Discovering Heterogeneous Evolving Web Service Communities Using Semi-Supervised Non Negative Matrix Factorization

Pankaj Andhale, Manjeet Rege, and Qi Yu

***Abstract*—There has been a paradigm shift in the design of software applications. Each offering is designed as a service that can easily integrate with other services. As a result, there are large numbers of web services that are constantly created and consumed. One of the key challenges is to create relevant sub-space domain for web services that can identify the best suitable service to perform a task. Web service discovery process addresses the problem of selecting the best service. In this paper, we propose semi-supervised service community discovery on heterogeneous evolving web services data. The semi-supervised knowledge about the current time step is incorporated into the heterogeneous evolving environment. The evolving changes in the web services are captured and service community is created based on their prior and current knowledge of the web services. Also, the heterogeneous model helps us to create a highly relevant evolving sub space taking into account the operations and services performed by the web services along with the terms used to define the web service simultaneously.**

***Index Terms*—Semi-supervised, heterogeneous, co-clustering, evolving data, web services.**

I. Introduction

With the proven benefits of service oriented architecture, more and more software development teams are designing software offerings as a service. The service oriented architecture not only allows easy integration of services internally within the product but also provides easy integration of services with external products. In the service oriented architecture, we have a web service provider providing the service and a web service consumer who consumes the available web services. The process of finding the best service for a given task is known as web service discovery.

The web service discovery process comes into play during software development for finding similar existing web services, as well as during the execution phase where the service consumer is trying to identify or consume the existing web service. The public UDDI registry is maintained which contains all the information about the web services [1]. A web service lookup is performed based on the search keywords.

Manuscript received January 4, 2013; revised April 23, 2013. This work was supported by Intuit Inc.

Pankaj Andhale is with Intuit Inc. Mountain View, CA 94043 USA (e-mail: pankaj_andhale@intuit.com).

Manjeet Rege and Qi Yu are with College of Computing and Information Sciences at Rochester Institute of Technology, NY 14623 USA (e-mail: mr@cs.rit.edu; qi.yu@rit.edu).

This approach is not an efficient one as the search is based on keywords and lacks semantic search approach to retrieve relevant web services.

There have been prior efforts on achieving service discovery by using the means of service communities. Specifically, service community formations are of two types, viz., top down approach and the bottom up approach. The top down approach imposes requirements on the service provider where categories of services are manually defined by domain experts. The bottom up approach on the other hand does not impose any requirements on the service providers. The earlier approaches while being effective are unable to capture the heterogeneous evolving changes in the web service community discovery. Moreover, they are completely unsupervised. That is, they are incapable of incorporating any available domain knowledge for efficient community discovery. To this end, we present a novel approach to generate service communities, contributions of which can be summarized as follows:

a) The service communities generated using our approach capture the evolving nature of web services.

b) We incorporate user provided semi-supervised knowledge in the form of must link and cannot link constraints.

Rest of the paper is organized as follows. Related work is covered in Section II. We will explain the proposed frame work for service discovery in section III and IV, followed by experiments and results in section V. Finally, we conclude in Section VI.

II. Related Work

In this section, we briefly review some of the prior research related to web service community discovery.

Peer to Peer based solution provides a distributed platform for the web discovery [1]. It is a decentralized, self organizing network of peers where a peer has the look up table to route the request as well as a server providing service access. The Speed-R system [1] is web service storage and discovery system which uses the Peer to Peer model along with the ontologies.

Elgazzar *et al.* [2] proposed the web service discovery approach based on extracting the five parameters from the Web Service Definition Language (WSDL) document and computing the similarity measure between them. The five parameters used were service name, WSDL contents, WSDL messages, WSDL ports and WSDL types. Clustering of web services is performed and similar services are grouped together based on the quality threshold selected. In [3], a similar approach has been adopted using a semantic similarity measure.

An algorithm performing singular value decomposition (SVD) based co-clustering to form heterogeneous communities was proposed by Yu and Rege [4]. They used the term frequency and inverse document term frequency to measure similarity between WSDL documents. The SVD based co-clustering is not able to handle the multiple numbers of communities for operations and services. Recently Salunke *et al.* [5] proposed a semi-supervised approach to incorporate user knowledge on the static data and perform co-clustering using the block value decomposition technique to create service communities.

As can be seen from the above discussion, these approaches deal with static data. That is, they are unable to handle and discover the evolving service communities.

