Node Behavior Prediction for Large-Scale Approximate Information Filtering*

Christian Zimmer[1], **Christos Tryfonopoulos**[1], **Klaus Berberich**[1], **Gerhard Weikum**[1] **and Manolis Koubarakis**[2]
[1] Max Planck Institute for Informatics
Department of Databases and Information Systems, 66123 Saarbruecken, Germany
{czimmer, trifon, kberberi, weikum}@mpi-inf.mpg.de
[2] National and Kapodistrian University of Athens
Department of Informatics and Telecommunications, GR715784 Athens, Greece
koubarak@di.uoa.gr

ABSTRACT

In this paper we investigate methods that allow us to identify the publishing behavior of individual nodes in large-scale distributed information filtering systems. The work presented here is based on our system MAPS (Minerva Approximate Publish/Subscribe), a novel approach to support approximate information filtering functionality in a peer-to-peer environment. In MAPS, a user subscribes to and monitors only carefully selected publisher nodes, and receives notifications from these information sources only. In this way, document-granularity dissemination is known from exact information filtering approaches is avoided, and the system is able to support very high publication rates. However this scalability benefits come at the cost of lower recall. To improve node selection and thus recall, in previous work we have proposed a ranking method that predicts nodes' publishing behavior based on time-series analysis techniques. In this work, we focus on the prediction parameters used by these techniques and show how their choice can affect the overall recall achieved. Additionally, we demonstrate how these parameters can be computed in a per node fashion, using only local information available at each node. Our method incurs no extra message traffic in the network and experiments show improvements in both prediction error and recall. When compared to an oracle opponent and an opponent with global information, our method performs almost as good, by resorting only to local node computations.

1. INTRODUCTION

Today, the Web provides enormous amounts of information to humans but smart techniques are required to stay informed and to handle the information avalanche. *Information retrieval* (IR) or one-time querying is used to search for content matching an one-time query issued by the user. This dynamic setting with new content becoming available in a continuous manner requires coping with scenarios that include subscribing to information and waiting to be notified for future content that matches this subscription. *Information filtering (IF)*, also referred to as *publish/subscribe* or *continuous querying* or *information push*, complements one-time searching, since users are able to subscribe to information sources and be notified whenever new documents of interest are published. This need for push technologies is also stressed by the deployment of new tools such as Google Alert or the QSR system [20].

In an information filtering scenario, a user issues a *subscription* (or *continuous query*, or *profile*) to the system, and receives *notifications* whenever certain events of interest take place (e.g., when a paper on P2P systems becomes available)[1].

In this paper we focus on our system MAPS (Minerva Approximate Publish/Subscribe) [3, 17] and show how to improve recall by tailoring prediction parameters in a per-node fashion. In MAPS, a user subscribes with a continuous query and monitors the most interesting information sources on the network. Only these selected sources store the user's query, and as a consequence, only documents published from these sources are forwarded to the user. Thus, the system is responsible for managing the users' queries, discovering new potential sources and sending queries to better or newly available sources. Contrary to most pub/sub approaches taken so far [13, 15], MAPS relaxes the assumption of potentially delivering notifications from every information producer and only monitors selected sources that are likely to publish documents relevant to the user's interests in the future.

1.1 Contribution

In MAPS the double exponential smoothing (DES) technique is used to predict the future publishing behavior of publisher nodes, and this prediction is used to calculate peer scores that rank peers according to the likelihood to publish in the future documents of interest to the user. Besides this property, DES assigns exponentially decreasing weights to older values and is able to recognize trends in a series of values. To achieve this DES utilises two parameters, η and γ, that are used to tune the prediction formula to follow a more aggressive or passive prediction of values. In this work, we investigate the most appropriate parameter choices for different value behaviors and argue that one global combination for η and γ is not sufficient to recognize individual publishing behaviors, since it leads to significantly lower recall. For this reason we

*This work was supported in part by the European Commission project Evergrow (6th Framework Programme IST/FET).

[1]The terms query, subscription, profile, and continuous query will be used interchangeably.

introduce our *selective method* to compute the best parameter setting per node. This results in the reduction of prediction error and significant recall improvements. Our method is scalable, since it does not incur additional communication costs between nodes, but rather utilises information that is available locally at each node. In our experimental evaluation we compare our method against two opponents: (i) an oracle that always predicts the monitored values accurately and (ii) an opponent selecting parameter combinations that require a global view of the network, which is highly inefficient for large-scale approaches. Our selective approach manages to perform almost as good as the oracle and centralised opponents, but without incurring any extra communication cost.

1.2 Outline

The rest of paper is organized as follows. Section 2 positions our work with respect to related work. The MAPS architecture and protocols to achieve approximate publish/subscribe are introduced in Section 3. Section 4 presents our node selection method, while Section 5 presents the double exponential smoothing prediction technique. Our experimental evaluation is presented in Section 6. Finally Section 7 concludes the paper.

2. RELATED WORK

P2P information filtering is a new field that has grown out of the efforts and results of various communities including information retrieval, databases, and distributed systems. In this section we review early work in the area of information filtering, present some P2P IF approaches, and compare our approximate filtering approach with the existing exact filtering systems in P2P systems.

