On Novelty in Publish/Subscribe Delivery

Dimitris Souravlias, Marina Drosou, Kostas Stefanidis and Evaggelia Pitoura

Computer Science Department, University of Ioannina, Greece
{dsouravl, mdrosou, kstef, pitoura}@cs.uoi.gr

***Abstract*— In publish/subscribe systems, users express their interests in specific items of information and get notified when relevant data items are produced. Such systems allow users to stay informed without the need of going through huge amounts of data. However, as the volume of data being created increases, some form of ranking of matched events is needed to avoid overwhelming the users. In this work-in-progress paper, we explore novelty as a ranking criterion. An event is considered novel, if it matches a subscription that has rarely been matched in the past.**

I. Introduction

With the explosion of the amount of information that becomes available online, publish/subscribe systems offer an attractive alternative to search by providing a proactive model of information supply. In such systems, users (or subscribers) express their interest in specific pieces of data (or events) via queries called subscriptions. Then, they are notified whenever some information source (or publisher) generates an event that matches one of their subscriptions. Examples of such proactive delivery include news alerts, RSS feeds and notification services in social networks.

Typically, all subscriptions are considered equally important and users are notified whenever a published event matches any of their subscriptions. However, as with search, user subscriptions are often exploratory in nature. Thus, recent research has suggested that event matching should be best effort by associating some form of ranking to the matching process. The rank associated with each matched event may depend on the user preferences or interests [6], [11], on the authority or relevance of its publisher [12] or, in the case of fuzzy or approximate matching, on the degree of relevance between the event and the matching subscriptions [10], [9].

In our previous work [6], [5], we have focused on ranked publish/subscribe delivery based on preferences and content diversity. In particular, we considered delivering to each user those k events among the matched ones that are both highly-ranked based on user preferences and also have different content with each other. In this work-in-progress paper, we present another aspect of ranking based on subscription novelty. Novelty and diversity are gaining increasing interest in information retrieval as evaluation measures along with relevance. Whereas there is no standard definition for either, one can define novelty as the need to limit redundancy by avoiding results with overlapping content and diversity as the need to resolve ambiguity by including results that cover different topics (for example, "Jaguar" as a car, a cat and the classic Fender guitar) [4]. Our interpretation of novelty in this work is that *an event is novel if it matches a subscription that has rarely been matched in the past*. This form of novelty is desirable for various reasons. We outline below two of them: making rare events visible and allowing expressing an information need with various levels of detail.

Consider a user that poses subscriptions with different and varying rates of matching events. As an example, take a user in a social networking application that follows both friends that are very productive in terms of content generation and friends that post information only seldom. Novel events (i.e. events that correspond to subscriptions that are rarely matched) will get high ranks and get noticed by the subscriber instead of potentially being overwhelmed by other less novel events. As another motivating application, novelty ranking allows users to express subscriptions with different levels of granularity. For example, a user may subscribe to both "movies" and "horror movies". The "horror movies" subscription is redundant in a publish/subscribe system without ranking, since an event that matches "horror movies" also matches "movies" and will be delivered to the user anyway. With novelty, an event that matches a detailed subscription, such as "horror movies", will implicitly get a higher rank than an event that matches only a more general one, such as "movies".

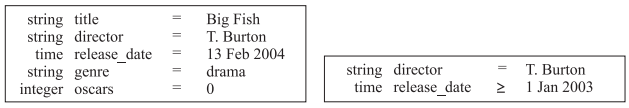
In the rest of this paper, we first present our publish/subscribe model, then formally define novelty and finally present some initial experimental results.

II. Publish/Subscribe Model

In general, a publish/subscribe system consists of three parts: (i) the publishers that provide events to the system, (ii) the subscribers that enter subscriptions and consume events and (iii) a notification service that stores the various subscriptions, matches the incoming events against them and delivers the matching events to the appropriate subscribers [7].