III. Overview of Proposed work

A. Creating the Star Structure Schema of Heterogeneous Web Service Data

We represent the web services data using a star structure [6]-[8] where a central data type is connected to other data types. Specifically, the data is stored using 2 matrices, viz., *OT,* an operations-terms matrix, and *ST,* which is a services-terms matrix.

B. Handling the Evolving Web Services Data

In the evolving environment [9], [10], the data changes at each time step. The number of terms, operations and services that define the web service may increase or decrease or the same words might be used in different domains to define a new web service. The clustering algorithm must handle these evolving changes in the data at every time step and perform clustering of the current data in accordance with the historic data [10]. That is, the current clustering must not differ too much from the previous clustering results, and should be able to capture the shift or drift over multiple time steps [11]. We use a parameter α which takes on values between 0 and 1, and specifies the emphasis on current or historic data [10], [11]. Lower value indicates more emphasis towards historic data. As a result, current data is clustered in accordance with historic data. Similarly, a value closer to 1 indicates that more weight is assigned to the current time step data and the clustering is performed according to the current data. There are also different scenarios that need to be handled in the evolving environment as we discuss later.

C. Incorporating the Semi-Supervised Knowledge

The user provided knowledge about the web services is incorporated in the form of must link and cannot link constraints [5], [6], [12]-[15] on the web service terms. These constraints ensure that similar terms that define a web services belonging to the same cluster must be placed together and similar terms that define web services belonging to different clusters must be placed apart. The current semi-supervised knowledge will guide the clustering algorithm to cluster similar web services into one cluster.

D. Creating the Compact Representation

Most of the real world datasets are sparse in nature. In the web service datasets, we have few terms, operations and services that define a web service. To efficiently compute the lager datasets we use the *Colibri* technique [16] to perform low rank approximation of the web service dataset so that the compact representation [17]-[19] preserves the data sparseness. The *Colibri* technique is applied to the *OT* and *ST* matrices.

E. Non-Negative Matrix Factorization Based Co-Clustering

We use the non-negative matrix factorization [5], [6], [15], [20] approach to perform co-clustering which produces three non-negative matrices. Two of these matrices consist of the cluster centroids and the third one is the indicator matrix which shows the co-relation of the data with its cluster. The results obtained using the non negative matrix factorizations are intuitive as opposed to the SVD based co-clustering in [4].

IV. A Semi-Supervised Approachfor Web Service Discovery on Heterogeneous Evolving Data

We present a semi-supervised heterogeneous evolving frame work for creating service communities. The framework captures the changes in the web services over a period of time. The initial process is to create the relational data matrices, OT and ST. The information about the terms, services and operations is extracted from the WSDL documents [5]. We performed data preprocessing such as removal of stop words and word stemming.

The star-structured schema of operations, terms and services is shown in figure 1. Terms form the central data type with operations and services connected to it. The data may evolve in various ways, such as the number of terms that define a web service may change, or new terms could be added or some existing terms can get removed or the terms that used to define a service in particular domain could now be used to define a new web service in a different domain. We create a relational matrix OTt and STt for every time step. Also as the data is evolving we have to make sure the dimensions of terms in OT_t and ST_t are equal. If the dimensions are different, we use Colibri technique to make the dimensions of the current time step data and the previous time step data equal. If the number of instances in the current time step *t* is greater than the previous time step *t-1* we take the mean of the features and append it to the *t-1* time step data to make the number of instances equal. If we use the deletion approach to make the number of instances equal we are unable to capture the change in the data. Also appending some random instances will cause a cluster drift. In the second scenario where the number of instances in the current time step data is less than the previous time step data we use *Colibri* approach to identify the instances that need to be removed from the previous time step data. When the number features in the current time step is more than the previous time step than we take the mean of the instances and append it to the previous time step data. When the number of features in the current time step is less than the previous time step we use *Colibri* to determine the features that are to be removed from the previous time step data and make the dimensions equal.

We then incorporate the user provided semi-supervised knowledge in the form of pair wise constraints [21] such as must-link (*T*) and cannot-link (*A*) constraints for the current time step data and obtain the new modified relational matrices represented by *B* [22]. Due to the pair wise constraints, the data gets projected in such a way that distance between the data points that belong to the same cluster is reduced and are brought closer to each other [23]. The distance between the data points that belong to the different clusters is maximized. We use the *Colibri* approach to create a low rank approximation of the modified relational matrix *B* which has the user provided knowledge incorporated into it and obtain the compact representation represented by B_t.

