A keyword-Recognized Service Recommendation Method on Hadoop using Map Reduce for Big Data Applications

M Nandakumari[1], R Suresh[2],
[1]*PG Scholar, Sri Manakula Vinayagar Engineering College, Pondicherry, India*
[2] *Associate Professor, Sri Manakula Vinayagar Engineering College, Pondicherry, India*
[1]*m.nandakumaari@gmail.com*
[2]*sureshramanujam78@gmail.com*

Abstract— **This paper says about providing flexible recommendations to user Service recommendations system has become a powerful tool. Since the rapid growth of information in every sectors such as the amount of services, customers, online information etc., big data analysis problem has been raised in service recommendation system. So, Scalability and inefficiency problem arises in traditional recommender system while processing large amount of data. Moreover, some of the existing service recommendation systems give same ratings and rankings to services to users, but that doesn't satisfy users' preferences, there is a failure in providing personalized requirements to users. So, I propose a keyword-Recognized Service Recommendation Method to address the above challenges. By giving the recommendation list, the users are effectively recommended with appropriate services. Keywords are being used to indicate the users' preference to generate recommendation with the adaptation of collaborative filtering algorithm. Finally, Hadoop using MapReduce is implemented to improve scalability and efficiency in big data environment.**

Keywords— **recommender system, preference, keyword, Big Data, MapReduce, Hadoop**

I. INTRODUCTION

In recent years, the amount of data in our world has been increasing explosively, and analyzing large data sets—so-called "Big Data"— becomes a key basis of competition underpinning new waves of productivity growth, innovation, and consumer surplus [1]. Then, what is "Big Data"?, Big Data refers to datasets whose size is beyond the ability of current technology, method and theory to capture, man-age, and process the data within a tolerable elapsed time. Today, Big Data management stands out as a challenge for IT companies. The solution to such a challenge is shifting increasingly from providing hardware to provisioning more manageable software solutions [2]. Big Data also brings new opportunities and critical challenges to industry and academia [3] [4]. Similar to most big data applications, the big data tendency also poses heavy impacts on service recommender systems. With the growing number of alternative services, effectively recommending services that users preferred have become an important research issue. Service recommender systems have been shown as valuable tools to help users deal with services overload and provide appropriate recommendations to them. Examples of such practical applications include CDs, books, web pages and various other products now use recommender systems [5], [6], [7]. Over the last decade, there has been much research done both in industry and academia on developing new approaches for service recommender systems [8], [9].

II. CLOUD COMPUTING AND MAPREDUCE

Cloud computing is a successful paradigm of service oriented computing and has revolutionized the way computing infrastructure is abstracted and used. The major goal of cloud computing is to share resources, such as infrastructure, platform, software, and business process [14]. Cloud computing can provide effective platforms to facilitate parallel computing, which has gained significant attention in recent years to process large volume of data.

There are several cloud computing tools available, such as Hadoop (http://hadoop.apache.org/), Mahout (http:// mahout.apache.org/), MapReduce of Google [15], the Dynamo of Amazon.com [16], the Dryad of Microsoft and Neptune of Ask.com [17], etc. Among these tools, Hadoop is the most popular open source cloud computing platform inspired by MapReduce and Google File System papers [18], which supports MapReduce programming framework and mass data storage with good fault tolerance. MapReduce is a popular distributed implementation model proposed by Google, which is inspired by map and reduce operations in the Lisp programming language. Nowadays, the trend "everything as a service" has been creating a Big Services era due to the foundational architecture of services computing. And "servicelization" is the way of offering social networking services, big data analytics, and Internet services [19] [20]. Thus the cloud computing tools aforementioned can be used to improve the scalability and efficiency of service recommendation methods in the "Big Data" environment.