The form of events and subscriptions depends on the specific application. In this work, we use a generic content-based model to form events and subscriptions, similar to the one used, for example, in [3], [6] and [8]. In particular, events are sets of attributes. Each event consists of an arbitrary number of attributes and each attribute has a type, a name and a value. Attribute types belong to a predefined set of primitive types, such as "integer" or "string". Attribute names are character strings that take values according to their type. An example event about a movie is shown in Fig. 1a. Formally:

An *event* e is a set of attributes $\{a_1, \ldots, a_p\}$, where each a_i, $1 \leq i \leq p$, is of the form $(a_i.type\ a_i.name = a_i.value)$.



(a) (b)

Fig. 1. (a) Event and (b) subscription examples.

Subscriptions are used to specify the kind of events users are interested in. Each subscription consists of a set of constraints on the values of specific attributes. Each attribute constraint has a type, a name, a binary operator and a value. Types, names and values have the same form as in events. Binary operators include common operators, such as $=$, $<$, $>$ and $*$ (substring). An example subscription is depicted in Fig. 1b. Formally:

A *subscription* s is a set of attribute constraints $\{b_1, \ldots, b_q\}$, where each b_i, $1 \leq i \leq q$, is of the form ($b_i.type$ $b_i.name$ θ_{b_i} $b_i.value$), $\theta_{b_i} \in \{=, <, >, \leq, \geq, \neq, *, > *, * <\}$.

Intuitively, we can say that an event e *matches* a subscription s, or alternatively s *covers* e, if and only if, every attribute constraint of s is satisfied by some attribute of e. Formally:

Definition 1: (COVER RELATION BETWEEN EVENTS AND SUBSCRIPTIONS). Given an event $e = \{a_1, \ldots, a_p\}$ and a subscription $s = \{b_1, \ldots, b_q\}$, s covers e, $s \succ_E^S e$, if and only if, $\forall b_j \in s$, $\exists a_i \in e$, such that, $a_i.type = b_j.type$, $a_i.name = b_j.name$ and $((a_i.value)\ \theta_{b_j}\ (b_j.value))$ holds, $1 \leq i \leq p$, $1 \leq j \leq q$.

An event e is delivered to a user, if and only if, the user has submitted at least one subscription s, such that, $s \succ_E^S e$. For example, the subscription of Fig. 1b covers the event of Fig. 1a and, therefore, this event will be delivered to all users who have submitted this subscription.

III. NOVELTY-AWARE PUBLISH/SUBSCRIBE

We aim at enhancing notification services by introducing a degree of importance for each delivered event. This degree expresses the novelty of the event for the receiver and is computed by taking into account the subscriptions of the receiver and any previously delivered events.

Let $\mathcal{U}$ be the set of users of a publish/subscribe system and $\mathcal{S}$ be the set of their subscriptions. Given a user $u \in \mathcal{U}$, we use $\mathcal{S}_u$, $\mathcal{S}_u \subseteq \mathcal{S}$, to denote the subscriptions of u.

A. Subscription Novelty

Intuitively, users are more interested in rare pieces of information, in the sense that being notified about something that rarely happens is more important than being notified about something that occurs all the time. Therefore, we would like to favor subscriptions that are not frequently matched by the published events. The *novelty* of a subscription captures this property by measuring how frequently the subscription is matched. Generally, the novelty of a subscription s is decreased whenever an event e is published, such that, $s \succ_E^S e$.

Assuming some initial default novelty for all subscriptions, we can formally define the novelty of a subscription recursively as follows:

Definition 2: (SUBSCRIPTION NOVELTY). Given a subscription $s \in \mathcal{S}$, after i events have matched s, the novelty of s, $nov^{(i)}(s)$, is:

$$nov^{(i)}(s) = \begin{cases} nov^{(i-1)}(s) - \alpha, & \text{if } i \geq 1 \\ \beta, & \text{if } i = 0 \end{cases}$$

where α and β are positive constants.

The α parameter calibrates the reduction rate of novelty. Here, we follow a linear approach. However, one could argue about other variations, such as an exponential reduction. We assume that the default novelty β of all subscriptions is the same. Alternatively, we could use a different default value for each subscription based, for example, on user preferences, similarly to our approach in [6], on relevance or some other criterion.

