Assessment Algorithm for Service Matching Based on Bloom Filter

Wendong Zhao, Jin Zhang, Laixian Peng, Dawei Niu and Jingnan Nie

Institute of Communication Engineering, PLA University of Science and Technology Nanjing 210007, China

Abstract. The main service search algorithms that used on content-based publish/subscribe system don’t support service fuzzy matching effectively. This paper presents a service matching degree assessment method based on Bloom filter. Facilitated by this approach, an algorithm that supports service fuzzy matching has been proposed. The main idea of this algorithm is using Bloom filter to describe the service and request, and assessing the similitude degree of service and request by the similarity of Bloom filter vectors. Experimental and theoretical results show that this algorithm can support content-based service fuzzy matching by simple algebraic operations on Bloom filter. The evaluation accuracy rate is beyond 90%.

Keywords: Fuzzy matching; bloom filter; publish/subscribe.

1. Introduction

Content-based publish/subscribe[1] is a powerful paradigm for information dissemination from publishers to subscribers in large-scale distributed networks. A service specifies values of a set of attributes associated with it. Subscribers register their interests in service request through expressive subscriptions which specify complex filtering criteria by using a set of predicates over request attributes. Upon receiving a service description published by a publisher, the system matches the service to the request, which serve as filters, and delivers the service to the matched subscribers.

Fabret et al. proposed a content-based pub/sub scheme[2] defined as \( U = \{A_1, A_2, \ldots, A_n\} \), where \( A_i \) means an atomic attribute. Each attribute consists of a name, type and domain and can be specified by a tuple < name, type, (min, max)> . Each service is a set of attributes that belong to set \( S \) and can be represented as \( S = \{c_1, \ldots, c_n\} \) , where \( c_i \) is the domain of \( A_i \). Each request can be expressed as the logical operation on the attributes, for example, request \( Q \) can be described as \( Q = \{(A_i = v_i) \wedge (v_i < A_j < v_i)\} \) . Without lost universality, the request is defined as \( Q = \{A_i = v_i, A_i = v_i, \ldots, A_i = v_i\} \) , where \( v_i \) means the restriction of \( A_i \). If and only if \( (\forall A_j \in Q \Rightarrow A_j \in S) \wedge (\forall v_i < c_i) \) , means request \( Q \) matches service \( S \) successfully.

Main matching algorithms used in content-based publish/subscribe system include tree-based algorithms[3-4], map-based algorithms[5] and XPath-based algorithms[6-7]. The main object of these algorithms is to decrease the operate times during the matching procession, but they all have the shortcoming described below:

They all are “Accurate Matching” algorithms, that is, if the service \( S \) matches request \( Q \), every attribute that belongs to \( Q \) must belongs to \( S \). But in the real world, users usually can not describe its need accurately. For example, if the user \( B \)’s request is \( Q = \{A_1 = v_1, A_2 = v_2, A_3 = v_3\} \) , the service node \( A \) can give is \( S = \{A_1 = v_1, A_2 = v_2, A_3 = v_3\} \). Maybe \( S \) is the service that \( B \) needs or \( S \) can partly meet \( Q \), but the matching result is false.

For the algorithms can not support service fuzzy matching, according to the model Fabret has raised, this paper proposes a service matching degree assessment algorithm based on Bloom filter and based on this, we design an service fuzzy matching algorithms. This algorithm can also be used in service matching preprocess. That is, first, use this algorithm to filter almost all services that stored in local node with simple algebraic operations, then, use other algorithms to achieve accurate matching. Experimental and theoretical results show
that this algorithm can support content-based service fuzzy matching. The evaluation accuracy rate is beyond 90%.

The rest of the paper is organized as follows: section 2 gives an overview of related work. Section 3 shows basic definitions used in this paper, introduces the main idea and the algorithm. Section 4 discusses experimental setup and results. We conclude the paper in section 5.

2. Related Work on Bloom Filters

Standard Bloom filter\cite{8} can be used to support membership queries because it uses a simple space-efficient data structure to represent a set. Bloom filters are widely used in network related applications, e.g., new network architecture design, route lookup, IP packet classification and network measurement.