First approaches to pub/sub by IR researchers focused mainly on appropriate representations of user interests [10] and on improving filtering effectiveness [8]. In [1] the authors address performance, aiming at scaling up to large numbers of filtering tasks in a centralized environment. They assume a server that receives documents at a high rate, and propose algorithms that support vector space queries by improving the algorithm SQI of [18]. InRoute [4] was another influential system based on inference networks with emphasis on filtering efficiency. There, documents and query networks are created, and belief propagation techniques are used to filter incoming documents. Other works in the area mainly focus on adaptive filtering [5, 21] and how dissemination thresholds of vector space queries can be adapted based on past documents.

New approaches to information filtering that use a DHT as the routing infrastructure to build filtering functionality for IR-based models and languages have recently been introduced with the proliferation of P2P systems. DHTrie [15] is a system that extends the Chord protocol to achieve exact information filtering functionality. It uses the Chord DHT [12] to create a global query index, and applies document-granularity dissemination to achieve the recall of a centralized system at low message costs. In the same spirit, LibraRing [14] presents a framework to provide information retrieval and filtering services for future digital libraries in a super-peer environment. Similarly, pFilter [13] uses a hierarchical extension of the CAN DHT [11] to store user queries and relies on multi-cast trees to notify subscribers. Again the DHT serves as a distributed index, and documents are multi-casted to reach stored subscriptions.

All the distributed information filtering systems described above involve some sort of node selection techniques to decide where to index a user query. This selection is critical, since future publications that may match this query should also be able to reach the node storing it to trigger a notification in case of a match. Query placement, as implemented in exact information filtering approaches such as [13, 15] is deterministic, and depends upon the terms contained in the query and the hash function provided by the DHT. These query placement protocols lead to filtering effectiveness that is exactly the same as that of a centralized system. Compared to a centralized approach, [13, 15] exhibit scalability, fault tolerance, and load balancing at the expense of high message traffic at publication time.

In our approach, only the most specialized and promising nodes store a user query and are thus monitored. Publications produced by each node are only matched against its local query database since, for scalability reasons, no publication forwarding is used. Thus, in the case of approximate filtering, the recall achieved is lower than that of exact filtering, but document-granularity dissemination to the network is avoided. This improves scalability and system efficiency since, in a typical pub/sub setting, the rate of publications is expected to be high. In the case of exact matching, the network cost (and thus system performance) is directly dependent on this rate, whereas in our approach it only triggers more local node computations. An interesting application scenario for approximate information retrieval and filtering is presented in [22], where the proposed MinervaDL architecture unifies both functionalities in a digital library setting.

3. THE MAPS ARCHITECTURE

We present the system architecture of MAPS [17, 3] based on the P2P search engine Minerva [9]. The main architectural components of MAPS are shown in Figure 1. Each node in the MAPS network implements the *directory service* and optionally one or two of the other services, i.e., the *publication service* and the *subscription service*.

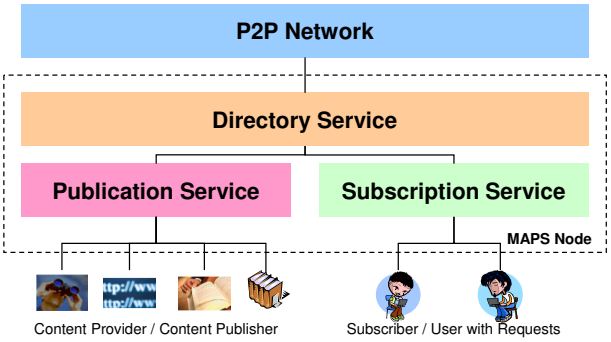

Figure 1: MAPS architecture with directory, subscription, and publication service.

The directory service is used to enable the node to participate in the distributed P2P network and maintains IR statistics in a distributed manner. A node implementing the publication service, e.g., a digital library or some other content provider, is able to make new content available to the rest of the network. Finally, using the subscription service, nodes send continuous queries to the MAPS network. Further, the subscription service service is also responsible to select the most appropriate information sources to receive the continuous query.

3.1 Directory Service

All nodes participating in MAPS have to implement the directory service. This service provides a DHT-based routing infrastructure and is responsible for the maintenance of a distributed index storing statistics for all terms in the network. This index forms a conceptually global, but physically distributed directory, which is layered

on top of a Chord-style DHT, and manages aggregated information about each node's local knowledge in compact form as described in the Minerva search engine architecture [2]. The DHT partitions the term space, such that every node is responsible for the statistics of a randomized subset of terms within the global directory. To keep IR statistics up-to-date, each node distributes per-term summaries of its local index along with contact information to the global directory. For efficiency reasons, these messages can be piggy-backed to DHT maintenance messages and batching is used.

Notice that the directory service does not necessarily have to use Chord or any other DHT; our architecture allows for the usage of any type of P2P network (structured or unstructured), given that the necessary information (i.e., the per-node IR statistics) is made available through appropriate protocols to the rest of the services.

3.2 Subscription Service

Nodes that want to monitor specific information producers implement the subscription service of MAPS which is critical to the recall that will be achieved at filtering time, since it is responsible for selecting the appropriate nodes that will index the query. The node selection procedure uses the MAPS directory service to discover and retrieve node statistics that will guide query indexing. Once these statistics are retrieved, a ranking of the potential sources is performed and the user query is sent to the *top-k* ranked publisher nodes. Since only these publishers will be monitored for new publications, the subscriber will only get notifications from the selected nodes. Query repositioning is necessary to achieve higher recall because the publication behavior of nodes may not be consistent over time.

