Service Discovery Oriented Management System Construction Method

To cite this article: Huawei Li and Ying Ren 2017 J. Phys.: Conf. Ser. 910 012009

View the article online for updates and enhancements.
Service Discovery Oriented Management System Construction Method

Huawei Li\textsuperscript{1a} and Ying Ren\textsuperscript{2b}

\textsuperscript{1}Shandong Business Institute, Yantai, 264001, China
\textsuperscript{2}Naval Aeronautical University, Shandong, 264000, China
Email: \textsuperscript{a}li_huawei@163.com, \textsuperscript{b}ren.ying@163.com

Abstract. In order to solve the problem that there is no uniform method for design service quality management system in large-scale complex service environment, this paper proposes a distributed service-oriented discovery management system construction method. Three measurement functions are proposed to compute nearest neighbor user similarity at different levels. At present in view of the low efficiency of service quality management systems, three solutions are proposed to improve the efficiency of the system. Finally, the key technologies of distributed service quality management system based on service discovery are summarized through the factor addition and subtraction of quantitative experiment.

1. Foreword
The service quality management system is a basic IT facility for service quality and providing data source for business logic. The system covers different core business modules, such as data storage services, data integration services and data access services, and responds to the request of the service sponsor in real time. Because of the continuous development of science and technology, it is inevitable that the quantity, speed and kind of service quality data are expanding constantly, and the quality of service data has entered a chaotic and disordered state. In the worst case, service requesters will be confused and easily frustrated by the mass, erratic, and disorderly quality of service that they face in a short period of time.

This paper provides a unified approach to design a distributed QoS management system. The purpose is to summarize the important factors that affect the performance of the service quality of the system (such as neighborhood effect, synergistic effect and effect way, etc.) and distill lessons to guide the next generation of high performance distributed service management system, so as to enhance the large-scale complex service discovery work performance and efficiency.

2. Total Distributed Architecture
Distributed service quality management system, designed to respond to a massive user QoS query request, through the management of user service interaction data of large-scale QoS information, analysis and prediction of loss, for all types of service discovery, service composition and service recommendation based support. Contains the following key modules.

2.1 Data Preprocessing Module
In distributed QoS management system, because the quality of data collected is a large amount of unstructured data, it is necessary to use stream preprocessing tools to organize in the acquisition phase. Using Cloudera Flume open source tools, Flume uses the distributed architecture of loose coupling, requires only a simple command will be deployed to the global API server, meet the demand of
hundreds of MB per second log data acquisition and transmission.

2.2 Persistence Module
Quality of service management data association is high, we use large file write method to optimize the overall performance in the persistence layer. The daemon monitor running on HDFS also supports the processing of large data set.

2.3 Distributed Cache Module
(1) Memcached is a high-performance distributed memory object caching system for dynamic Web applications to alleviate database load. It reduces the number of database reads by caching data and objects in memory, thereby increasing the speed of dynamic, database driven web sites.

(2) The Redis cluster is also a key-value storage system. Similar to Memcached, it supports the storage of value types with more, including string (string), list (linked list), set (set) and hash (hash type)

2.4 Computation Module
(1) Map stage: In accordance with the requirements of different measurement functions, the user of the same clustering cluster is hashed in the Nmapper computing node. Within each mapper computing node, we reconstruct the matrix in a small amount of iterations, in accordance with a predefined matrix decomposition strategy. At this stage, in order to improve efficiency, we do not seek predefined loss function to achieve the global optimum.

(2) Reduce stage: responsible for collecting the processing information of all mapper computing nodes, and then re classifying the users according to the requirements of the measurement function, and pushing the classification results back to the mapper in the Map phase.

