τ_c defined in (13):

$$\begin{aligned} L_i(e) &= \frac{1}{n}\Big[f(1-\bar{\mu}^{|\mathcal{T}_c|})+(1-\bar{\mu}^{|\mathcal{T}_f|}))\Big]+ \\ &+\left(1-\frac{1}{n}\right)\Big[\tau_f f(1-\bar{\mu}^{|\mathcal{T}_c|})+ \\ &+\tau_c((f-1)(1-\bar{\mu}^{|\mathcal{T}_c|})+ \\ &+(1-\bar{\mu}^{|\mathcal{T}_f|}))\Big] \end{aligned} \quad (16)$$

As with subscriptions, the formula above can be simplified in the case of a full tree, by leveraging Equation (14):

$$\begin{aligned} L_i(e) &= \frac{1}{n}\Big[f(1-\bar{\mu}^{\sum_{i=0}^{h-l-1} f^i})+ \\ &+(1-\bar{\mu}^{n-\sum_{i=0}^{h-l} f^i}))\Big]+ \\ &+\left(1-\frac{1}{n}\right)\Big[\tau_f f(1-\bar{\mu}^{\sum_{i=0}^{h-l-1} f^i})+ \\ &+\tau_c((f-1)(1-\bar{\mu}^{\sum_{i=0}^{h-l-1} f^i})+ \\ &+(1-\bar{\mu}^{n-\sum_{i=0}^{h-l} f^i}))\Big] \end{aligned} \quad (17)$$

Leaves. Leaves forward events only when they publish them and there is at least one receiver in the rest of the tree:

$$L_l(e)=\frac{1}{n}(1-\bar{\mu}^{n-1})$$

3.2.3 Total Forwarding Load for a Node

Similar to what we did for message traffic in Section 3.1.3, we can estimate the total forwarding load on a node in a given interval by considering all events and subscriptions generated in the system:

$$L_x^{SF} = \sum_{p=1}^{F_p} L_x(p)+\sum_{e=1}^{F_e} L_x(e) \qquad x\in\{r,i,l\}$$

4. VALIDATING THE MODEL

To validate the model introduced in Section 3, we implemented the subscription forwarding protocol in PEERSIM [1], a discrete event simulator written in Java and a popular choice for simulating application-level protocols in large-scale scenarios. Our simulator can be configured based on the same input parameters we introduced in the analytical model. We investigated several scenarios with different combinations of parameters, whose default values are summarized in Table 1. We averaged the simulation results over 50 runs, each with different seeds.

Traffic Model. We used as a metric the difference (in percentage) between the traffic derived analytically and through simulation, defined as $\%\Delta T=(T_{model}-T_{sim})/T_{sim}$.

Parameter	Default value
n	21,845
F_p	100
F_e	1000
f	4
h	7
$\sigma(p)$	0.02
$\mu(e)$	0.1

Table 1: Default parameters used in the validation.

	n=341 ($h=4$)	n=1,365 ($h=5$)	n=5,461 ($h=6$)	n=21,845 ($h=7$)	n=87,381 ($h=8$)
$\%\Delta T(p)$	-0.12	0.02	-0.19	0.01	0.005
$\%\Delta T(e)$	-0.30	0.33	0.70	0.23	0.02

(a) h increasing, f fixed.

	n=63 ($f=2$)	n=1,365 ($f=4$)	n=9,331 ($f=6$)	n=37,449 ($f=8$)	n=111,111 ($f=10$)
$\%\Delta T(p)$	1.7	0.02	0.003	-0.002	0.002
$\%\Delta T(e)$	0.36	0.33	0.05	0.03	-0.03

(b) f increasing, h fixed.

Table 2: Simulated vs. theoretical traffic w.r.t. system scale.

σ,μ	0.001	0.01	0.1	0.5	0.8	0.9
$\%\Delta T(p)$	-0.0004	0.01	-0.01	0.005	0.002	-0.003
$\%\Delta T(p)$	0.072	0.32	-0.14	-0.0002	0.004	0.006

(a) Uniform distribution.

α	1.5	1	0.5
$\%\Delta T(p)$	-0.27	-0.09	-0.01
$\%\Delta T(e)$	-2.01	-0.88	-0.56

(b) Zipf distribution.

Table 3: Simulated vs. theoretical traffic w.r.t. distribution of patterns and events. We assume $\sigma(p)=\mu(e)$.

In Table 2, we report the results of the first simulation set, where we analyzed the two approaches w.r.t. system scale. In Table 2(a) we varied the height h of the tree between 4 and 8. Since we kept the number of children f constant, the network size grew proportionally from 341 to 87,381 nodes. Dually, in Table 2(b), we kept the height constant ($h=5$) and used values of f between 2 and 10, therefore increasing the network size from 63 to 111,111 nodes.