Publications and subscriptions could be expressed using any appropriate IR model (e.g., Boolean, VSM or LSI). For simplicity, we assume that published documents and subscriptions are sets of words, and we use the Boolean model to decide when a document matches a subscription.

We assume that a subscriber node wants to subscribe with a *multi-term query* with k distinct query terms. To determine which nodes in the network are promising candidates to satisfy the continuous query with appropriate documents published in the future, the subscriber node collects appropriate statistics from the directory. To do so, the subscriber contacts all directory nodes responsible for the query terms requesting the appropriate statistics. This way, the querying node collects all node lists for the query terms and utilizes a scoring function to compute a node score for each publisher node with respect to the requested continuous query. The scoring function will be described in Section 4. Based on the score calculated for each node, a ranking of nodes is determined and the highest ranked nodes are candidates for storing the query. The querying node sends the query to these most promising publishers which store the continuous query using a local query indexing mechanism such as BestFitTrie [16] or SQI [19].

Notice that only publications occurring *in those nodes* will be matched against the query and create appropriate notifications. All nodes publishing relevant documents, but not indexing the query, will not produce a notification, simply because they are not aware of the information demand. Since only selected nodes are monitored for publications, the node ranking function becomes a critical component, which will determine the final recall achieved.

Filtering and node selection are dynamic processes, therefore periodic query repositioning is necessary to adapt to changes in publisher behavior. The subscriber re-executes the subscription protocol, acquires new node statistics, computes a new ranking, and modifies the set of nodes indexing the query.

3.3 Publication Service

In MAPS, the publication service can be used by nodes that want to expose their content to the network. A publisher node utilizes the directory service to update statistics about the terms contained in the documents it publishes. Besides this, the nodes are also responsible for storing continuous queries submitted by subscribers and matching them against their own publications. All continuous queries that match a publication produce appropriate notifications to interested subscribers.

Whenever a document is published by a publisher node, it is matched against the local query database to determine which subscribers should be notified. Then, for each one of these nodes, the publisher constructs a notification message and sends it to the subscriber.

4. NODE SELECTION

To select which publisher nodes should be monitored, the subscription service of MAPS described in Section 3.2 utilises a scoring function to rank publisher nodes. In our approach the querying node computes a node score based on a combination of *resource selection* and *node behavior prediction* formulas as shown by the equation below:

$$score(P, q) = \alpha \cdot sel(P, q) + (1 - \alpha) \cdot pred(P, q)$$

In this equation, q is a query, P is a publisher node, and $sel(P, q)$ and $pred(P, q)$ are scoring functions based on resource selection and prediction methods respectively that assign a score to a node P with respect to a query q. The scoring function $score(P, q)$ decides the final score of a node P with respect to q. The tunable parameter α affects the balance between authorities (high $sel(P, q)$ scores) and nodes with potential to publish matching documents in the future (high $pred(P, q)$ scores). Based on these scores calculated for each node, a ranking of nodes is determined, and q is forwarded to the highest ranked nodes.

4.1 Resource Selection

The function $sel(P, q)$ is calculated using standard resource selection algorithms from the IR literature (such as simple tf-idf based methods, CORI [6], language models, etc.). Using $sel(P, q)$ we can identify authorities specialized in a topic. In this paper we disregard resource selection in the node selection procedure. In [3, 22, 17] we have shown that resource selection can improve node selection in several publishing scenarios but the focus of this work is to improve node behavior prediction such that we only consider appropriate scenarios where node behavior prediction dominates the scoring function, e.g., nodes only publish documents corresponding to their main interest but with different publishing rates. Disregarding resource selection means that we use $\alpha = 0.0$ in our basic scoring function above.

4.2 Node Behavior Prediction

The function $pred(P, q)$ returns a score for a publisher node that represents its likelihood of publishing documents that are relevant to query q in the future. For all query terms, we predict the number of documents a publisher node will publish that contain the term. Therefore, MAPS predicts the value for the document frequency (denoted as $df^*_{P,t}$), and uses the difference (denoted as $\delta(df^*_{P,t})$) between this predicted and the last received value obtained from the directory to calculate the score for P (the symbol δ signifies difference). Value $\delta(df^*_{P,t})$ reflects the number of relevant documents that P will publish in the next time-unit. All statistical input

needed for our prediction mechanism is obtained by the querying node by regularly retrieving the underlying values from the directory. The predicted behavior for node P can now be quantified as follows:

$$pred(P,q) = \sum_{t \in q} \log\left(\delta(df^*_{P,t}) + 1\right)$$

The addition of 1 in the log formula is used to yield positive predictions and to avoid $\log(0)$.

5. DOUBLE EXPONENTIAL SMOOTHING

In this section we introduce our prediction technique that is based on *double exponential smoothing*. Initially we give a short introduction to time-series analysis and explain why double exponential smoothing is our technique of choice. After that, we investigate this approach in detail by considering different value behaviors. Further we introduce our *selective method* to automatically adapt the double exponential smoothing technique to the observed data series. We present an algorithm for our approach that does not need any additional communication cost.

5.1 Time-Series Analysis

We consider time-series of IR statistics to predict node behavior, thus making a rich repository of techniques from time-series analysis [7] applicable to our problem. These techniques are used to predict future time-series values based on past observations and all the techniques differ in their assumptions about the internal structure of the time-series (e.g., whether it exhibits a trend or seasonality).

Next, we give a short overview of the most important techniques that we considered in this area, and explain why we chose *double exponential smoothing* for our setting. A more detailed overview can be found in [7]. In the following, let the values $x_1, \ldots, x_{n-1}$ denote the observed time-series values and let x^*_n be the predicted value.