III. RECOMMENDER SYSTEM AND COLLABORATIVE FILTERING

Recommender systems developed as an independent re-search area in the mid-1990s when recommendation problems started focusing on rating models [10], [11]. According to the definition of recommender system in [12], recommender system can be defined as system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful services in a large space of possible options. Current recommendation methods usually can be classified into three main categories: content-based, collaborative, and hybrid recommendation approaches [13]. Content-based approaches recommend services similar to those the user preferred in the past. Collaborative filtering (CF) approaches recommend services to the user that users with similar tastes preferred in the past. Hybrid approaches combine content-based and CF methods in several different ways. CF algorithm is a classic personalized recommendation algorithm, which is widely used in many commercial recommender systems [13]. In CF based systems, users receive recommendations based on people who have similar tastes and preferences, which can be further classified into item-based CF and user-based CF. In item-based systems; the predicted rating depends on the ratings of other similar items by the same user. While in user-based systems, the prediction of the rating of an item for a user depends upon the ratings of the same item rated by similar users. And in this work, we will take advantage of a user-based CF algorithm to deal with our problem.

IV. KEYWORD RECOGNIZED SERVICE RECOMMENDATION METHOD

In this paper, we propose a keyword-aware service recommendation method, named KASR. In this method, keywords are used to indicate both of users' preferences and the quality of candidate services. A user-based CF algorithm is adopted to generate appropriate recommendations. KASR aims at calculating a personalized rating of each candidate service for a user, and then presenting a personalized service recommendation list and recommending the most appropriate services to him/her. Moreover, to improve the scalability and efficiency of our recommendation method in "Big Data" environment, we implement it in a MapReduce framework on Hadoop by splitting the proposed algorithm into multiple MapReduce phases. Table 1 summarizes the basic symbols and notations used in this paper.

TABLE 1
Basic symbols and notations

Symbol	Definition
K	The keyword-candidate list, $K=\{k_1, k_2, \ldots, k_n\}$
APK	The preference keyword set of the active user
PPK	The preference keyword set of a previous user
$sim(APK,PPK)$	The similarity between APK and PPK
$\vec{W}_P$	A preference weight vector
$\vec{W}_{AP}$	The preference weight vector of the active user
$\vec{W}_{PP}$	The preference weight vector of a previous user

V. KEYWORD CANDIDATE LIST AND DOMAIN THESAURUS

In our method, two data structures, "keyword-candidate list" and "specialized domain thesaurus", are introduced to help obtain users' preferences.

Keyword Candidate List: The keyword-candidate list is a set of keywords about users' preferences and multi-criteria of the candidate services, which can be denoted as K= {k1, k2, k3…, kn}, *n* is the number of the key-words in the keyword-candidate list. An example of a simple keyword-candidate list of the hotel reservation system is described in Table 2. Keywords in the keyword-candidate list can be a word or multiple words related with the quality criteria of candidate services.

In this paper, the preferences of previous users will be extracted from their reviews for candidate services and formalized into a keyword set. Usually, since some of words in reviews cannot exactly match the corresponding keywords in the keyword-candidate list which characterize the same aspects as the words. The corresponding key-words should be extracted as well. In this paper, we assume that specialized domain thesauruses are built to sup-port the keyword extraction, and different domain thesauruses are built for different service domains.

TABLE 2
Keyword-candidate list of hotel reservation system

No.	Keyword	No.	Keyword	No.	Keyword
1	Service	7	Transportation	13	Airport; Train
2	Room	8	Family; Friends	14	Wi-Fi
3	Shopping	9	Location	15	Environment
4	Cleanliness	10	View	16	Bar
5	Food	11	Quite	17	Beach
6	Value	12	Fitness		

Domain Thesaurus: A domain thesaurus is a reference work of the keyword-candidate list that lists words grouped together according to the similarity of key-word meaning, including related and contrasting words and antonyms [21] [22]. An example of a simple domain thesaurus of hotel reservation system is shown in Fig. 2. As shown in Fig. 2, the words in the red rectangle are the key-words in the corresponding keyword-candidate list, and the words in the ovals are the related words of the key-words. Often, domain thesauruses are updated regularly to ensure the timeliness of the words.