3. Key index

3.1 Measure factor
In order to study the influence of measure function on the design of distributed QoS system, we set the following different measurement functions:

(1) Random grouping: That is, for any target user \( u_i \), the user is arbitrarily assigned to the nearest neighbor \( G(i) \) by using a random measure function. Strategy is as follows:

\[
G(i) = \{j | \forall j, \text{rand}(j) \% M = m(i), i \neq j \} \tag{1}
\]

Among, \( \text{rand} \) measure function is a hash function with time and random factor. The specific operation is to map the user \( u_i \) to the 16 bit string space, and then assign it to the Map group belonging to the target user through the redundancy operation. \( M \) is the number of mapper in this calculation, \( m(i) \) is the mapper index of the target user \( u_i \)

(2) PCC: Strategy is as follows:

\[
sim(i, j) = \frac{\sum_{s \in S} (r_{is} - r_{uj})(r_{js} - r_{uj})}{\sqrt{\sum_{s \in S} (r_{is} - r_{uj})^2 \sum_{s \in S} (r_{js} - r_{uj})^2}} \tag{2}
\]

The PCC measure function calculates the similarity of the user's history call record and determines the packet. Therefore, the size and number of mapper generated by the measure function are determined.

(3) PCC* in the field of service management, we can extend basic PCC with domain knowledge (geographic location information, call failure rate, and call time factor). Strategy is as follows:

\[
sim(i, j) = \frac{\sum_{s \in S} (r_{is} - r_{uj})(r_{js} - r_{uj})}{\sqrt{\sum_{s \in S} (r_{is} - r_{uj})^2 \sum_{s \in S} (r_{js} - r_{uj})^2}} \tag{3}
\]
3.2 Domain Factor

(1) Weighting strategy: The final QoS prediction performance of the target user is affected by multiple neighbors. We can adjust the relative importance of nearest neighbors by weighted and unweighted methods.

We define $W(0)$ as an unweighted policy, and the mathematical representation of the $W(0)$ policy for the target user is as follows:

$$U_i \leftarrow \sum_{j=1}^{G(i)} U_j$$

(4)

We define $W(1)$ as a weighted strategy, and the mathematical representation of the $W(1)$ policy for the target user $u_i$ is as follows:

$$U_i \leftarrow \sum_{j=1}^{G(i)} U_j \times w_{ij}$$

(5)

Where $w_{ij}$ is defined as follows:

$$w_{ij} = \frac{\text{sim}(ij)}{\sum_{k=1}^{G(i)} \text{sim}(ik)}$$

(6)

(2) Contribution factor: Given a target user, its nearest neighbor user not only plays a role in the decomposition model of the extended matrix, but also can transmit the cooperative information according to its own situation when the prediction result is synthesized. The target prediction $r_{ij}$ of the contributing factor $Mu$ is the following formula.

$$r_{ij}^\wedge = \mu \cdot r_{ij} + (1 - \mu) \cdot \frac{1}{|G(i)|} \sum_{g \in G(i)} r_{ig}^\ast$$

(7)

3.3 Alternative Factor

The calculation method is the key factor that affects the efficiency and accuracy of the matrix decomposition model. In order to extract the key design factors of the distributed QoS management system, the following algorithms are used:

(1) GD: The method of gradient descent is to compute the minimum value in the direction of gradient descent (and also to solve the maximum value along the gradient ascending direction). When the gradient descent algorithm is used to solve the optimization problem, the termination condition of the algorithm iteration is that the magnitude of the gradient vector is close to 0, and a very small constant min value can be set.

(2) SGD: The method of stochastic gradient descent takes each sample into a single sample. When the total number of samples is large, the method of the gradient descent iteration is much slower than the gradient descent method.

4. Experimental Results

4.1 Experimental Configure

The dataset used contains two types of QoS attributes: response time, and reliability. The range of response time is 0-20 seconds, and the range of reliability is 0-1. The single service quality management system has been unable to deal with the data of this magnitude, so the key factors that affect the QoS prediction will be quantitatively analyzed through a distributed framework. We randomly remove several QoS values to achieve sparse matrix purposes. By doing so, our QoS matrix is divided into two parts: the training set and the test set. The density matrix introduced in the experiment (density) this concept. Usually, if the density matrix is equal to 5%, we retained 5% QoS data as training set, the remaining 95% as a benchmark to verify the prediction performance of the system. All the experimental results are not separate sampling, we make it by taking the 100 times
The square root of the mean variance (RMSE) is a generic QoS prediction model testing standards, defined as follows:

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i,j} (\hat{r}_{ij} - \tilde{r}_{ij})^2}
\]  

(8)

Among them, \( \hat{r}_{ij} \) is forecasting results, \( N \) is the total number of prediction samples.