Event matching is continuous, thus, the novelty of each subscription keeps reducing as events match it over time. In this paper, we adopt a simple periodic model for refreshing novelty. We assume time intervals or periods of a fixed length T. The novelty of each subscription is reset to its default value β at the beginning of each period. We use $nov(s, t)$ to denote the value of novelty of subscription s at time instant t.

B. Event Degree of Importance

A published event e is delivered to all users who have submitted subscriptions that cover it. Each event presented to a user u is associated with a degree of importance. This degree is computed with respect to the novelty of its matching subscriptions. In the simple case, where e matches only one subscription $s \in \mathcal{S}_u$, the degree of importance of e for u is equal to the novelty of s. However, in most cases, there are more than one such subscriptions. In these cases, we base the computation of the degree of importance of the event on the novelty of the *most specific* subscriptions of u for e.

To define the most specific subscriptions for an event, we first need to define the cover relation among two subscriptions:

Definition 3: (COVER RELATION BETWEEN SUBSCRIPTIONS). Given two subscriptions s and s', s covers s', $s \succ_S^S s'$, if and only if, for each event e, such that, $s' \succ_E^S e$, it holds that $s \succ_E^S e$.

Now, given all subscriptions submitted by u that cover an event e, denoted $\mathcal{S}_u^e$, we say that a subscription $s \in \mathcal{S}_u^e$ is a *most specific* one of u for e, if and only if, there is no other subscription $s' \in \mathcal{S}_u^e$, such that, $s \succ_S^S s'$. Formally:

Definition 4: (MOST SPECIFIC SUBSCRIPTION). Let $u \in \mathcal{U}$ be a user and s a subscription, such that, $s \in \mathcal{S}_u$. Given an event e, we say that s is a most specific subscription of u for e, if and only if:

1) $s \succ_E^S e$ and
2) $\nexists s' \in \mathcal{S}_u$, $s' \neq s$, such that, $s' \succ_E^S e$ and $s \succ_S^S s'$.

For example, assume the event of Fig. 1a and the subscriptions {genre = drama} and {genre = drama, director = T. Burton} submitted by the same user (for ease of presentation, we omit the type of each attribute). Both subscriptions cover the event. Between the two, the latter subscription is more specific than the former one, in the sense that in the latter subscription the user imposes an additional, more specific requirement to movies.

We next define the importance of an event with regards to the novelty of the most specific subscriptions.

Definition 5: (EVENT DEGREE OF IMPORTANCE). Given an event e, a user $u \in \mathcal{U}$ and a set of subscriptions $\mathcal{S}$, the degree of importance of e for u at time t, $doi(e, u, \mathcal{S}, t)$, is:

$$doi(e, u, \mathcal{S}, t) = \max_{s \in \mathcal{S}_u^e} nov(s, t)$$

where $\mathcal{S}_u^e$ is the set of the most specific subscriptions of u for e and $nov(s, t)$ is the novelty of s at the time instant t of the doi computation.

There are two subtle points regarding Definition 5. First, an event gets the novelty of the most specific subscriptions matching it. Note, however, that by Definition 2, we update the novelty of *all* subscriptions that match an event. In the example, we update the novelty of both subscriptions {genre = drama} and {genre = drama, director = T. Burton}. Thus, the novelty of general subscriptions, such as {genre = drama}, tends to be reduced faster than the novelty of most specific ones, such as {genre = drama, director = T. Burton}. Consequently, events that match very specific information needs of a user and may be rarely generated tend to get high scores in general. Second, when more than one most specific subscription matches an event, we use the maximum novelty. To see why, take again the event in Fig. 1a and the subscriptions {genre = drama, director = T. Burton} and {genre = drama, release_date = 13 Feb 2004}. Both subscriptions match the event and none of the two covers the other. The event will get the best novelty. For example, if there have been many events about drama movies by T. Barton but very few events about drama movies released on 13 Feb 2004, the event in Fig. 1a will get the novelty of the second subscription, that is, the highest one, so that such rare events are noticed.