Determining the number of clusters in the evolving data is a challenging task. We use four different techniques to determine the number of clusters in current time step data, viz., Krzanowski-Lai [24], Davies-Bouldin index [25], Silhouettes [26],[27] and Calinski-Harabasz index [28]. The number of clusters is determined by taking a majority of the four techniques. Finally we perform the co-clustering using the non-negative matrix factorization and obtain the clustering of services and operations simultaneously represented by *P* and *Q* matrices for each time step *t*.

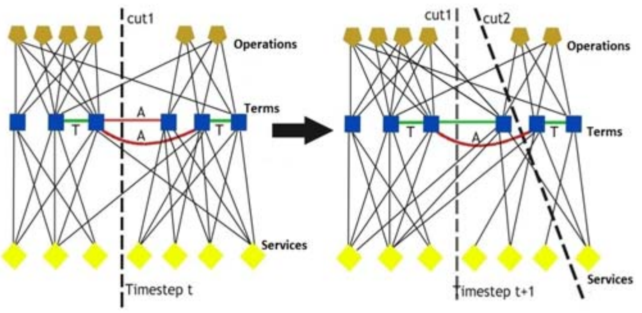


Fig. 1. Semi-supervised co-clustering of heterogeneous start structure relational evolving data at time step t and t+1.

In the star-structured schema, the user provided must-link and cannot-link constraints are placed on the central type, i.e. the terms. As shown in Fig. 1, at current time step t we can see the must link and cannot link constraints that are placed on the data. At time step t+1, of the two clustering choices, the algorithm opts for cut1, so that must link and cannot link constraints are not violated.

In the evolutionary environment we have two costs that determine the cost of the cut J. The first cost is the current snap shot quality of the current clusters represented by f_{sq}. The second cost is the historic cost represented by f_{hc} which tells us how well the current clusters are associated with the historic clusters. So the total cost is defined as the summation of the current and the historic cost which is multiplied by the parameter α.

$$f_{sq} = -\sum_{1\leqslant i\leqslant j\leqslant m} \| \hat{B}_t^{mj} - P_t^m Q_t^{mj} P_t^{(j)T} \|_F^2 \tag{1}$$

$$f_{hc} = \sum_{1\leqslant i\leqslant j\leqslant m} \| \hat{B}_{t-1}^{(mj)} - P_{t-1}^m Q_{t-1}^{mj} P_{t-1}^{(j)T} \|_F^2 \tag{2}$$

$$J_{cost} = -\alpha . f_{sq} + (1-\alpha) . f_{hc} \tag{3}$$

The total cost for the cut is determined from the equation (1.3) above. We use the non-negative matrix tri-factorization approach to obtain the clustering of services and operations simultaneously as represented in the following equations. The P^m and the P^i matrices indicates the services and operations. Q matrix represents the co-relations between the services and terms for all the time steps. The total number of time steps is S where S >1.

$$P_{(S)(ab)}^{(m)} \Leftarrow P_{(S)(ab)}^{(m)} \frac{\sum_{i=1}^{l}((\sum_{t=1}^{S}\alpha(1-\alpha)^{S-t}\tilde{B}_t^{(mi)})P_{(S)}^{(i)^T} Q_{(S)}^{(mi)^T})_{ab}}{\sum_{i=1}^{l}(P_{(S)}^{(m)} Q_{(S)}^{(mi)} P_{(S)}^{(i)} P_{(S)}^{(i)^T} Q_{(S)}^{(mi)^T})_{ab}} \tag{4}$$

$$P_{(S)(ab)}^{(i)} \Leftarrow P_{(S)(ab)}^{(i)} \frac{(Q_{(S)}^{(mi)^T} P_{(S)}^{(m)^T} (\sum_{t=1}^{S}\alpha(1-\alpha)^{S-t}\tilde{B}_t^{(mi)}))_{ab}}{\sum_{i=1}^{l}(Q_{(S)}^{(mi)^T} P_{(S)}^{(m)^T} P_{(S)}^{(m)} Q_{(S)}^{(mi)} P_{(S)}^{(i)})_{ab}} \tag{5}$$

$$Q_{(S)(ab)}^{(mi)} \Leftarrow Q_{(S)(ab)}^{(mi)} \frac{(P_{(S)}^{(m)^T} (\sum_{t=1}^{S}\alpha(1-\alpha)^{S-t}\tilde{B}_t^{(mi)})P_{(S)}^{(i)^T})_{ab}}{\sum_{i=1}^{l}(P_{(S)}^{(m)^T} P_{(S)}^{(m)} Q_{(S)}^{(mi)} P_{(S)}^{(i)} P_{(S)}^{(i)^T})_{ab}} \tag{6}$$

V. Experiments

We performed the experiments using the dataset from [4] which consists of 384 terms, 72 operations, and 97 services. Terms belong to 5 categories, viz., communication, education, food, medical and travel.