4.2. Key Index Effect Analysis

In order to measure the effect of key indicators on the final prediction effect, we use factor addition and subtraction to evaluate the predictive performance. Table 1 summarizes the key elements involved in the test.

4.3. Overall Effect

We used the factor addition and subtraction in the quantitative analysis to analyze the key factors. All settings are using the same training set and test set. We set the number of mapper (clustering) to 50.

Table 1 Significance Of Key Indicators

| Notation | Definition and Description |
|----------|----------------------------|
| rand     | Random measure function with time factor |
| CC       | A universal measure of degree of similarity in industry |
| PCC*     | Extended PCC measure function combined with service domain knowledge |
| W(0)     | Matrix decomposition framework without weight |
| W(1)     | A weighted matrix decomposition framework based on user similarity |
| CI(0)    | QoS prediction method without contribution factor |
| CI(1)    | QoS prediction method with contribution factor |
| GD       | Batch gradient descent algorithm |
| SGD      | Stochastic gradient descent algorithm |
| Adagrad  | A randomized gradient descent algorithm with adaptive learning rate |

Table 2 Comparison of prediction accuracy of response time

| Method          | QoS Attribute | D=5% | D=95% |
|-----------------|---------------|------|------|
| rand+W(0)+GD    | Response Time | 16.05| 16.43|
| PCC+W(0)+GD     |               | 5.13 | 3.4  |
| PCC*+W(0)+GD    |               | 4.36 | 2.84 |
| rand+W(1)+GD    |               | 13.34| 14.12|
| PCC+W(1)+GD     |               | 4.3  | 3.0  |
| PCC*+W(1)+GD    |               | 3.85 | 1.74 |

Tables I and II show the accuracy of the key factor combinations in the two sets of data on QoS response time and reliability. We can observe that:(1) Compared to PCC and extended version PCC, the prediction of random functions is always the worst. Different range of QoS characteristics directly lead to large fluctuations in the RMSE index value.(2) In order to solve the problem, adding the non knowledge measurement function will affect the final prediction effect of the service quality management system. With neighborhood factors as dimensions, both the weighted strategy and the policy with contribution factor are always optimal, regardless of the response time or reliability of the two sets of data.(3) With the method factor as the dimension, the performance is not obvious. In the response time data set, the Adagrad policy produces better prediction results in most cases. However,
this difference is not apparent in the reliability data set. Even when the sample density was high, SGD showed greater predictive power.

Table 3 Comparison of reliability prediction accuracy

| Method            | QoS Attribute | D=5% | D=95% |
|-------------------|---------------|------|-------|
| rand+W(0)+GD      | Reliability   | 3.2  | 6.2   |
| PCC+W(0)+GD       |               | 1.37 | 2.51  |
| PCC°+W(0)+GD      |               | 1.45 | 3.41  |
| rand+W(1)+GD      |               | 2.85 | 5.6   |
| PCC+W(1)+GD       |               | 1.2  | 2.49  |
| PCC°+W(1)+GD      |               | 2.1  | 2.65  |

4.4. Distributed Efficiency Analysis

Figure 1 Comparison of operating efficiencies
We see that the calculation method does not have much effect on the prediction effect, which is in fact expected: The use of the method is to improve the efficiency of the quality of service management system to meet the online query requirements. We focus on the impact of computational approaches on distributed performance. We use the response time data set, the PCC* measure function and the coordination factor in the weighted W(1) and the band contribution factor CI(1) strategy. By adjusting the number of mapper nodes, we observe three sets to obtain globally optimal corresponding time overhead.

Figure I is a comparison of operating efficiency. As can be seen, given a low density learning sample, the actual running time of SGD and Adagrad increase linearly with the number of mapper nodes. At the same time, the actual running of GD which is using the strategy of adding all samples as gradients increases exponentially with the increase of mapper nodes and the operation efficiency is much lower than that of SGD and Adagrad. In the same number of mapper nodes, Adagrad can be better than the original version of SGD because it can determine the learning step size according to the previous historical gradient dynamics. In the case of sufficient sample number (95%), when the number of mapper nodes is relatively small, the inflection point is observed when the number of mapper exceeds the threshold. This phenomenon can be resolved as the time cost of the single wheel gradient will increase when the number of mapper nodes increases. However, when mapper is at a balanced point, the total number of iterations will be reduced (the loss function converges faster), so the overall run time will be lower than the small mapper setting. The stability of the global optimal convergence of the matrix decomposition model is shown in Figure 1.