The standard Bloom filter is a bit vector of $M$ bits used to represent a set $S = \{s_1, s_2, s_3, \ldots, s_n\}$ of $n$ items. All bits in the vector are initially set to 0. Then, the Bloom filter uses $k$ independent hash functions to map the set to bit vector. The domain of each hash function is $1$ to $M$. For each item $s$, the bit $h_i(s)$ is set to 1. To check whether an item $s$ belongs to set $S$, we need to check whether all $h_i(s)$ are set to 1. If not, $s$ is not in the set $S$. If so, $s$ is regarded as a member of $S$ with a false positive probability, which suggests that set $S$ contains an item $s$ although it in fact does not.

Based on the standard Bloom filter, many other types of Bloom filter have been designed for specific applications, such as counting Bloom filter, compressed Bloom filter, split Bloom filter, dynamic Bloom filter.

The standard Bloom filter maps the items in the set to a vector. The vector must include some normal property about the set. Can we deduce the relation of different sets based on the Bloom filter vectors of these sets? In \cite{9}, the author designs three approaches for multi-attribute representation on network services based on parallel Bloom filter. But he does not talk about service matching based on this. In \cite{10}, the author first research the algebraic operations on Bloom filters from the point of set. In \cite{11}, the author analysis the effect on the vector when items in the set change and designs a quantitative assessment algorithm for the dynamic change of the set based on counting Bloom filter distance. Different from \cite{11}, this paper defines the similitude based on standard Bloom filter and uses the similitude to assessment the matching degree between the service and the request. Based on this, a fuzzy matching algorithm has been designed.

3. Algorithm and Analysis

3.1. Definition

In order to describe the algorithm clearly, we first show definitions as follows:

Definition 1 (Service description based on Bloom filter). For each attribute in Service description set, map the name and type of the attribute to the standard Bloom filter vector used $k$ hash function. Denote as $BF_{m,n}^k(S)$.

Definition 2 (Coverage). The Coverage between service and request is defined as $C(S,Q) = \frac{|S \cap Q|}{|Q|}$.

$|S \cap Q|$ means the total attributes with the same name and type.

Definition 3 (B-Coverage). The B-Coverage, based on Bloom filter, between service and request is defined as $C_{BF}^{k,m}(S,Q) = \frac{|BF_{m,n}^k(S) \cap BF_{m,n}^k(Q)|}{|BF_{m,n}^k(Q)|}$.

$|BF_{m,n}(Q)|$ means the sum of the bits in the vector that are set to 1. $|BF_{m,n}^k(S) \cap BF_{m,n}^k(Q)|$ means the sum of the bits in these two vectors that the same bits are all set to 1.

For example, service $S$ is $S = \{n_1, n_2, n_3, n_4\}$, it’s bloom filter description is $BF_{m,n}^k(S) = [1,0,1,0,0,0,1,1,0,0]$. Request $Q$ is $Q = \{n_1, n_2, n_3, n_4\}$, it’s bloom filter description is $BF_{m,n}^k(Q) = [1,1,1,0,0,0,0,1,0,1]$. So $C(S,Q) = \frac{|S \cap Q|}{|Q|} = \frac{3}{4}$.

$C_{BF}^{k,m}(S,Q) = \frac{|BF_{m,n}^k(S) \cap BF_{m,n}^k(Q)|}{|BF_{m,n}^k(Q)|} = \frac{4}{5}$.

3.2. Service matching algorithm
The main idea of this algorithm is using Bloom filter to describe the service and request, and assessing the similitude degree of service and request by the similarity of Bloom filter vectors.

All services that the network can support use the same Bloom filter to generate the service description and are denoted as \( S_{BF} = \{BF^{k,s}(s_1), BF^{k,s}(s_2), \ldots, BF^{k,s}(s_n)\} \). The request of the user also use the same Bloom filter to generate request description, denote as \( BF^{k,q}(q_j) \). The coverage limitation is denoted as \( Val, (0 < Val < 1) \).