In both experiments, the difference w.r.t. our analytical model is always below 0.5% with only one exception (still below 2%) when n is very small. This confirms that our model is accurate regardless the scale of the system and of the particular topology adopted.

The impact of the distribution of patterns and events is assessed in Table 3. Here, we focused on a tree with the default values shown in Table 1, i.e., $n=21,845$ brokers with $f=4$ children per broker ($h=7$). To enable a fair comparison and remove bias, we set $\sigma(p)=\mu(e)$, essentially assuming that all subscriptions are such that they match uniquely one event. This way, we can easily evaluate $T(p)$ against σ and $T(e)$ against μ.

Table 3(a) shows the results in the case of a uniform distribution of events and patterns. Our model appears to be very accurate, with an error w.r.t. simulation results lower than 0.4%. Table 3(b), instead, shows the case of a Zipf distribution, with different values of α. Since we set $\sigma=\mu$, we use the same Zipf distribution for both. The results confirm that our model approximates very closely the simulated behavior also in this case. It is worth noting that the best results are obtained for $\alpha\leqslant 1$, which represents the values commonly found in the literature.

Thus far, we assumed brokers organized in a regular tree with

	$h=4$ $\bar{f}=10$	$h=5$ $\bar{f}=6.22$	$h=6$ $\bar{f}=4.53$	$h=7$ $\bar{f}=3.61$
$\%\Delta T(p)$	0.012%	0.004%	0.045%	0.042%
$\%\Delta T(e)$	0.311%	0.142%	0.045%	0.533%

(a) Standard deviation is 1, increasing height and average number of children.

standard deviation	1	2	3	4	5
$\%\Delta T(p)$	0.005	0.019	0.017	0.043	0.086
$\%\Delta T(e)$	0.091	0.443	0.651	1.789	2.428

(b) Average number of children $\bar{f} = 10$, increasing standard deviation.

Table 4: Simulated vs. theoretical traffic for arbitrary tree topologies. We assume a normal distribution.

	n=341 ($h=4$)	n=1,365 ($h=5$)	n=5,461 ($h=6$)	n=21,845 ($h=7$)	n=87,381 ($h=8$)
average $\Delta L(p)$	-0.374	-0.131	0.018	-0.011	0.003
stdev. $\Delta L(p)$	3.121	1.965	2.002	1.945	1.887
% stdev. $\Delta L(p)$	0.702	0.399	0.400	0.389	0.377
average $\Delta L(e)$	-1.189	-0.157	0.431	-0.129	0.024
stdev. $\Delta L(e)$	28.285	25.788	21.789	20.448	19.882
% stdev. $\Delta L(e)$	0.883	0.644	0.544	0.511	0.499

(a) h increasing, f fixed.

	n=63 ($f=2$)	n=1,365 ($f=4$)	n=9,331 ($f=6$)	n=37,449 ($f=8$)	n=111,111 ($f=10$)
average $\Delta L(p)$	-0.841	0.109	0.027	-0.011	-0.005
stdev. $\Delta L(p)$	6.513	3.181	2.844	2.775	2.557
% stdev. $\Delta L(p)$	2.171	0.636	0.406	0.308	0.232
average $\Delta L(e)$	-0.731	0.366	0.230	-0.031	-0.067
stdev. $\Delta L(e)$	17.333	20.456	19.946	19.123	17.255
% stdev. $\Delta L(e)$	0.866	0.511	0.332	0.239	0.172

(b) f increasing, h fixed.

Table 5: Simulated vs. theoretical load w.r.t. system scale.

exactly f children per node, i.e., the same assumption we introduced in Section 2 for deriving the model. However, in Section 2 we stated that our model can indeed encompass arbitrary topologies and provide accurate estimates, as long as the *average* number of children $\bar{f}$ is known and used in place of f. Here we validate this assumption, assuming a normal distribution for f. Table 4 shows the results. In Table 4(a), we set the standard deviation to 1 and varied $\bar{f}$. The values of $\bar{f}$ are derived by maintaining the fixed default network size while increasing h in unit increments, therefore obtaining increasingly sparse trees. In Table 4(b), instead, we set $\bar{f} = 10$ and varied the standard deviation from 1 to 5. Recall that a standard deviation of 5 means that the number of children is within the interval $[5, 15]$ for 68% of the nodes, and within $[0, 25]$ for 99.7% of the nodes. In this highly variable case, our model is still within 2.5% of the simulated results. Therefore, the results confirm that replacing f with $\bar{f}$ in all the formulae indeed provides accurate results, even with very sparse and very variable topologies.

Load Model. To validate our analytical model of the node forwarding load, we carried out a simulation campaign similar to the one we showed for the overall traffic. Nevertheless, we took a more conservative (i.e., less favorable for our model) choice, with a much lower probability for patterns and events ($\sigma = \mu = 0.001$) yielding a load distribution among nodes with more marked variations. The other default parameters in Table 1 are unchanged.