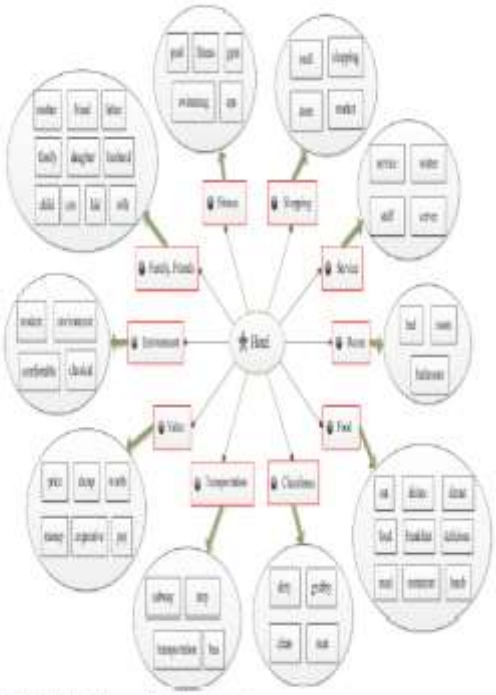

Fig. 2 An example of a simple domain thesaurus of hotel reservation system.

VI. KEYWORD RECOGNIZED SERVICE RECOMMENDATION METHOD

The main steps of KASR are depicted in Fig. 3, which are described in detail as follows.

1. **Capture user preferences by a keyword-aware approach:**

In this step, the preferences of active users and previous users are formalized into their corresponding preference keyword sets respectively. In this paper, an active user re-fers to a current user needs recommendation.

 Preferences of an active user. An active user can give his/her preferences about candidate services by selecting keywords from a keyword-candidate list, which reflect the quality criteria of the services he/she is concerned about. The preference keyword set of the active user can be

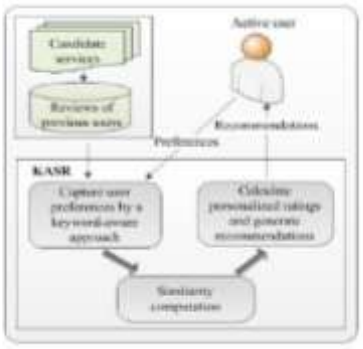

Fig. 3 KASR's main steps.

noted as APK={ak1, ak2, ak3.., akn}, where aki is the keyword selected from the keyword-candidate list by the active user, *l* is the number of selected keywords.
Besides, the active user should also select the importance degree of the keywords. The importance degree of the keywords is shown in Table 3: "1" represents the general, "3" represents important and "5" represents very important.

TABLE 3
Importance degree of the keywords

Measurement	general		important		very important
Importance degree	1	2	3	4	5

 Preferences of previous users. The preferences of a previous user for a candidate service are extracted from his/her reviews for the service according to the keyword-candidate list and domain thesaurus. And a review of the previous user will be formalized into the preference key-word set of him/her, which can be denoted as PPK={pk1, pk2, pk3,… ,pkn}, where pki is the *i*th keyword extracted from the review, *h* is the number of extracted keywords.

The keyword extraction process is described as follows:
a) Preprocess: Firstly, HTML tags and stop words in the reviews snippet collection should be removed to avoid affecting the quality of the keyword extraction in the next stage. And the Porter Stemmer algorithm (keyword strip-ping) [23] is used to remove the commoner morphological and inflexional endings from words in English. Its main use is as part of a term normalization process that is usually done when setting up Information Retrieval systems [23].
b) Keyword extraction: In this phase, each review will be transformed into a corresponding keyword set according to the keyword-candidate list and domain thesaurus. If the review contains a word in the domain thesaurus, then the corresponding keyword should be extracted into the preference keyword set of the user. For example, if a review of a previous user for a hotel has the word "spa", which is corresponding to the keyword ``Fitness'' in the domain

thesaurus, then the keyword ``Fitness'' should be contained in the preference keyword set of the previous user. If a keyword appears more than once in a review, the times of repetitions will be recorded. In this paper, it is regarded that keywords appearing multiple times are more important. The times of repetitions will be used to calculate the weight of the key-word in preference keyword set in the next step.

(2) Similarity computation: The second step is to identify the reviews of previous users who have similar tastes to an active user by finding neighbourhoods of the active user based on the similarity of their preferences. Before similarity computation, the reviews unrelated to the active user's preferences will be filtered out by the intersection concept in set theory. If the intersection of the preference keyword sets of the active user and a previous user is an empty set, then the preference keyword set of the previous user will be filtered out. Two similarity computation methods are introduced in our recommendation method: an approximate similarity computation method and an exact similarity computation method. The approximate similarity computation method is for the case that the weights of the keywords in the preference keyword set are unavailable, while the exact similarity computation method is for the case that the weight of the keywords are available.