5.1.1 Moving Average Techniques

Moving averages are a prominent group of time-series prediction techniques. Single moving average is the simplest technique that uses the mean of the k most recent observations to predict the next time-series value, i.e.,

$$x^*_n = (x_{n-k}, \ldots, x_{n-1})/k \ .$$

Two general objections about moving average techniques are that they cannot cope well with trends observed in the data and assign equal weights to past observations. Both weaknesses are critical in our scenario. The considered IR statistics exhibit trends, for instance, when nodes gradually change their thematic focus. In our setting, it is also reasonable to put emphasis on a node's recent behavior and thus to assign higher weight to recent observations.

5.1.2 Exponential Smoothing Techniques

The second group of prediction techniques considered here, address both issues. *Single exponential smoothing* is similar to the moving average technique presented above but takes into account all past observations (in contrast to only k) with exponentially decaying weights. The smoothed value S_n that is used as a predictor for x^*_n is recursively defined as

$$S_n = \eta \cdot x_{n-1} + (1-\eta) \cdot S_{n-1} \ .$$

The parameters η is used to control the speed at which the older observations are dampened. When η is close to 1, dampening is quick and when η is close to 0, dampening is slow.

Similar to the moving average techniques, single exponential smoothing cannot cope with trends in the observed data. Therefore, we use *double exponential smoothing* that eliminates this weakness by taking into account trends in the observed data. This technique maintains two smoothed values L_n and T_n representing the *level* and the *trend* respectively. A predictor for the next time-series value is obtained as follows:

$$\begin{aligned} x^*_n &= L_n + T_n \\ L_n &= \eta \cdot x_{n-1} + (1-\eta) \cdot (L_{n-1} + T_{n-1}) \\ T_n &= \gamma \cdot (L_n - L_{n-1}) + (1-\gamma) \cdot T_{n-1} \end{aligned}$$

The parameter γ is introduced to dampen the effect of trend over time, similar to η. For completeness, we mention that there is also triple exponential smoothing that, in addition, handles seasonality in the observed data. We argue that many many queries are expected to be short-lived such that no seasonality will be observed in the IR statistics time-series. For an application with many long-lasting queries, triple-exponential smoothing could be used to take seasonality also into account.

5.2 Analysing Different Behaviors

To get a better understanding of the exponential smoothing techniques, we investigate how the double exponential smoothing is able to predict the correct future values. For this, we assume eight different data series, each simulating a different publishing behavior. Table 1 shows all behaviors. All values range from 0 to 600, and the series length is 10. Table 1 also shows how the i-th value of the series is computed.

LOG_INC	$log(i) * (600/log(10))$ $0, 180, 286, ..., 541, 572, 600$
LOG_DEC	$log(10-i+1) * (600/log(10))$ $600, 572, 541, ..., 286, 180, 0$
LIN_INC	$(600/10) * i$ $60, 120, 180, ..., 480, 540, 600$
LIN_DEC	$(600/10) * (10-i+1)$ $600, 540, 480, ..., 180, 120, 60$
QUAD_INC	$(i^2) * (6))$ $6, 24, 54, ..., 384, 486, 600$
QUAD_DEC	$((10-i+1)^2) * (6))$ $600, 486, 384, ..., 54, 24, 6$
EXP_INC	$600/(10-i+1)$ $60, 66, 75, ..., 200, 300, 600$
EXP_DEC	$600/i$ $600, 300, 200, ..., 75, 66, 60$

Table 1: Different Data Series.

Now, we investigate the influence of the two double exponential smoothing parameters η and γ by looking at all possible combinations from 0 to 1 in steps of 0.1 such that there are 121 different parameter combinations. For each combination, we use the first four data values of the different behaviors as bootstrapping values, and we compute the average prediction error per round between the real data value and the predicted value. Figure 2 shows for each behavior the corresponding prediction errors where the chart is ordered by the average prediction value per parameter combination in descending order. Consider that parameter combinations are not in numerical order.

As a result, Figure 2 shows that there is a high variation of prediction errors depending on the choice of the parameter com-

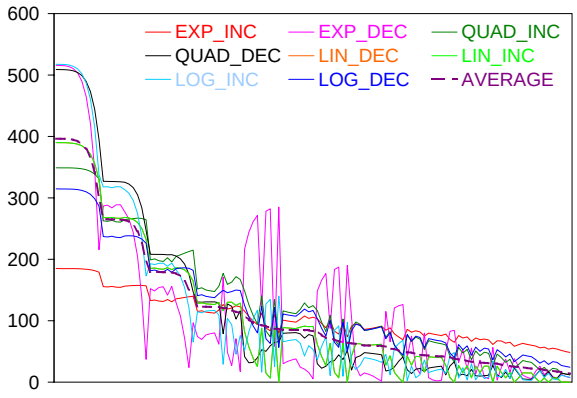

Figure 2: Prediction errors (y-axis) with double exponential smoothing using for different data series with all parameter combination (x-axis).

bination of η and γ. For different value pairs, double exponential smoothing presents various best fitting settings but there is no ideal parameter combination. The best combination for one behavior does not necessarily result in satisfying predictions for another one. We conclude that a *global choice* for η and γ that is used to predict all node behaviors for all active continuous queries is not ideal. We have to adapt the parameter combination to the observed value behavior and this will lead to our selective method we will explain next.