We report the difference $\Delta L = (L_{model} - L_{sim})$ (in absolute value) between the load on each node as derived analytically and through simulation, averaged over all nodes, along with its standard deviation (absolute and in percentage w.r.t. the maximum load).

σ, μ	0.001	0.01	0.1	0.5	0.8	0.9
average $\Delta L(p)$	0.017	-0.017	0.005	0.103	0.023	0.094
stdev. $\Delta L(p)$	1.261	1.810	1.100	1.004	0.499	0.231
% stdev. $\Delta L(p)$	0.252	0.362	0.220	0.200	0.099	0.046
average $\Delta L(e)$	0.018	-0.037	0.084	0.097	-0.034	0.005
stdev. $\Delta L(e)$	6.918	15.629	23.961	25.931	11.100	7.227
% stdev. $\Delta L(e)$	0.223	0.390	0.599	0.648	0.277	0.180

(a) Uniform distribution.

α	1.5	1	0.5
average $\Delta L(p)$	-0.021	0.054	-0.101
stdev. $\Delta L(p)$	10.443	7.221	4.042
% stdev. $\Delta L(p)$	2.088	1.444	0.808
average $\Delta L(e)$	0.456	0.225	0.004
stdev. $\Delta L(e)$	35.173	40.213	53.104
% stdev. $\Delta L(e)$	1.004	1.005	1.327

(b) Zipf distribution.

Table 6: Simulated vs. theoretical load w.r.t. distribution of patterns and events. We assume $\sigma(p) = \mu(e)$.

	$h=4$ $\bar{f}=10$	$h=5$ $\bar{f}=6.22$	$h=6$ $\bar{f}=4.53$	$h=7$ $\bar{f}=3.61$
average $\Delta L(p)$	-0.095	-0.023	0.029	0.025
stdev. $\Delta L(p)$	2.409	3.581	1.994	1.567
% stdev. $\Delta L(p)$	0.185	0.329	0.249	0.223
average $\Delta L(e)$	0.159	-0.121	0.132	-0.091
stdev. $\Delta L(e)$	10.541	12.313	17.234	14.775
% stdev. $\Delta L(e)$	0.307	0.476	0.615	0.484

(a) Standard deviation is 1, increasing height and average number of children.

standard deviation	1	2	3	4	5
average $\Delta L(p)$	-0.085	0.071	-0.037	0.145	0.041
stdev. $\Delta L(p)$	1.704	6.725	6.062	7.372	7.245
% stdev. $\Delta L(p)$	0.154	0.611	0.551	0.670	0.658
average $\Delta L(e)$	-0.191	0.072	0.131	-0.277	-0.041
stdev. $\Delta L(e)$	24.026	65.900	71.222	75.430	80.147
% stdev. $\Delta L(e)$	0.221	0.409	0.612	0.624	0.671

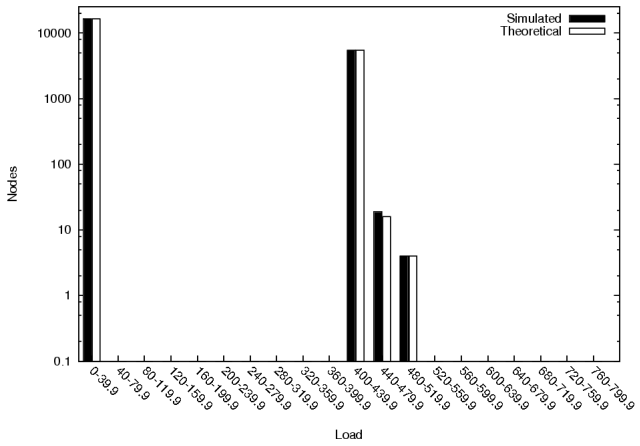
(b) Average number of children $\bar{f} = 10$, increasing standard deviation.

Table 7: Simulated vs. theoretical load for arbitrary tree topologies. We assume a normal distribution.

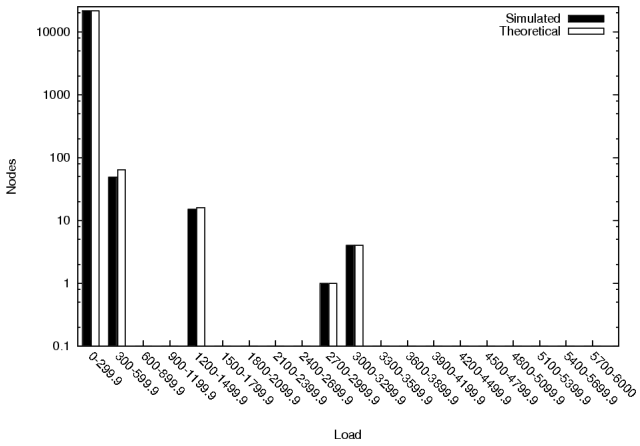
Table 5 and 6 show the results w.r.t. system scale and different distributions of patterns and events, respectively, assuming a full, regular tree. These experiments confirm the good results achieved for the model of overall traffic in analogous settings, shown in Table 2 and 3. As the reader can see, the average value exhibits a minimal difference between the simulated and theoretical values, and its standard deviation is very small. Therefore, our analytical model indeed is able to accurately estimate also the forwarding load. To further prove this, Figure 4 compares the load distribution (i.e., how many nodes experience a given load) obtained through simulation against the one predicted by our model. The reader can appreciate visually how the distributions are very similar, at most showing small differences.