Then, the pseudo code of the fuzzy matching algorithm is shown in Fig. 1:

```
for \( \forall s_k \in S \)
    if \( C_{BF}^{s,k}(s_k, q_j) \geq Val \)
        put \( s_k \) into Set V
    else
        return Set V
return NULL
```

Fig. 1: Fuzzy matching algorithm

When the node receives the request from the user, first compute the B-Coverage with each service saved in local node. If the B-Coverage satisfies the limitation, then, put the service to a set. When the computation is over, if the set is not empty, return the set, else return null.

**Theorem 1** Based on the same Bloom filter and attribute universe, for the service \( S \) and the request \( Q \), if \( C_{BF}^{s,k}(S, Q) > C_{BF}^{s,k}(S, Q) \), then \( C(S, Q) > C(S, Q) \). The \( \bar{X}(\cdot) \) stands for the statistics average value.

Proof. Suppose the service \( S \), denotes as \( S = \{A_1 = c_1, A_2 = c_2, \ldots, A_n = c_n\} \), the request \( Q \), denotes as \( Q = \{A_i = v_1, A_j = v_2, \ldots, A_n = v_n\} \), and \( |S| = n, |Q| = m, |S \cap Q| = n \), the vector length of the Bloom filter is \( m \), the total number of hash functions is \( k \). Depending on definition 1 to 3, we have:

**The Coverage between** \( S \) **and** \( Q \) **is**

\[
C = \frac{n}{n_2} \tag{1}
\]

We divide the map process into two steps. First, map the attributes belong to \( S \cap Q \), then insert the attributes belong to \( S - S \cap Q \) or \( Q - S \cap Q \) into vector \( BF_s \) or \( BF_q \). The probability that \( BF_s[i] \neq BF_q[i] \) is,

\[
P_i(BF_s[i] \neq BF_q[i]) = P_i(BF_s[i] = 0, BF_q[i] = 1) + P_i(BF_s[i] = 1, BF_q[i] = 0) = (1 - \frac{1}{m})^k \frac{1}{m} \frac{k}{n} (1 - \frac{1}{m})^{k - \frac{k}{n}} + (1 - \frac{1}{m})^k \frac{1}{m} \frac{k}{n} (1 - \frac{1}{m})^{k - \frac{k}{n}} = (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right] + (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right] = (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right] + (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right] = (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right] + (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right]
\]

The probability that \( BF_s[i] = BF_q[i] = 1 \) is,

\[
P_i(BF_s[i] = BF_q[i]) = 1 - P_i(BF_s[i] \neq BF_q[i]) - P_i(BF_s[i] = BF_q[i] = 0) = 1 - (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right] - 2 \left(1 - \frac{1}{m}\right)^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right] = (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right] + (1 - \frac{1}{m})^k \left[1 - (1 - \frac{1}{m})^{\frac{k}{n}}\right]
\]

The probability that either \( BF_s[i] = 1 \) or \( BF_q[i] = 1 \) is,
Then,

\[
P_{\text{BF}_i}[i] = 1 + P_{\text{BF}_i}[i] = 0\]

\[
= 1 - (1 - \frac{1}{m})^{n_i} = 1 - (1 - \frac{1}{m})^{n_i}
\]

Then,

\[
C_{\text{BF}}^m(S, Q) = \frac{|BF_{\text{BF}}^m(S) \cap BF_{\text{BF}}^m(Q)|}{|BF_{\text{BF}}^m(Q)|}
\]

\[
= \frac{P_{\text{BF}_i}[i] = \text{BF}_i[i] = 1}{P_{\text{BF}_i}[i] = 1}
\]

\[
= 1 + (1 - \frac{1}{m})^{n_i} - (1 - \frac{1}{m})^{n_k} - (1 - \frac{1}{m})^{n_{\text{BF}}}
\]

\[
= 1 - (1 - \frac{1}{m})^{n_i}
\]

\[
(C_{\text{BF}}^m(S, Q))^z = \frac{-\ln(1 - \frac{1}{m})}{1 - (1 - \frac{1}{m})^b} \geq 0
\]