5.3 The Selective Method

In the previous section we have seen that there exists no single setting for parameters η and γ to effectively model all different publishing behaviors. Therefore, we introduce our selective method to adapt the parameter setting to the given scenario. Our approach works as follows:

Let the values $x_1, \ldots, x_{n-1}$ denote the observed time-series values and let x_n^* be the predicted value. The selective method uses the values $x_1, \ldots, x_{n-2}$ to predict the already known last observed value x_{n-1}. Let $x^*_{n-1,\eta,\gamma}$ denote the predicted value for all combinations of η and γ. Now, we select the parameter combination with the smallest error concerning the real observed value of x_{n-1}. If there are more than one combination with smallest error, we pick the one with the smallest distance to $\eta = 0.5$ and $\gamma = 0.5$. The selected parameters are used to predict the next future value x_n^*.

The algorithm 1 explains the selective method with input values $\{x_1, \ldots, x_{n-1}\}$ and the function $DES_{\eta,\gamma}$ to predict the next value. The algorithm outputs the selected parameter values for η and γ, as well as the prediction value x_n^* as the result of $DES_{\eta,\gamma}$. Notice that we have simplified the algorithm such that the case that several parameter combinations have the smallest error is not considered. Moreover, the observed time-series needs at least two values. If we have only observed one or no value, we can not apply algorithm 1.

The selective method means that we always use the most appropriate parameter setting concerning the last observed value. In this way, we adapt double exponential smoothing better to the given data series. Obviously, we need at least four observed values to apply this method because three values are indispensable to properly predict the last observed value. Next, to analyse our approach, we implemented the selective method and computed the prediction errors for the various behaviors in Section 5.2.

Figure 3 compares the selective method with the minimum pre-

Algorithm 1 Selective Method

input: value $\{x_1, \ldots, x_{n-1}\}$, function $DES_{\eta,\gamma}$
output: parameter η, γ and prediction value x_n^*
$error_{min} := \infty$
for $e = 0.0$ to 1.0 **do**
  for $g = 0.0$ to 1.0 **do**
    $x^*_{n-1,\eta,\gamma} := DES_{e,g}(\{x_1, \ldots, x_{n-2}\})$
    $error := \left|x^*_{n-1,\eta,\gamma} - x_{n-1}\right|$
    if $error \leq error_{min}$ **then**
      $error_{min} := error$
      $\eta := e$
      $\gamma := g$
    end if
  end for
end for
$x_n^* := DES_{\eta,\gamma}(\{x_1, \ldots, x_{n-1}\})$

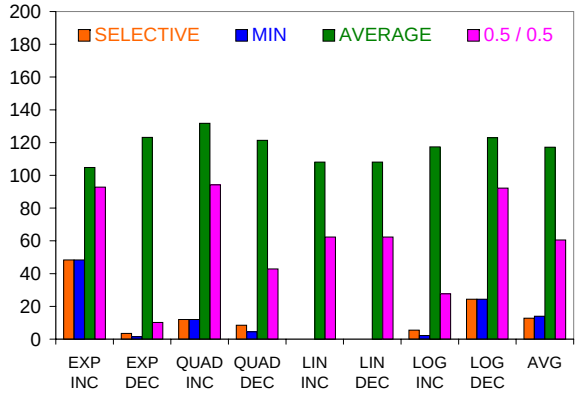

Figure 3: Comparing our selective method with a global parameter setting considering different value behaviors.

diction error per round (MIN), the average prediction error (average), and the prediction errors for parameter combination $\eta = 0.5$ and $\gamma = 0.5$ (*0.5 / 0.5*). Since the maximum prediction error is very high it is not plotted in the graph to allow for the better illustration of the other methods.

As we can see, the selective method is in all behaviors almost as good as the MIN that means that our approach selects the appropriate parameters. The combination *0.5 / 0.5* does not really result satisfying error rates although this could be an obvious choice of parameters. As expected, AVERAGE even yields to prediction errors worst than *0.5 / 0.5*. Looking at the different behaviors, the selective method accurately predicts the real values for the *LIN_INC* and *LIN_DEC* behaviors.

In addition, Figure 3 shows the average over all eight behaviors. Here, our selective method performs even better than MIN. This means, the average prediction error over all behaviors using our selective method is lower than the best global combination for all behaviors. This is caused by the fact that for different behaviors other parameter combinations yield to the best possible prediction. If we combine the errors of the selective method, we almost reach the individual best combination and this is still better than the global MIN. In the next Section, we will evaluate our selective method in terms of usability in the MAPS system.

5.4 Alternative Approach

Here, we shortly describe an alternative approach to improve the

prediction of double exponential smoothing. Usually, to select the most appropriate values for the parameters η and γ, each node would investigate the parameter influence using a test collection with a centralized computation, and this training phase would estimate the parameters. Obviously, each node would have one fixed combination for all publisher nodes. In this case, MAPS would not be able to recognize the individual publishing behaviors of nodes. Therefore, this training approach helps selecting a local parameter combination for each querying node but is not as flexible as the selective method which is able to individually recognize the publishing behavior of every node that is a candidate to monitor.

All alternative ways to select parameter values for η and γ suffer from the the problem of selecting the appropriate training set. Even if we consider distributed approaches, the wrong choice of text collections can result in unsatisfying results.

6. EXPERIMENTAL EVALUATION

This section evaluates our selective method implemented for in the Minerva Approximate Publish/Subscribe system. We will explain the experimental setup, measurements, and data and we will present the experimental results using different node publishing scenarios.