As for arbitrary tree topologies, unlike with traffic we cannot simply replace f with the average value $\bar{f}$ in Equation (15) and (17). In fact, the load on every node has a stronger dependence on the underlying topology; using the average value in some cases may lead to gross errors. For instance, consider a hypothetical tree where the root is connected to all the internal nodes, which in turn all have exactly one child. The average value $\bar{f}$ in this case is close[2] to 2.

[2]More precisely, $\bar{f} = \frac{2I}{I+1}$, being I the number of internal nodes.

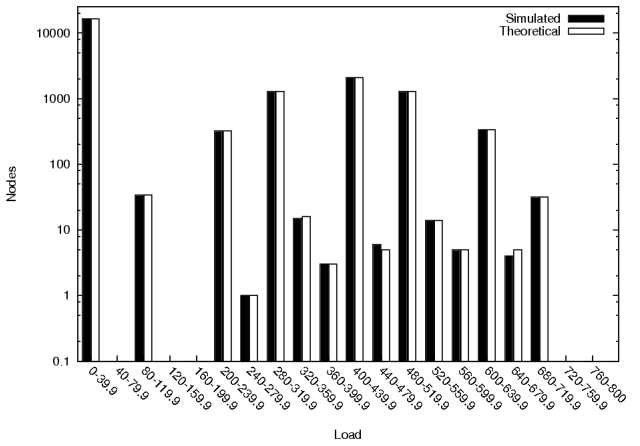


(a) Subscriptions.

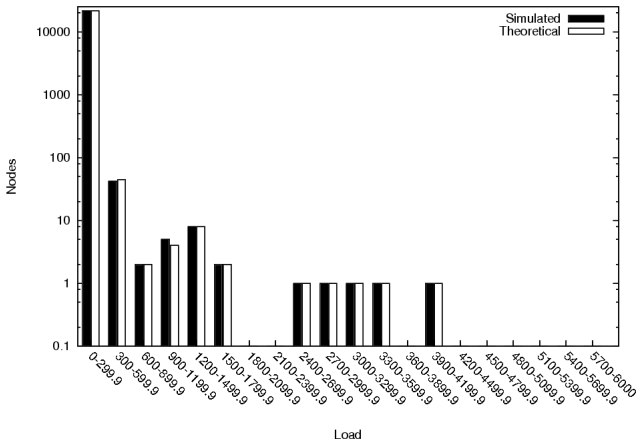


(b) Events.

Figure 4: Forwarding load on full trees ($\sigma = \mu = 0.001$).



(a) Subscriptions.



(b) Events.

Figure 5: Forwarding load on arbitrary trees ($\sigma = \mu = 0.001$).

However, the load on the root would be quite different from the one in a binary tree, where nevertheless $\bar{f} = 2$ holds as well.

Nevertheless, as we mentioned in Section 3.2, if more information about the tree topology is available, we can compute for each node the number of nodes contained in the sub-trees $\mathcal{T}_f$ and $\mathcal{T}_c$, as well as the actual value of f for the given node, and use Equation (12) and (16). Note how this information about $|\mathcal{T}_c|$, $|\mathcal{T}_f|$, and f is easy to obtain in practice. Indeed, the topology is often determined by the system designer or, in the case of systems already deployed, it can be easily discovered at run-time. Interestingly, the availability of such information would allow one to compute $|\mathcal{T}_l|$ in Equation (3), therefore enabling better estimates of traffic as well.

The experiments we performed generated topologies by using normal distributions, with the same criteria we used for traffic. In addition, this time we computed the values of f, $|\mathcal{T}_f|$, and $|\mathcal{T}_c|$ over all the nodes for each topology generated, and used them in the model. The results, shown in Table 7, confirm once more that our model is very close to the simulated behavior. This can be observed also in Figure 5, where analogously to what we did for full trees we plot the distribution of load over nodes of arbitrary trees.

In summary, the results we showed confirm that our model can effectively replace simulations, with the practical effect of drastically reducing the amount of time required to reproduce the behavior of subscription forwarding. Moreover, since the model is an accurate representation of this protocol, it enables a more fine-grained understanding of its behavior, which can be used at design time or to enable novel research insights. Some examples of use of our analytical model are shown next.