a) Approximate similarity computation

A frequently used method for comparing the similarity and diversity of sample sets, Jaccard coefficient, is applied in the approximate similarity computation. Jaccard coefficient is measurement of asymmetric information on binary (and non-binary) variables, and it is useful when negative values give no information. The similarity between the preferences of the active user and a previous user based on Jaccard coefficient is described as follows:

$$sim(APK, PPK) = Jaccard(APK, PPK) = \frac{|APK \cap PPK|}{|APK \cup PPK|} \quad (1)$$

where *APK* is the preference keyword set of the active user, *PPK* is the preference keyword set of a previous user. And the weight of the keywords is not considered in this approach.

b) Exact similarity computation

A cosine-based approach is applied in the exact similarity computation, which is similar to the Vector Space Model (VSM) in information retrieval [24] [25].

In this cosine-based approach, The preference keyword sets of the active user and previous users will be transformed into *n*-dimensional weight vectors respectively, namely preference weight vector, which can be denoted as wp=[w1, w2, w3,… wn], n is the number of keywords in the keyword-candidate list, wi is the weight of the keyword *ki* in the keyword-candidate list. If the keyword *ki* is not contained in the preference keyword set, then the weight of *ki* in the preference weight vector is 0, i.e., *wi*=0. The preference weight vectors of the active user and a previous user are noted as Wap and Wpp respectively.

In this paper, we use the Analytic Hierarchy Process (AHP) model to decide the weight of the keywords in the preference keyword set of the active user. AHP method is provided by Saaty in 1970s to choose the best satisfied business role for its hierarchy nature [26]. The weight computing based on the AHP model is decide as follows: Firstly, we construct the pair-wise comparison matrix in terms of the relative importance between each two key-words. The pair-wise comparison matrix *A*m = (*aij*) *m* must satisfy the following properties, *aij* represents the relative importance of two keywords:

1) $a_{ij} = 1, \quad i = j = 1,2,3,\ldots, m$

2) $a_{ij} = 1/a_{ji}, \quad i, j = 1,2,3,\ldots, m \text{ and } i \neq j$

3) $a_{ij} = a_{ik}/a_{jk}, \quad i, j, k = 1,2,3,\ldots, m \text{ and } i \neq j$

After checking the consistence of the matrix, then we calculate the weight by the following function:

$$w_i = \frac{1}{m}\sum_{j=1}^{m}\frac{a_{ij}}{\sum_{k=1}^{m} a_{kj}} \quad (2)$$

where *aij* is the relative importance between two keywords, *m* is the number of the keywords in the preference keyword set of the active user. The weight vector of the preference keyword set of a previous user can be decided by *the term frequency/inverse document frequency* (*TF-IDF*) measure [27], which is one of the best-known measures for specifying the weight of key-words in Information Retrieval. In the *TF-IDF* approach, to calculate the preference weight vector of a previous user *u'*, "all reviews" by user *u'* should be collected. Here, "all reviews" contain the re-views by user *u'* for the candidate services and similar services not in the candidate services. The reviews should also be transformed into keyword sets respectively according to the keyword-candidate list and the domain thesaurus. *TF*, the term frequency of the keyword *pki* in the preference keyword set of user *u'* is defined as

$$TF = \frac{N_{pk_i}}{\sum_g N_{pk_i}} \quad (3)$$

Where N_{pki} is the number of occurrences of the keyword *pki* in all the keyword sets of the reviews commented by the same user *u'*, *g* is the number of the keywords in the preference keyword set of the user *u'*. The inverse document frequency (*IDF*) is obtained by dividing the number of all reviews by the number of reviews containing the keyword *pki*.

$$IDF = \log \frac{|R'|}{|r' : pk_i \in r'|} \quad (4)$$

where $|R'|$ is the total number of the reviews commented by user *u'*, and $|r':pk_i \in r'|$ is the number of reviews where keyword *pki* appears. So the *TF-IDF* weight of the keyword *pki* in the preference keyword set of user *u'* can be decided by the following function:

$$w_{pk_i} = TF \times IDF = \frac{N_{pk_i}}{\sum_g N_{pk_i}} \times \log \frac{|R'|}{|r' : pk_i \in r'|} \quad (5)$$