In the first experiment, the accuracy is measured in terms of micro-accuracy. Fig. 2 shows the plot of co-clustering accuracy on the evolving data for each time step t0 to *t*7 vs. accuracy with varying the percentage of constraints. The value of α is kept constant at 0.8. At time step t_1, 10 percent of the terms are shifted from one cluster to another. In the next time step t_2, 10 percent of the terms are added to a cluster, at time step t_3, 5 percent of the terms are removed. We can see the co-clustering accuracy is reduced at this time step. But at the next time step i.e. t_4, 10 percent of operations are shifted from one cluster to another and shows the algorithms ability to perform consistent co-clustering even as the data evolves. Also as the percentage of the constraints is increased the co-clustering accuracy is improved as we can see the accuracy for 0, 1, 2, 3 and 4 percent constraints. The constraints are placed on the current data at every time step. As we can see the overall accuracy improves when constraints are placed. The lowest accuracy is when we perform co-clustering without any semi-supervised knowledge. i.e. 0 percent constraint.

Fig. 3 plots the co-clustering accuracy on the evolving web service data where the value of alpha is set to 0.9, 0.5 and 0.2 to measure if the value of alpha impacts accuracy. When the value of alpha is 0.9 it means more importance is given to the current time step data while performing clustering with respect to historic data. The value of alpha equals 0.2 means more importance is given to the historic data while clustering the current time step data and the current time step data will be clustered according to the historic clusters. The value of alpha = 0.5 means equal weights are assigned to the current time step data and the historic time step data to obtain the cluster.

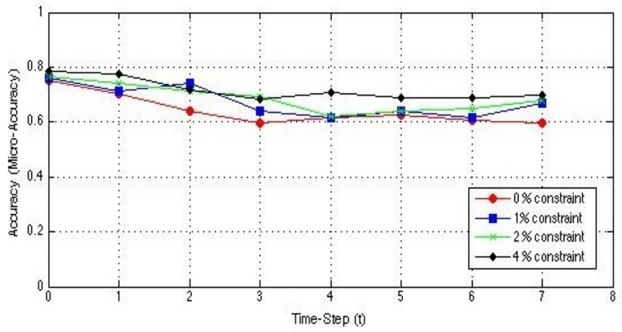


Fig. 2. Semi-supervised co-clustering accuracy is measured by placing 0, 1, 2 and 4 percent constraint on evolving web service dataset from time step t_0 to t_7 where the numbers of instances features are shifted, added or removed from clusters.

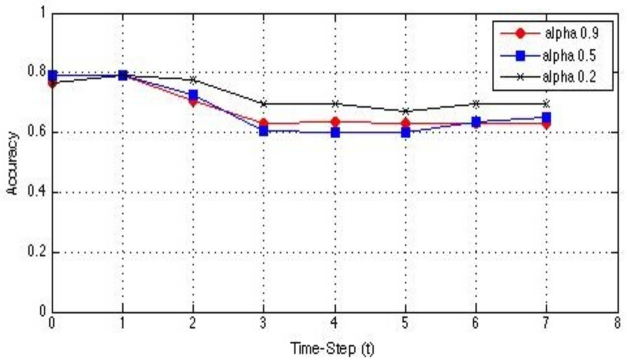


Fig. 3. Measuring the semi-supervised co-clustering accuracy by varying the value of alpha while keeping the constraint percentage constant on the web service dataset.

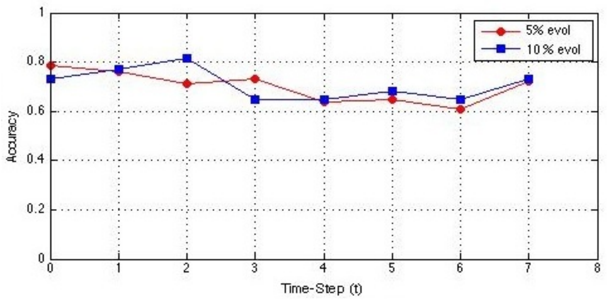


Fig. 4. Comparing the results when the percentage of evolved instances change keeping the constraints and alpha constant on the web service dataset.