5. USING THE MODEL

In this section, we show how the model presented in Section 3 can be used in practice to analyze the trade-offs among different solutions and support system design and deployment. We focus on three examples. First, we compare subscription forwarding and event forwarding in terms of generated message traffic. The trade-offs between the two protocols are well-known, and our analysis bears no surprises. However, the novelty is that our model enables one to characterize precisely the break-even point between the two protocols. The second example shows how the results we derived for the node forwarding load can be used in practice in conjunction with application profiles, to identify bottlenecks in the system. Finally, our third example shows that the model can be used to determine the optimal shape of the tree overlay, by choosing the value of f that best suits the characteristics of the scenario at hand.

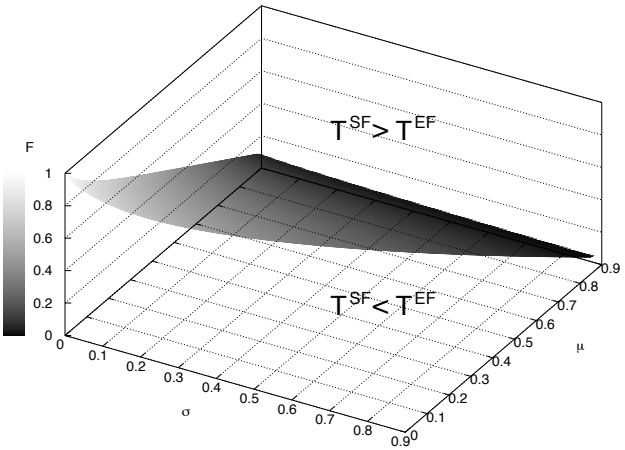


Figure 6: The function $F(\sigma, \mu)$, representing the values of $F = \frac{F_p}{F_e}$ for which subscription forwarding and event forwarding generate the same message traffic. The chart assumes uniform distribution of patterns and events, $n = 1111111$, $f = 10$.

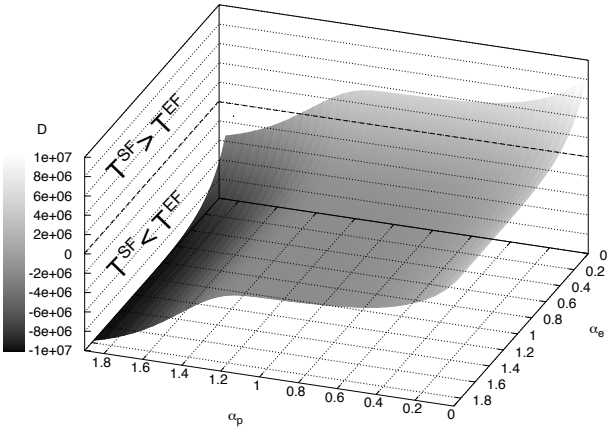


Figure 8: The function $D(\alpha_p, \alpha_e)$, representing the difference between the traffic generated by event forwarding and the traffic generated by subscription forwarding. The chart assumes a Zipf distribution of patterns and events, $n = 1111111$, $f = 10$.

Note that the goal of this section is not to provide extensive evaluations or comparisons. These are indeed possible, but we are here constrained by space limits. Instead, we want to provide the reader with a feel of the *potential* of our model, when used in practice.

Overall Traffic: Subscription vs. Event Forwarding. As our first example, we illustrate how our model can be used to compare in terms of message overhead the two classical CBR protocols, subscription forwarding and event forwarding, and assess the one most suited to a given scenario. To further stress the versatility and power of our model, in the following we show results that are obtained with a network size of $n = 1,111,111$ nodes, one that is quite impractical to analyze through simulation.

We begin our analysis under the assumption of uniform distribution. To compare the two approaches, we equate Equation (1) and (9) and, leveraging off the fact that σ and μ are constant, solve w.r.t. $F = \frac{F_p}{F_e}$. We obtain a curve $F(\sigma, \mu)$ that describes the combination of parameters for which the two protocols generate the same amount of traffic. If F_{real} is the actual ratio between subscriptions and events in a given scenario, $F_{real} > F(\sigma, \mu)$ means that the event forwarding generates less traffic, while $F_{real} < F(\sigma, \mu)$ means that subscription forwarding is more efficient, in a scenario with the given σ and μ.

The function $F(\sigma, \mu)$ is plotted[3] in Figure 6, for $f = 10$. To better understand the trends, in Figure 7 we also charted the projections of $F(\sigma, \mu)$ on the planes (σ, μ), (σ, F), (μ, F).