Then the similarity based on the cosine-based approach is defined as follows:

$$sim(APK, PPK) = \cos(\vec{W}_{AP}, \vec{W}_{PP}) = \frac{\vec{W}_{AP} \bullet \vec{W}_{PP}}{\left\|\vec{W}_{AP}\right\|_2 \times \left\|\vec{W}_{PP}\right\|_2}$$

$$= \frac{\sum_{i=1}^{n} \vec{W}_{AP,i} \times \vec{W}_{PP,i}}{\sqrt{\sum_{i=1}^{n} \vec{W}^2_{AP,i}} \sqrt{\sum_{i=1}^{n} \vec{W}^2_{PP,i}}} \quad (6)$$

Where W_{AP} and W_{PP} are respectively the preference weight vectors of the active user and a previous user. $W_{AP,\,I}$ is the *i*-th dimension of W_{AP} and represents the weight of the key-word *ki* in preference keyword set *APK*, $W_{pp,I}$ is the *i*-th dimension of W_{pp} and represents the weight of the key-word *ki* in preference keyword set *PPK*.

(3) Calculate personalized ratings and generate recommendations:

Based on the similarity of the active user and previous users, further filtering will be conducted. The thresholds given in two similarity computation methods are different, which are both empirical values. Once the set of most similar users are found, the personalized ratings of each candidate service for the active user can be calculated. Finally, a personalized service recommendation list will be presented to the user and the service(s) with the highest rating(s) will be recommended to him/her. Here, we use a weighted average approach to calculate the personalized rating *pr* of a service for the active user.

$$pr = \bar{r} + k \sum_{PPK_j \in R} sim(APK, PPK_j) \times (r_j - \bar{r}) \quad (7)$$

$$k = 1 \Big/ \sum_{PPK_j \in R} sim(APK, PPK_j)$$

Where $sim(\text{APK},\text{PPK}_j)$ is the similarity of the preference keyword set of the active user APK and the preference keyword set of a previous user *PPKj*; multiplier *k* serves as a normalizing factor; R denotes the set of the remaining preference keyword sets of previous users after filtering; *rj* is the rating of the corresponding review of *PPKj*, and r is defined as the average ratings of the candidate service. Repeating the steps above, we can calculate the personalized ratings of all candidate services for the active user. Then we can rank the services by the personalized ratings and present a personalized service recommendation list to him/her. Without loss of generality, we assume that the services with higher ratings are more preferable to the user. So the service(s) with the highest rating(s) will be recommended to the active user. Alternatively, we can recommend the Top-K services to the user.

VII. CONCLUSIONS

In this paper, we have proposed a keyword-Recognized Service Recommendation Method. In this method, keywords are used to indicate users' preferences, and a user-based Collaborative Filtering algorithm is adopted to generate appropriate recommendations. More specifically, a keyword-candidate list and domain thesaurus are provided to help obtain users' preferences. The active user gives his/her preferences by selecting the keywords from the keyword-candidate list, and the preferences of the previous users can be extracted from their reviews for services according to the keyword-candidate list and domain thesaurus. My method aims at presenting a personalized service recommendation list and recommending the most appropriate service(s) to the users. Moreover, to improve the scalability and efficiency of this method in "Big Data" environment, this is implemented on a MapReduce framework in Hadoop platform. Finally, the Keyword-Recognized service recommendation significantly improves the accuracy and scalability of service recommender systems over existing approaches.

REFERENCES

[1] J. Manyika, M. Chui, B. Brown, et al, "Big Data: The next frontier for innovation, competition, and productivity," 2011.

[2] C. Lynch, "Big Data: How do your data grow?" Nature, Vol. 455, No. 7209, pp. 28-29, 2008.

[3] F. Chang, J. Dean, S. Ghemawat, and W. C. Hsieh, "Bigtable: A distributed storage system for structured data," ACM Transactions on Computer Systems,Vol. 26, No. 2 (4) , 2008.

[4] W. Dou, X. Zhang, J. Liu, J. Chen, "HireSome-II: Towards Privacy-Aware Cross-Cloud Service Composition for Big Data Applica-tions," IEEE Transactions on Parallel and Distributed Systems, 2013.

[5] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," IEEE Internet Computing, Vol. 7, No.1, pp. 76-80, 2003.