5. Acknowledgement
Project of scientific research development of universities in Shandong Province in 2017(J17KB175)

6. Reference
[1] Taha Osman, Dhawal Thakker, David Al-Dabass. Semantic-driven matchmaking of web services using case-based reasoning[C]//Web Services, 2006. ICWS’06. International Conference on. IEEE, 2006: 29-36.
[2] Luciano Baresi, Matteo Miraz, Pierluigi Plebani. A distributed architecture for efficient web service discovery[J]. Service Oriented Computing and Applications, 2015: 1-17.
[3] Zhizhong Liu, Yulan Liu, Yihui He. A two-layered p2p model for semantic service discovery[C]//New Trends in Information Science and Service Science (NISS), 2010 4th International Conference on. IEEE, 2010: 41-46.
[4] Min-Xiou Chen, Fu-Hsing Sung. Integrated service discovery architecture for heterogeneous networks[J]. International Journal of Communication Systems, 2015.
[5] Vinay Kumar Tiwari, Amit Kumar Gautam, Saurabh Diwaker. Web service discovery and integration with qos parameter using soa based repository[C]//Computer Engineering and Applications (ICACEA), 2015 International Conference on Advances in. IEEE, 2015: 1014-1009.
[6] Angelo Furno, Eugenio Zimeo. Efficient cooperative discovery of service compositions in unstructured p2p networks[C]//ParalM,Distributed and Network-Based Processing (PDP), 2013 21st Euromicro International Conference.
[7] Wenyu Zhang, Shuai Zhang, Feng Qi, Ming Cai. Self-organized p2p approach to manufacturing service discovery for cross-enterprise collaboration[J]. Systems, Man, and Cybernetics: Systems, IEEE Transactions on, 2014.44(3): 263-276.
[8] Liang Chen, Jian Wu, Zibin Zheng, Michael R Lyu, Zhaohui Wu. Modeling and exploiting tag relevance for web service mining[J]. Knowledge and information systems, 2014.39(1): 153-173.
[9] Jiemeng Zhu, Pinjia He, Zibin Zheng, Michael R Lyu. Towards online, accurate, and scalable qos prediction for runtime service adaptation[C]//Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on. IEEE, 2014: 318-327.
[10] Pinjia He, Jiemeng Zhu, Zibin Zheng, Jianlong Xu, Michael R Lyu. Location—based hierarchical matrix factorization for web service recommendation[C]//Web Services (ICWS), 2014 IEEE
International Conference on, IEEE, 2014: 297-304.

[11] Liang Chen, Qi Yu, Philip S Yu, Jian Wu. Ws-hfs: A heterogeneous feature selection framework for web services mining [C] // Web Services (ICWS), 2015 IEEE International Conference on. IEEE, 2015: 193-200.

[12] Shuiguang Deng, Longtao Huang, Wei Tan, Zhaohui Wu. Top-automatic service composition: A parallel method for large-scale service sets [J]. Automation Science and Engineering, IEEE Transactions on, 2014, 11(3): 891-905.

[13] Xinhuai Tang, Feilong Tang, Liu Bing, Delai Chen. Dynamic web service composition based on service integration and htn planning [C] // Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2013 Seventh International Conference on. IEEE, 2013: 307-312.

[14] Gopal N Rai, GR Gangadharan, Vineet Padmanabhan. Algebraic modeling and verification of web service composition [J]. Procedia Computer Science, 2015, 52: 675-679.

[15] Nikolaos Tsapanos, Anastasios Tefas, Nikolaos Nikolaidis, Ioannis Pitas. A distributed framework for trimmed kernel k-means clustering [J]. Pattern Recognition, 2015, 48(8): 2685-2698.

[16] Aamer Shah, Angshul Majumdar. Accelerating low-rank matrix completion on gpus [C] // Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on. IEEE, 2014: 82-187.