From equation 3, we can see, if we only consider the change of n, we can get

\[
C_{\text{BF}}^m(S, Q) > C_{\text{BF}}^m(S, Q) \Rightarrow \tilde{C}(S, Q) > \tilde{C}(S, Q)
\]

Depend on theorem 1 , we can drive the relation between \( C(S, Q) \) and \( \tilde{C}(S, Q) \) from the relation between \( C_{\text{BF}}^m(S, Q) \) and \( \tilde{C}_{\text{BF}}^m(S, Q) \). But in the real engineer, we use the relation between \( C_{\text{BF}}^m(S, Q) \) and \( C_{\text{BF}}^m(S, Q) \) to deduce the relation between \( C(S, Q) \) and \( C(S, Q) \). But for the false positive of the Bloom filter, the probability that \( C(S, Q) > C(S, Q) \), but \( C_{\text{BF}}^m(S, Q) < C_{\text{BF}}^m(S, Q) \) is exist. In the next section, we conduct simulation to evaluate the accuracy of the algorithm.

4. Performance Evaluation

We use matlab to simulate and analysis the algorithm. This section focuses the simulation on three parts, (1) verify that the variation between the Coverage and B-Coverage is consistent by statistical method. (2) analysis the effect of parameters of the Bloom filter on our algorithm. (3) evaluate the accuracy of the assessment method.

4.1. Correctness analysis of the algorithm

The attributes sets of the service and request are created randomly depend on attribute universe set. Table 1 shows the simulate parameters.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes of universe</td>
<td>200</td>
</tr>
<tr>
<td>Attributes of each request</td>
<td>1~200</td>
</tr>
<tr>
<td>Attributes of each service</td>
<td>1~200</td>
</tr>
<tr>
<td>Hash functions</td>
<td>3</td>
</tr>
<tr>
<td>Length of the vector</td>
<td>1000</td>
</tr>
<tr>
<td>Simulate times</td>
<td>1000</td>
</tr>
</tbody>
</table>

We use the definition 2 to get the original Coverage, use the definition 3 to get the B-Coverage, use the formula 2 to get the theoretical average of the B-Coverage. The simulate result shows in Fig. 2. Each spot of the original Coverage and B-Coverage is the average of 1000 simulate results.

Fig. 2: Coverage simulation
From the figure 2 we can see that the statistical average of B-Coverage fully consistent with the theoretical average. At the same time, the variation of the B-Coverage is the same as the variation of the original Coverage. This means that we can deduce the matching degree between the original service and the original request by this assessing method.

We also can see from the figure 2 that the larger the difference between the service and request, the larger the difference between the B-Coverage and original Coverage. This is because the false positive of the Bloom filter.

The figure 2 also shows that for any spot it is always right that \( \bar{C}_{BF}(S, Q) > \bar{C}(S, Q) \). This is because the original attribute of the Bloom filter. For any attribute \( n \), we can deduce \( \forall n \in S \cap Q \Rightarrow BF(n) \in BF(S) \land BF(n) \in BF(Q) \), but we can not deduce \( n \in S \cap Q \) from \( BF(n) \in BF(S) \land BF(n) \in BF(Q) \).

It should be noted that we use the relativity of the B-Coverage to deduce the relativity of the original Coverage. If the relativity unchanged, the false positive of the Bloom filter will do no effect on the assessment result.

### 4.2. Effect of Bloom filter parameters

Keep the service and the request unchanged, different hash functions and different vector length will generate different Bloom filter. Next, we will simulate the effect on the assessment result when the parameters of the Bloom filter changed. Table 2 shows the simulate parameters.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes of universe</td>
<td>200</td>
</tr>
<tr>
<td>Attributes of each request</td>
<td>1~200</td>
</tr>
<tr>
<td>Attributes of each service</td>
<td>1~200</td>
</tr>
<tr>
<td>Hash functions</td>
<td>1~4</td>
</tr>
<tr>
<td>Step of the vector</td>
<td>500</td>
</tr>
<tr>
<td>Simulate times</td>
<td>1000</td>
</tr>
</tbody>
</table>

The simulate result shows in Fig. 3-4. Fig. 3 shows the changes with the number of hash functions growth and Fig. 4 shows the changes with the length of Bloom filter vector increases.