In Fig. 4, only 1 percent constraint was placed on the evolving web service data and the value of alpha was kept same. The only change was in the percentage of the instances that evolve. In the first set, only 5 percent of the instances were evolved at each time step t. In the second set, the data was evolved by 10 percent at each time step. As we can see, the co-clustering accuracy is almost similar in both the runs. This implies that the algorithm performs well and provides consistent results and is capable of handling the changes in the evolving data, even when new terms, services or operations are added or removed or shifted from one cluster to another.

This experiment models the real word scenario where the data is evolving and the percentage of instances that are added or removed or shifted can change. The above experiments test the robustness of our proposed approach for creating a evolving service community to enable efficient web service discovery.

VI. Conclusion

We have presented a novel approach to perform semi-supervised co-clustering of heterogeneous evolving web services data. The semi-supervised knowledge is incorporated in the evolving data by placing the must-link and cannot-link constraints on the central data type of the current time step. The new relational matrix is then reduced into a sparse representation using the *Colibri* approach and finally co-clustering is performed using the non-negative matrix factorization technique.

Acknowledgment

We are grateful to the authors of [16] and [6] for providing us with the implementation of Colibri and semi-supervised distance learning, respectively.

References

[1] J. Garofalakis, Y. Panagis, E. Sakkopoulos, and A. Tsakalidis. *Web service discovery mechanisms: Looking for a needle in a haystack*?
[2] K. Elgazzar, A. E. Hassan, and P. Martin, "Clustering wsdl documents to bootstrap the discovery of web services," in *Proc. ICWS '10.*
F. Liu, Y. Shi, J. Yu, T.Wang, and J. Wu, "Measuring similarity of web services based on wsdl," in *Proc. ICWS '10*
[3] Q. Yu and M. Rege, "On service community learning: A co-clustering approach," in *Proc. IEEE International Conference on Web Services*, 2010.
[4] Amit Salunke, Minh Nguyen, Xumin Liu, and Manjeet Rege. Web service discovery using semi-supervised block value decomposition. *In IRI*, pp. 36–41, 2011.
[5] Y. H. Chen, L. J. Wang, and M. Dong, "Non-negative matrix factorization for semisupervised heterogeneous data coclustering," *IEEE Trans. on Knowl. and Data Eng.*, 22, pp.1459–1474, October 2010.
[6] B. Gao, Tie-Yan Liu, X. Zheng, "Qian-Sheng Cheng, and Wei-Ying Ma, Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data coclustering," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, KDD '05*, pp. 41–50, New York, NY, USA, 2005. ACM.
[7] M. Rege, M. Dong, and F. Fotouhi, "Co-clustering documents and words using bipartite isoperimetric graph partitioning," in *Proceedings of the Sixth International Conference on Data Mining, ICDM '06*, pp. 532–541, Washington, DC, USA, 2006. IEEE Computer Society.
[8] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '06*, pp. 554–560, New York, NY, USA, 2006. ACM.
[9] Y. Chi, X. D. Song, D. Y. Zhou, K. Hino, and B. L. Tseng, "Evolutionary spectral clustering by incorporating temporal smoothness," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '07*, pp. 153–162, New York, NY, USA, 2007. ACM.

[10] X. Liu N. Green, M. Rege and R. Bailey, "Evolutionary spectral co-clustering," in *Proc. International Joint Conference on Neural Networks,* 2011.

[11] R. Bekkerman and M. Sahami, "Semi-supervised clustering using combinatorial mrfs," in *ICML-06 Workshop on Learning in Structured Output Spaces*, 2006.

[12] Y. H. Chen, M. Rege, M. Dong, and J. Hua, "Non-negative matrix factorization for semi-supervised data clustering," *Knowl. Inf. Syst.,* 17, pp. 355–379, November 2008.

[13] X. Ji and W. Xu, "Document clustering with prior knowledge," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '06*, pp. 405–412, New York, NY, USA, 2006.ACM.

[14] B. Long, Z. F. Zhang, and P. S. Yu, "Co-clustering by block value decomposition," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, KDD '05,* pp. 635–640, New York, NY,USA, 2005. ACM

[15] H. H. Tong, S. Papadimitriou, J. Sun, P. S. Yu, and C. F. Colibri, "fast mining of large static and dynamic graphs," in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '08*, pp. 686–694, New York, NY, USA, 2008. ACM.