6.1 Setup

In all experiments the following steps are executed. The network is set up and the underlying DHT is initiated. Next, we have four *bootstrapping* rounds where the subscriber nodes collect the posted directory statistics to the requested queries. After this bootstrapping phase, in the six subsequent rounds the querying nodes subscribe to selected publishers, to reach a total of ten rounds. During these six rounds that represent the monitoring phase we will say that a publisher node is *monitored* by a subscriber, when the subscriber's continuous query is stored in the publishers local query database. The monitored nodes notify the subscriber for all publications matching the stored continuous query.

All nodes in the network publish documents during both phases and at certain intervals (or rounds), the continuous queries are repositioned. The intervals and the number of published documents per round depend on the individual node behavior that we will inspect in Section 6.3. At the end of each round of the subscription phase, the subscriber nodes rank publishers using the formula described in Section 4 and reposition their queries accordingly.

6.2 Measurements

To evaluate the retrieval effectiveness, we use *recall* as the ratio of the total number of notifications received by subscribers to the total number of published documents matching subscriptions. We are interested in the average recall over all rounds of the subscription phase.

Similar to the publishing behaviors shown previously in the analysis of double exponential smoothing, we compare the recall of our selective method with the minimum and maximum recall that would be possible if we used a single pair of parameter values for η and γ for the all the nodes in the network. In addition we use an *oracle* node selection approach (referred to as ORACLE) such that it always predicts the accurate IR statistics. The recall of ORACLE represents the highest possible recall that could be achieved by MAPS. The random node selection approach is implemented only for comparison purposes and demonstrates the performance of a baseline approach.

Besides recall, we also analyse the *prediction quality*. We measure the average prediction error per term, node and round. In the graphs our method is compared only to the minimum prediction error opponent (MIN) since the maximum prediction error (MAX) is very high. Notice that, similarly to the case of recall measurements, the MIN and MAX opponents refer to globally selecting the set of parameters that would minimize or maximize the prediction error. In Section 6.3 we investigate the different node publishing behaviors in terms of recall and average prediction error. The two other approaches are not considered because *random* does not utilise a prediction-based node selection and ORACLE has by definition a prediction error of zero.

6.3 Data

The document collection that was used for the experiments contains more than 2 million documents from a focused Web crawl. All documents are categorised in one of ten categories: *Music*, *Finance*, *Arts*, *Sports*, *Natural Science*, *Health*, *Movies*, *Travel*, *Politics*, and *Nature*. The category size ranges from 68,000 to more than 325,000 documents. There are more than 500,000 different terms (stop words are not considered) and we use the documents from different categories to categorize our node set. In all experiments, the network consists of 100 nodes with 10 nodes per category.

Using the document collection, we extracted seven strong representative single-term queries: *music*, *arts*, *sports*, *travel*, *hotel*, *offer*, *city*. Single-term queries have the advantage that there is a direct dependency between correctly predicting values and recall. In the case of multi-term queries, we can have the effect that a node publishes a lot of documents containing the single terms but only a few containing the whole query term set. For simplicity, we have only considered single-term queries in this experimental setting. Considering multi-term queries would result in interfering with the prediction parameters and would not let us isolate and study individual publishing behaviors. There are approaches in the literature [9] on how to overcome this restriction in a search environment, and in future work we plan to adapt these approaches to our setting.

6.4 Results

After explaining the setup and the dataset of our experimental evaluation, we investigate the recall and prediction error results for different node behaviors. In the set of experiments shown in this section we utilise to the publishing behaviors listed in 1. This means that a node following the *LOG_INC* behavior publishes in the first round no documents and in the last of the ten rounds 600 documents. In addition, we consider a constant publishing behavior where a node constantly publishes 300 documents per round during both publishing phases.

To investigate the effectiveness of our selective method, we analyze three different scenarios. In the first scenario, all mentioned behaviors are used such that some nodes have a constant publishing behavior and others increase or decrease their publication accordingly. The second scenario looks at the results when all nodes in the network have an *EXP_INC* behavior and in the last scenario all nodes follow a *QUAD_DEC* publishing behavior.

The graphs for all scenarios illustrate the performance of the different opponents. The ORACLE opponent shows the maximum recall MAPS can reach by accurately predicting the IR statistics. Naturally, the prediction error for the ORACLE opponent is zero. The random node selection shows the results when publisher nodes are selected completely at random. In the random setting, the prediction error is not of interest, because prediction is completely ignored. The MIN and MAX opponents present the best and worst recall MAPS can get with a global setting of parameters across all nodes. Similarly to Section 5.3, in the graphs we only consider the MIN prediction error to better illustrate the differences between

the different opponents. The selective method shows the recall and prediction error of our local parameter computation that is used to adapt the double exponential smoothing parameters.

6.4.1 *Mixed Publishing Scenario*

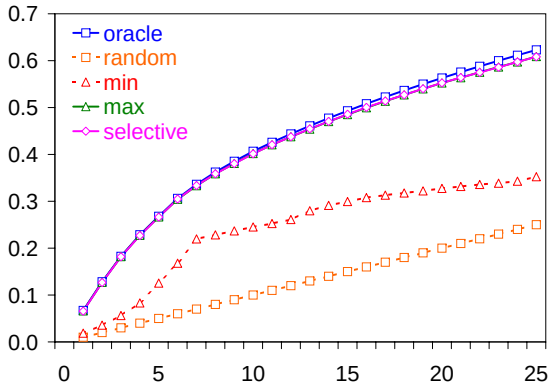

Figure 4: Recall depending on the number of monitored nodes in the mixed publishing scenario.