From the Fig. 3 and Fig. 4, we can see, under different simulate parameters the B-Coverage maintains the same variation with original Coverage. That is to say, from a statistical sense, given Bloom filter parameters, our algorithm would not make a mistake.

### 4.3. Analysis of the algorithm accuracy

Though theoretical analysis and simulation based on statistical shows that B-Coverage maintains the same variation with original Coverage, in real engineers, the case still exist the original Coverage between service \( A \) and request \( B \) is larger than service \( B \) and request \( B \) but the B-Coverage is on the contrary. This section simulates the accuracy of the algorithm. Table 3 shows the simulate parameters.
Table 4 shows the statistical result under different parameters.

<table>
<thead>
<tr>
<th>Vector Length</th>
<th>Hash functions</th>
<th>Times</th>
<th>Error Times</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>1</td>
<td>200000</td>
<td>966</td>
<td>0.48%</td>
</tr>
<tr>
<td>500</td>
<td>2</td>
<td>200000</td>
<td>1089</td>
<td>0.54%</td>
</tr>
<tr>
<td>500</td>
<td>3</td>
<td>200000</td>
<td>1176</td>
<td>0.59%</td>
</tr>
<tr>
<td>500</td>
<td>4</td>
<td>200000</td>
<td>1471</td>
<td>0.74%</td>
</tr>
<tr>
<td>1000</td>
<td>1</td>
<td>200000</td>
<td>1096</td>
<td>0.55%</td>
</tr>
<tr>
<td>1000</td>
<td>2</td>
<td>200000</td>
<td>615</td>
<td>0.31%</td>
</tr>
<tr>
<td>1000</td>
<td>3</td>
<td>200000</td>
<td>792</td>
<td>0.40%</td>
</tr>
<tr>
<td>1000</td>
<td>4</td>
<td>200000</td>
<td>938</td>
<td>0.47%</td>
</tr>
<tr>
<td>1500</td>
<td>1</td>
<td>200000</td>
<td>567</td>
<td>0.28%</td>
</tr>
<tr>
<td>1500</td>
<td>2</td>
<td>200000</td>
<td>284</td>
<td>0.14%</td>
</tr>
<tr>
<td>1500</td>
<td>3</td>
<td>200000</td>
<td>260</td>
<td>0.13%</td>
</tr>
<tr>
<td>1500</td>
<td>4</td>
<td>200000</td>
<td>205</td>
<td>0.10%</td>
</tr>
<tr>
<td>2000</td>
<td>1</td>
<td>200000</td>
<td>312</td>
<td>0.16%</td>
</tr>
<tr>
<td>2000</td>
<td>2</td>
<td>200000</td>
<td>440</td>
<td>0.22%</td>
</tr>
<tr>
<td>2000</td>
<td>3</td>
<td>200000</td>
<td>317</td>
<td>0.16%</td>
</tr>
<tr>
<td>2000</td>
<td>4</td>
<td>200000</td>
<td>436</td>
<td>0.22%</td>
</tr>
</tbody>
</table>

The statistical results show that the error rate of the B-Coverage based assessment algorithm is super than 99%. It means that we can deduce the relativity of the original Coverage based on the relativity of the B-Coverage in the real engineers.

5. Conclusion

Based on the standard Bloom filter, we first define the Coverage and B-Coverage, then, analysis the relation between Coverage and B-Coverage. We conclude that from the statistical point of view, the B-Coverage maintains the same variation with original Coverage and we can deduce the matching degree between the original service and the original request by their description based on standard Bloom filter.

At last, we apply these concepts to content-based publish/subscribe system. We design a fuzzy matching algorithm. This algorithm can be used alone or be used as a preprocess algorithm with other accuracy matching algorithm to filter the invalid services and decrease the computation of second matching operation. Simulation shows the accuracy of this algorithm in real engineer is super than 90%. In the future work, we will try to apply this idea to overlay network based service discovery.

6. Acknowledgement

This work was supported in part under Grant BK2010103 from Nature Science Foundation of Jiangsu Province, China.

7. References