[6] M. Bjelica, "Towards TV Recommender System Experiments with User Modeling," IEEE Transactions on Consumer Electronics, Vol. 56, No.3, pp. 1763-1769, 2010.

[7] M. Alduan, F. Alvarez, J. Menendez, and O. Baez, "Recommender System for Sport Videos Based on User Audiovisual Consumption," IEEE Transactions on Multimedia, Vol. 14, No.6, pp. 1546-1557, 2013.

[8] Y. Chen, A. Cheng and W. Hsu, "Travel Recommendation by Min-ing People Attributes and Travel Group Types From Community-Contributed Photos". IEEE Transactions on Multimedia, Vol. 25, No.6, pp. 1283-1295, 2012.

[9] Z. Zheng, X Wu, Y Zhang, M Lyu, and J Wang, "QoS Ranking Pre-diction for Cloud Services," IEEE Transactions on Parallel and Dis-tributed Systems, Vol. 24, No. 6, pp. 1213-1222, 2013.

[10] W. Hill, L. Stead, M. Rosenstein, and G. Furnas, "Recommending and Evaluating Choices in a Virtual Community of Use," In CHI '95 Proceedings of the SIGCHI Conference on Human Factors in Com-puting System, pp. 194-201, 1995.

[11] P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews," In CSCW '94 Proceedings of the 1994 ACM conference on Computer supported cooperative work, pp. 175-186, 1994.

[12] R. Burke, "Hybrid Recommender Systems: Survey and Experi-ments," User Modeling and User-Adapted Interaction, Vol. 12, No.4, pp. 331-370, 2002.

[13] G. Adomavicius, and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the Stateof- the-Art and Possi-ble Extensions," IEEE Transactions on Knowledge and Data Engi-neering, Vol.17, No.6 pp. 734-749, 2005.

[14] D. Agrawal, S. Das, A. El Abbadi, "Big Data and cloud computing: new wine or just new bottles?" Proceedings of the VLDB Endow-ment, Vol. 3, No.1, pp. 1647-1648, 2010.

[15] J. Dean, and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Communications of the ACM, Vol. 51, No.1, pp. 107-113, 2005.

[16] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Laksh-man, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, "Dynamo: Amazons highly available key-value store," In: Proceed-ings of the 21st ACM Symposium on Operating Systems Principles, pp. 205-220, 2007.

[17] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad:Distributed data-parallel programs from sequential build-ing blocks," European Conference on Computer Systems, pp. 59-72, 2007.

[18] S. Ghemawat, H. Gobioff, and S. T. Leung, "The Google File Sys-tem," The 19th ACM Symposium on Operating Systems Principles, pp. 29-43, 2003.

[19] L. Zhang, "Editorial: Big Services Era: Global Trends of Cloud Computing and Big Data". IEEE Transactions on Services Computing, Vol. 5, No. 4, pp. 467-468, 2012.

[20] Z. Luo, Y. Li and J. Yin, "Location: a feature for service selection in the era of big data," 2013 IEEE 20th International Conference on Web Service, pp. 515-522, 2013.

[21] H. Schütze and J. O. Pedersen, "A cooccurrence-based thesaurus and two applications to information retrieval," Information Processing & Management, Vol. 33, No. 3, pp. 307-318, 1997.

[22] Y. Jing and W. Croft, "An association thesaurus for information retrieval," Proceedings of RIAO, Vol. 94, No. 1994, pp.146-160, 1994.

[23] B. Issac and W. J. Jap, "Implementing spam detection using baye-sian and porter stemmer keyword stripping approaches," TEN-CON 2009-2009 IEEE Region 10 Conference, pp. 1-5, 2009.

[24] P. Castells, M. Fernandez, and D. Vallet, "An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval," IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No.2, pp. 261-272, February 2007.

[25] Y. Zhu, Y. Hu,"Enhancing search performance on Gnutella-like P2P systems," IEEE Transactions on Parallel and Distributed Sys-tems, Vol. 17, No. 12, pp. 1482-1495, 2006.

[26] A. Chu, R. Kalaba, and K. Spingarn, "A comparison of two me-thods for determining the weights of belonging to fuzzy sets", Journal of Optimization Theory and Applications, Vol. 27, No.4, pp.531-538, 1979