The charts visualize the well-known tradeoffs between the two protocols. In Figure 6, the shape of the surface $F(\sigma, \mu)$ shows that when the number of patterns becomes comparable with the number of events (i.e., approaching $F = 1$), event forwarding is more efficient because it avoids disseminating subscriptions. On the other hand, if events are generated at a higher frequency than patterns, subscription forwarding is preferable because it optimizes event propagation. However, these tradeoffs are affected by the values of σ and μ, which are best appreciated in Figure 7. For instance, Figure 7(a) shows the projection of $F(\sigma, \mu)$ on the plane (σ, μ), where the curves represent points yielding the same value of F: values of μ below a curve mean that the point is below the $F(\sigma, \mu)$ surface, and therefore subscription forwarding is preferable. For instance, if $F = 0.2$, subscription forwarding is preferable only if the pair (σ, μ) is in the shadow area in Figure 7(a). Interestingly, as μ increases, the effect of σ becomes negligible and does not affect the value of F. Indeed, if $\mu \sim 1$, the vast majority of nodes are receivers and therefore event forwarding is always the best choice, regardless of the value of σ.

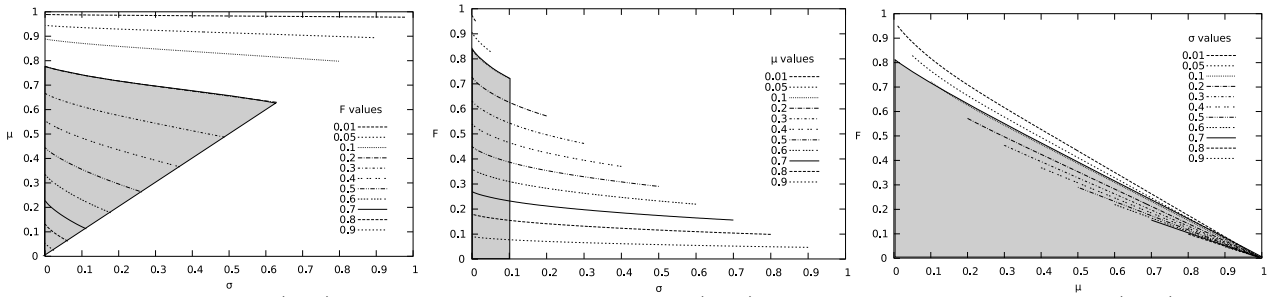
As we already mentioned, these charts give a precise characterization of the tradeoffs by identifying the break-even point. Interestingly this is also helpful in choosing the parameters for the scenarios to analyze through simulation. For instance, a commonly used value in simulated scenarios is $\mu = 0.1$ (e.g., in [5]). Figure 7(b) shows that in our network this is a reasonable assumption for subscription forwarding only if the values F and σ fall inside the shadowed area, otherwise event forwarding is actually preferable. Similarly, if we decided to set $\sigma = 0.1$, the values of F and μ must be chosen among those in the shadowed area in Figure 7(c).

A similar analysis can be carried out using a Zipf distribution instead of a uniform one. The results are shown in Figure 8. Notably, since σ and μ are no longer constant, the chart reports instead the coefficients of the corresponding Zipf distributions, α_p and α_e, respectively. Moreover, in this case the shape of the equations is such that the traffic can no longer be expressed as a function of F. Therefore, we plot the function $D(\alpha_p, \alpha_e) = T^{SF} - T^{EF}$.

Interestingly, when α_p is high (to the left in the chart), D is always negative (i.e., subscription forwarding always performs better) regardless of the value of α_e. The reason lies in the fact that if α_p is high, most subscriptions revolve around a restrict set of patterns, the most popular ones. This implies that the traffic due to the propagation of subscriptions is small and, therefore, the efficient event dissemination of subscription forwarding pays off. Conversely, if α_p is close to zero, basically each subscriptions is tied to a different pattern and this dramatically affects the performance of subscription forwarding, because subscriptions for a new pattern must be flooded to all the brokers. In these cases, $D(\alpha_p, \alpha_e)$ is positive, and therefore event forwarding is preferable.

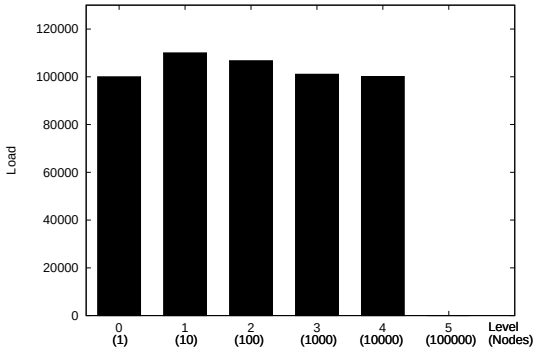
The considerations we made for the case of uniform distribution, e.g., about the support for setting simulation parameters, still hold. However, we note that, given the shape of the function $D(\alpha_p, \alpha_e)$ characterized by a saddle in the middle, the precise tradeoffs would

[3]Note that the domain of the function is restricted to only half of the plane since we impose $\mu \geq \sigma$ as explained in Section 2.