Over the ten rounds of this scenario, all 100 nodes publish together 285,960 documents following the different publishing behaviors. Figure 4 shows the recall depending on the percentage of monitored nodes in the network. As it can be seen, the selective method performs marginally better than the MAX opponent and slightly worse than the ORACLE, whereas MIN and random achieve significantly lower recall. This means that the correct choice for the parameters can greatly affect the recall level. Notice that in this scenario the selective method we propose not only performs slightly better than the best global parameter choice for η and γ, but also this performance is not greatly affected by the percentage of monitored publishers. This shows that a local auto-adjustment of prediction parameters is possible and results in recall similar to approaches that require global knowledge to compute these parameters. Thus, our method is able to achieve recall of 60% by monitoring only 20% of all publisher nodes in the network.

Additionally, Figure 5 shows the prediction error per term, node, and round. The prediction error of our selective method is about as good as the minimum prediction error computed using a global parameter setting. The error varies from 13 to 20 on average.

6.4.2 *EXP_INC Publishing Scenario*

Figures 6 and 7 show the results of our second scenario where all 100 nodes follow a *EXP_INC* publishing behavior. The total number of published documents amounts to 175,600. In this scenario, recall is almost independent of the parameter choice. All approaches (MIN, MAX, ORACLE, and selective) perform almost the same. Even the prediction quality of the selective method and MIN are similar and vary from about 22 to 35 per term, round, and node. The prediction error for MAX is still very high and not included in Figure 7, but there is no influence to the recall. This is caused by the fact that all nodes follow the same behavior.

6.4.3 *QUAD_DEC Publishing Scenario*

The last scenario considers the case that all nodes publish documents using a *QUAD_DEC* behavior. The overall number of published documents is 231,000. Here, our selective method performs much better than MIN in terms of recall but does not reach the level of MAX. This result corresponds to the observations of Section 5.3.

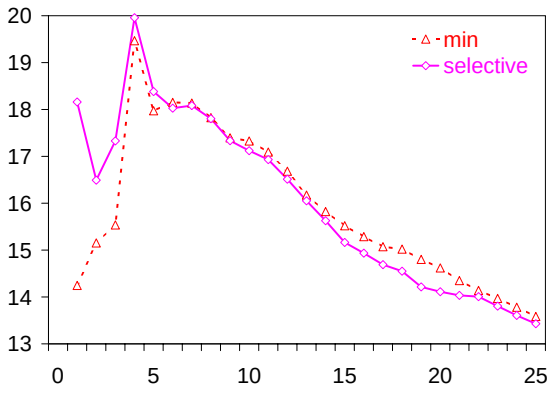

Figure 5: Average prediction error of MIN and the selective method per term, node, and round in the mixed publishing scenario.

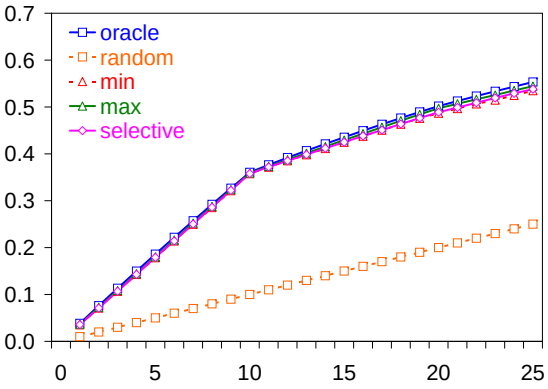

Figure 6: Recall depending on the number of monitored nodes in the *EXP_INC* publishing scenario.

The same result holds for the prediction error where the selective method causes higher errors than the best global parameter choice. Of course, the selective method still performs much better than the worst parameter choice for η and γ.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented the system architecture of MAPS and a new node selection technique based on time-series analysis to support approximate IF in a P2P environment. Our main focus was to improve the prediction process of IR statistics. We explained how double exponential smoothing can be effectively used as a prediction technique, and investigated the influence of the parameter setting in recall. We introduced our *selective method* that depends on past observations and selects an appropriate parameter setting per node by imposing no extra cost in the network.

Our experimental evaluation demonstrated efficiency and scalability of our approach in many diverse scenarios, and it showed that our selective method is a good approach to individually adapt our prediction technique to the per-node publishing behaviors. In some scenarios, the selective method even outperforms the best possible choice for a global parameter setting such that recall quality of MAPS improves. Our selective method does not need any additional communication over the network such that only local computations are necessary.

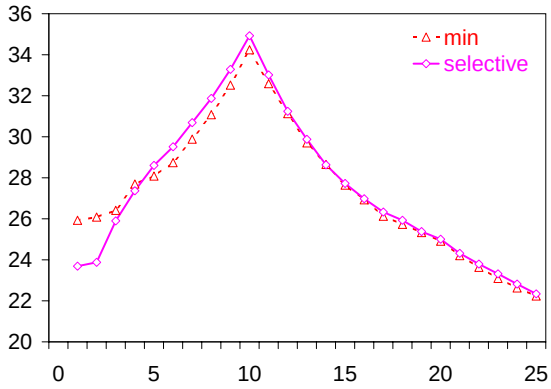

Figure 7: Average prediction error of MIN and the selective method per term, node, and round in the *EXP_INC* publishing scenario.