[16] M. W. Berry, S. A. Pulatova, and G. W. Stewart, "Algorithm 844: Computing sparse reduced-rank approximations to sparse matrices," *ACM Trans. Math. Softw.*, 31, pp. 252–269, June 2005.

[17] P. Drineas, R. Kannan, M. W. Mahoney, and Let A., "Fast monte carlo algorithms for matrices iii: Computing a compressed approximate matrix decomposition," *SIAM Journal on Computing*, vol. 36, pp. 2006, 2004.

[18] J. M. Sun, Y. L. Xie, H. Zhang, and C. Faloutsos, "Less is more: Compact matrix decomposition for large sparse graphs," in *Proc. SIAM Intl. Conf. Data Mining,* 2007.

[19] L. J. Wang, M. Rege, M. Dong, and Y. S. Ding, "Low-rank kernel matrix factorization for large scale evolutionary clustering," *IEEE Transactions on Knowledge and Data Engineering,* 99(PrePrints), 2010.

[20] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, "Mixed membership stochastic blockmodels," *The Journal of Machine Learning Research,* 9, pp.1981–2014, 2008.

[21] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha, "A generalized maximum entropy approach to bregman co-clustering and matrix approximation," *J. Mach. Learn. Res.*, 8, pp. 1919–1986, December 2007.

[22] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell, "Distance metric learning, with application to clustering with side-information," in *Advances in Neural Information Processing Systems* 15, pp. 505–512. MIT Press, 2002.

[23] W. J. Krzanowski and Y. T. Lai, "A criterion for determining the number of groups in a data set using sum-of-squares clustering," *Biometrics*, pp. 23–34, 1988.

[24] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li, "Rcv1: A new benchmark collection for text categorization research," *J. Mach. Learn.* Res., 5, pp. 361–397, December 2004

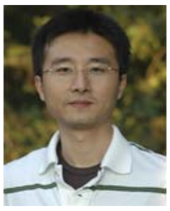
[25] P. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math*., vol. 20, no.1, pp. 53–65, November 1987.

[26] K. Tasdemir and E. Merenyi, "A new cluster validity index for prototype based clustering algorithms based on inter- and intra-cluster density," in *Proc. International Joint Conference on Neural Networks (IJCNN 2007),* 2007.

[27] Y. Peng, Y. Zhang, G. Kou, and Y. Shi, "A multicriteria decision making approach for estimating the number of clusters in a data set," *PLoS ONE*, vol. 7, no. 7, pp.e41713,07 2012.

Pankaj Andhale is a data engineer at Intuit, Inc. specializing in building scalable, cost effective data processing platforms. He uses data mining techniques to measure the success of products that will help make better data driven business decisions. He holds a Masters in Computer Science from Rochester Institute of Technology, and has also worked previously at Cognizant Technology Solutions.

Manjeet Rege is currently an assistant professor in the College of Computing and Information Sciences at Rochester Institute of Technology. He received the PhD in Computer Science from Wayne State University, Detroit, MI. His research interests lie in the areas of Data Mining, Multimedia Analysis, and Information Retrieval. Dr. Rege's research has been published in various international conferences and journals such as IEEE International Conference on Data Mining (ICDM), ACM International Conference on Multimedia (ACM MM), World Wide Web Conference (WWW), IEEE International Conference on Image Processing (ICIP), ACM Multimedia Systems Journal, Data Mining and Knowledge Discovery Journal, and the Knowledge and Information Systems Journal. He also serves regularly on the program committees of various international conferences and workshops.



Qi Yu received the PhD degree in computer science from Virginia Polytechnic Institute and State University (Virginia Tech). He is an assistant professor at the college of computing and information sciences of Rochester Institute of Technology. His current research interests lie in the areas of service computing, data management, and analytics. His publications have mainly appeared in well-known journals (e.g. the VLDB journal and ACM Transactions on the Web) and conference proceedings (e.g., ICSOC and ICWS). He is a guest editor of the IEEE Transactions on Services Computing special issue on service query models and efficient selection. He frequently serves as a program committee member on service computing and database conferences (e.gSOCA, CollaborateCom, IRI, ICSOC, and APSCC). He is also a reviewer for various journals (e.g., the VLDB Journal, ACM Transactions on the Web, and IEEE Transactions on Service Computing). He is a member of the IEEE.