IV. PRELIMINARY EVALUATION

To evaluate our approach, we have extended the SIENA notification service [2] with our novelty functionality. Our goal is to demonstrate that novel events are brought to the foreground. To do this, we use a set of subscriptions constructed based on a real movie dataset [1] and generate events that match the subscriptions uniformly. We report results for a subset $S = \{s_1, \ldots, s_7\}$ of the subscriptions, where s_1 covers s_3 and s_4, and s_2 covers s_5, s_6 and s_7. There are 1000 events, each one covered by at least one subscription in S. In Fig. 2, we show the change in the novelty of each subscription as events are being published. We use $\alpha = 0.001\beta$. The more general the subscription, the higher the reduction rate of its novelty. Also,

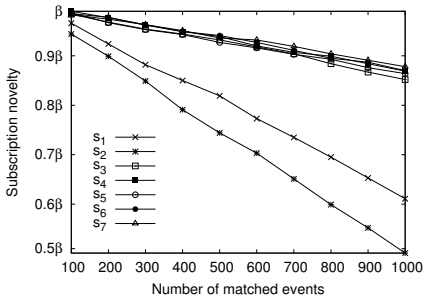


Fig. 2. Subscription novelty per number of matched events during one period.

the reduction rate of the novelty of a subscription increases as the number of the subscriptions covered by it increases.

V. CONCLUSIONS

In this short paper, we have argued for making novelty a ranking criterion in publish/subscribe systems. There are many issues for further research. First, novelty is only one of the criteria to characterize the importance of an event. Other possible criteria include relevance, source authoritativeness, diversity and user preferences. How to combine such criteria for effectiveness is a difficult problem. Then, we have assumed that all ranked matching events are delivered to users. It is reasonable to define a top-k variant of the problem, where only the k events with the highest degrees are delivered, as well as, a threshold-based variant of the problem, where only the events having a degree above a threshold are delivered. Furthermore, since publish/subscribe provides a form of continuous delivery, we are looking into a sliding window model for novelty to replace our periodic one. Finally, there are implementation and performance issues that we have not considered in this paper that are worthy of further research, such as an efficient architecture for the matching service.

REFERENCES

[1] Movies dataset. http://had.co.nz/data/movies.

[2] SIENA. http://serl.cs.colorado.edu/∼serl/dot/siena.html.

[3] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Trans. Comput. Syst.*, 19(3):332–383, 2001.

[4] C. L. A. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *SIGIR*, pages 659–666, 2008.

[5] M. Drosou and E. Pitoura. Diversity over continuous data. *IEEE Data Eng. Bull.*, 32(4):49–56, 2009.

[6] M. Drosou, K. Stefanidis, and E. Pitoura. Preference-aware publish/subscribe delivery with diversity. In *DEBS*, pages 1–12, 2009.

[7] P. T. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec. The many faces of publish/subscribe. *ACM Comput. Surv.*, 35(2):114–131, 2003.

[8] F. Fabret, H.-A. Jacobsen, F. Llirbat, J. Pereira, K. A. Ross, and D. Shasha. Filtering algorithms and implementation for very fast publish/subscribe. In *SIGMOD*, pages 115–126, 2001.

[9] H. Liu and H.-A. Jacobsen. Modeling uncertainties in publish/subscribe systems. In *ICDE*, pages 510–522, 2004.

[10] A. Machanavajjhala, E. Vee, M. N. Garofalakis, and J. Shanmugasundaram. Scalable ranked publish/subscribe. *PVLDB*, 1(1):451–462, 2008.

[11] K. Pripuzic, I. P. Zarko, and K. Aberer. Top-k/w publish/subscribe: finding k most relevant publications in sliding time window w. In *DEBS*, pages 127–138, 2008.

[12] C. Zimmer, C. Tryfonopoulos, K. Berberich, M. Koubarakis, and G. Weikum. Approximate information filtering in peer-to-peer networks. In *WISE*, pages 6–19, 2008.