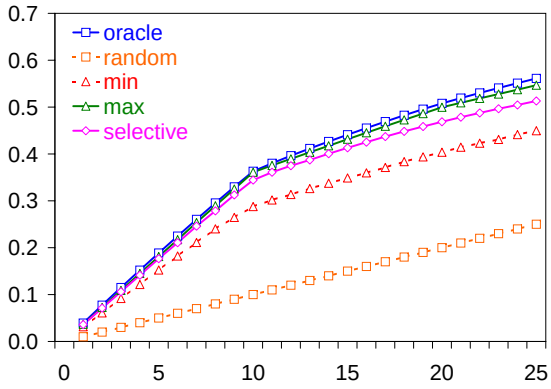

Figure 8: Recall depending on the number of monitored nodes in the *QUAD_DEC* publishing scenario.

As future work we plan to consider term correlations as in [9] in order to extend the statistics stored in the directory and improve result quality. Other interesting directions involve investigating the query repositioning strategy to reduce message costs in directory updates.

8. REFERENCES

[1] T. Bell and A. Moffat. The Design of a High Performance Information Filtering System. In *SIGIR*, 1996.

[2] M. Bender, S. Michel, P. Triantafillou, G. Weikum, and C. Zimmer. Improving Collection Selection with Overlap-Awareness. In *SIGIR*, 2005.

[3] K. Berberich, M. Koubarakis, C. Tryfonopoulos, G. Weikum, and C. Zimmer. MAPS: Approximate Publish/Subscribe Functionality in Peer-to-Peer Networks. In *ADPUC*, 2006.

[4] J. Callan. Document Filtering With Inference Networks. In *SIGIR*, 1996.

[5] J. Callan. Learning While Filtering Focuments. In *SIGIR*, 1998.

[6] J. P. Callan, Z. Lu, and W. B. Croft. Searching Distributed Collections with Inference Networks . In *SIGIR*, 1995.

[7] C. Chatfield. *The Analysis of Time Series - An Introduction*. CRC Press, 2004.

[8] D. Hull, J. Pedersen, and H. Schütze. Method Combination For Document Filtering. In *SIGIR*, 1996.

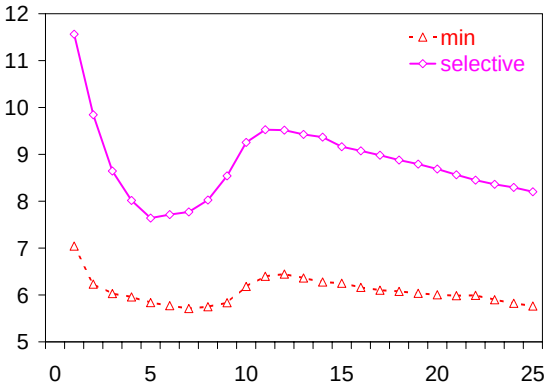

Figure 9: Average prediction error of MIN and the selective method per term, node, and round in the *QUAD_DEC* publishing scenario.

[9] S. Michel, M. Bender, N. Ntarmos, P. Triantafillou, G. Weikum, and C. Zimmer. Discovering and Exploiting Keyword and Attribute-Value Co-Occurrences to Improve P2P Routing Indices. In *CIKM*, 2006.

[10] M. Morita and Y. Shinoda. Information Filtering Based on User Behaviour Analysis and Best Match Text Retrieval. In *SIGIR*, 1994.

[11] S. Ratnasamy, P. Francis, M. Handley, R. M.Karp, and S. Shenker. A Scalable Content-Addressable Network. In *SIGCOMM*, 2001.

[12] I. Stoica, R. Morris, D. R. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In *SIGCOMM*, 2001.

[13] C. Tang and Z. Xu. pFilter: Global Information Filtering and Dissemination Using Structured Overlay Networks. In *FTDCS*, 2003.

[14] C. Tryfonopoulos, S. Idreos, and M. Koubarakis. LibraRing: An Architecture for Distributed Digital Libraries Based on DHTs. In *ECDL*, 2005.

[15] C. Tryfonopoulos, S. Idreos, and M. Koubarakis. Publish/Subscribe Functionality in IR Environments Using Structured Overlay Networks. In *SIGIR*, 2005.

[16] C. Tryfonopoulos, M. Koubarakis, and Y. Drougas. Filtering Algorithms for Information Retrieval Models with named Attributes and Proximity Operators. In *SIGIR*, 2004.

[17] C. Tryfonopoulos, C. Zimmer, M. Koubarakis, and G. Weikum. Architectural Alternatives for Information Filtering in Structured Overlay Networks. *IEEE Internet Computing*, 2007.

[18] T. Yan and H. Garcia-Molina. Index Structures for Information Filtering under the Vector Space Model. *ICDE*, 1994.

[19] T. Yan and H. Garcia-Molina. The SIFT Information Dissemination System. *TODS*, 1999.

[20] B. Yang and G. Jeh. Retroactive Answering of Search Queries. In *WWW*, 2006.

[21] Y. Zhang and J. Callan. Maximum Likelihood Estimation for Filtering Thresholds. In *SIGIR*, 2001.

[22] C. Zimmer, C. Tryfonopoulos, and G. Weikum. MinervaDL: An Architecture for Information Retrieval and Filtering in Distributed Digital Libraries. In *ECDL*, 2007.