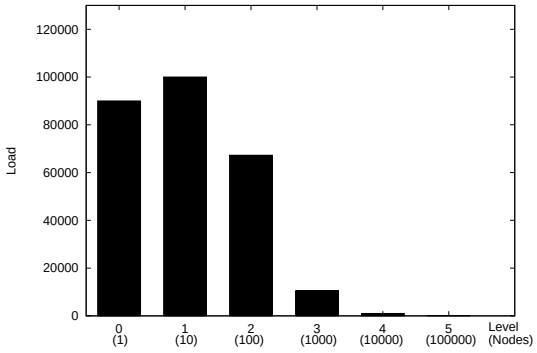


(a) Projection on (σ, μ). (b) Projection on (σ, F). (c) Projection on (μ, F).

Figure 7: Projections of the function $F(\sigma, \mu)$ shown in Figure 6 on the three planes (σ, μ), (σ, F), (μ, F). Shadowed areas identify the value combinations where subscription forwarding is more convenient, for values of $F = 0.2$, $\mu = 0.1$, and $\sigma = 0.1$, respectively.



(a) Subscription load.



(b) Event load.

Figure 9: Message forwarding load against node level for subscription forwarding, in a network with $n = 111,111$, $f = 10$, $\sigma = 0.1\%$, $\mu = 1\%$, $F_p = F_e = 10,000$. Values between parentheses indicate the number of nodes on each level.

be even more difficult to "guess" without the characterization provided by our model.

Identifying Bottlenecks. A prominent feature of our model is the ability to accurately estimate the forwarding load experienced by each node. This enables one to identify bottlenecks at design time and remove them, e.g., by replicating the most overloaded nodes.

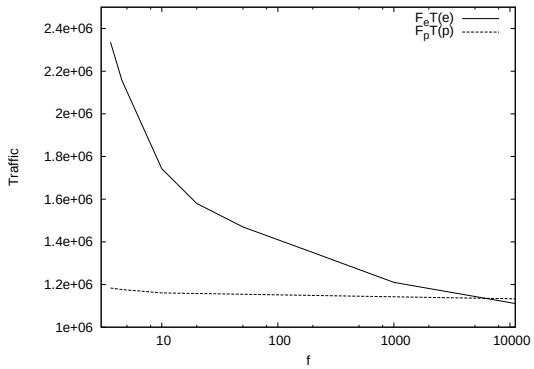
An example is provided in Figure 9, where we plot the message forwarding load per node against the level of the node. The chart assumes a subscription forwarding protocol with $n = 111,111$, $f = 10$, $\sigma = 0.1\%$, $\mu = 1\%$. Moreover, we set $F_p = F_e = 10,000$: although these values are actually unfavorable for subscription forwarding they are useful here to show how patterns and events stress differently the broker tree.

Under these settings, comparing Figure 9(a) and 9(b) shows that while the forwarding load due to subscriptions is distributed fairly evenly, the one due to events concentrates on the nodes at high levels. This is a direct consequence of the fact that, in subscription forwarding, the first subscription for a pattern must always be flooded to all brokers, while events are propagated only towards intended receivers. Therefore, most of the nodes handle most of the subscriptions, while only a small set of nodes, namely those placed in the middle of the tree (i.e., at the lowest levels) dispatch most of the events. In particular, Figure 9(b) shows that the maximum forwarding load due to events is incurred by the nodes on the second level. Indeed, not only these nodes dispatch most of the events because of their central positions in the tree, but they also have one neighbor more than the root (i.e., their fanout is $f + 1$).

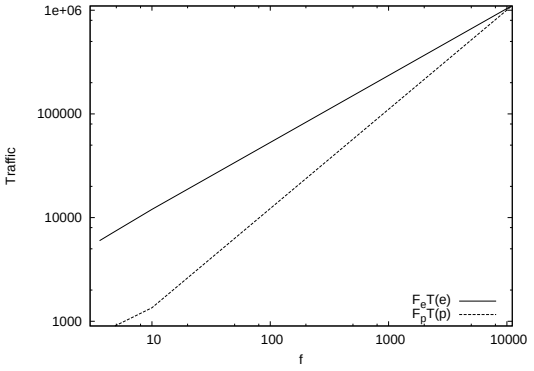
Looking at pictures like these, engineers can immediately realize before deployment time which nodes will experience high load and take appropriate countermeasures, without the need to resort to expensive and time-consuming simulations.

Overlay Management: The Impact of f. Thus far, we assumed that the number of children f was a parameter characterizing the network scenario. Nevertheless, in many cases, f is under the control of the designer, and represents a "knob" that can be used to optimize performance. For instance, in the case of subscription forwarding, the higher the f the more efficient event dissemination is, because the tree has fewer levels and therefore events are steered more quickly towards receivers. However, on the other hand, increasing f results in a higher forwarding load imposed on nodes, thus preventing scalability. These tradeoffs are immediately evident when applied to a star topology, which exhibits the lowest event traffic ($T(e) = \mu(n-1)$) but also imposes a huge load on the root ($n-1$ messages per pattern, and all of the $\mu(n-1)$ messages per event).

Figure 10 shows how this analysis can be carried out in practice, by plotting the message overhead and the *maximum* forwarding load against the node degree f, and by keeping the contributions due to patterns and events separate. The chart assumes a scenario with $n = 11,111$, $\sigma = 0.02$, $\mu = 0.1$, $F_p = 100$, $F_e = 1,000$. As expected, Figure 10(a) shows that the traffic due to events significantly decreases when f increases, while the subscriptions traffic is reduced only marginally. Indeed, since most subscriptions are dis-



(a) Message overhead.



(b) Maximum forwarding load.

Figure 10: Message overhead and maximum forwarding load against f. The components due to patterns and events are plotted separately. The chart assumes a subscription forwarding protocol and $n = 11,111$, $\sigma = 0.02$, $\mu = 0.1$, $F_p = 100$, $F_e = 1,000$.

seminated to all nodes, the impact of f is negligible[4]. Conversely, as depicted in Figure 10(b), the maximum forwarding load dramatically increases with f, for both subscriptions and events.

By relying on the information derived from similar analysis performed with our model, engineers can determine the value of f that guarantees the right balance between traffic and load, w.r.t. the scenario workload and the forwarding capacity of each node.

6. CONCLUSIONS AND FUTURE WORK

We presented an analytical model of the communication costs, in terms of overall message traffic and node forwarding load, of the subscription forwarding CBR protocol. This model constitutes, to the best of our knowledge, the most complete among the rare ones in the literature, and the only one validated through simulation.

Moreover, the examples we discussed in the last section evidence the important role an analytical model like ours can play during the design phase. Indeed, thanks to this tool, practitioners can explore the parameter space and identify appropriate solutions more quickly and reliably than with simulation. On the other hand, by formally capturing the properties of subscription forwarding the model is clearly very valuable also to researchers, in that not only it precisely characterizes this basic strategy, but also its techniques can be used as a stepping stone for modeling alternative strategies.

Our future work will indeed proceed in this direction, by reusing and adapting the techniques described here to other tree-based protocols (e.g., hierarchical forwarding and variations using advertisements), thus enabling an extensive and formally grounded comparison among them. Moreover, we are working to remove the assumption of a uniform distribution of subscribers and publishers, therefore increasing even further the applicability of our model.

Acknowledgements. This work was partially supported by the European Community under the IST-004536 RUNES project.

7. REFERENCES

[1] `http://peersim.sourceforge.net`.

[2] A. Carzaniga and C.P. Hall. Content-Based Communication: a Research Agenda. In *In Proc. of Software Engineering and Middleware Workshop (SEM)*, November 2006.

[3] A. Carzaniga, A.J. Rembert, and A.L. Wolf. Understanding Content-Based Routing Schemes. Technical Report 2006-05, University of Lugano, September 2006.

[4] A. Carzaniga, D.S. Rosenblum, and A.L. Wolf. Design and Evaluation of a Wide-Area Event Notification Service. *ACM Trans. on Computer Systems*, 19(3):332–383, 2001.

[5] A. Carzaniga, M. Rutherford, and A. Wolf. A routing scheme for content-based networking. In *Proc. of INFOCOM*, 2004.

[6] P. Costa and G.P. Picco. Semi-probabilistic content-based publish-subscribe. In *Proc. of the 25th Int. Conf. on Distributed Computing Systems (ICDCS)*, June 2005.

[7] G. Cugola, E. Di Nitto, and A. Fuggetta. The JEDI Event-Based Infrastructure and its Application to the Development of the OPSS WFMS. *IEEE Trans. on Software Engineering*, 27(9):827–850, September 2001.

[8] A. Gupta, O. Sahin, D. Agrawal, and A. El Abbadi. Meghdoot: content-based publish/subscribe over P2P networks. In *Proc. of the 5th Int. Conf. on Middleware*, 2004.

[9] M. Jaeger and G. Mühl. Stochastic Analysis and Comparison of Self-Stabilizing Routing Algorithms for Publish/Subscribe Systems. In *Proc. of the 13th IEEE Int. Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2005.

[10] G. Mühl, L. Fiege, and P.R. Pietzuch. *Distributed Event-Based Systems*. Springer, August 2006.

[11] P. Pietzuch and J. Bacon. Hermes: A Distributed Event-Based Middleware Architecture. In *Proc. of the 2nd Int. Workshop on Distributed Event-Based Systems*, 2002.

[4]Note that this is not necessarily true in general: with other combinations of parameters, the subscription traffic exhibits a more marked decrement. For instance, with few patterns in the system, only few subscriptions are flooded to the whole tree while the majority are sent only to a subset of nodes; higher values of f can reduce the path length.