MapReduce for Big Data Applications using Keyword-Aware Service Recommendation Method

Geeta Y Talawar
Department of computer science
T. John Institute of Technology

Annie Sujith
Assistant Professor
Department of computer science
T. John Institute of Technology

***Abstract*— Service recommender systems has been shown as valuable tools. It also provides appropriate recommendations to users, now a days services and online information has been grown rapidly, yielding the big data analysis problem, Here Hadoop is an open source tool mainly used for big data analysis, but it can be used for developing searching applications, in this project we will use user based collaborative filtering algorithm in order to generate appropriate recommendations, in this project we also introduce keyword aware service recommendation method ,named KASR in order to address the problems that can be occurred during big data analysis, In order to improve scalability and efficiency in big data environment, KASR is implemented on Hadoop. Hadoop is mainly widely adopted distributed computing platform using the Map reduce parallel processing paradigm, finally the results or experiments demonstrate that KASR significantly improves the accuracy and scalability of service recommender systems over existing approaches.**

Keywords*— *MapReduce, Hadoop, Preference, Big Data, Recommender System. Cloud Computing

1.INTRODUCTION

This paper proposes the **Keyword-Aware Service Recommendation method**, named **KASR**, to address problem in performance and problems when processing or huge-**scale data (bigdata)**.It aims at presenting a **personalized service** recommendation list and recommending the most appropriate services to the users effectively. It uses**Mapreduce technique** to handle bigdata. To achieve **scalability** in service recommendation based on keyword aware. To solve inefficiency in handling the large amount services during recommendation This paper may contribute some techniques that tells how to deal with the case where term appears in different categories of a **domain thesaurus** from context and how to distinguish the **positive and negative preferences** of the users from their reviews to make the predictions more accurate.Service recommender systems have been shown as valuable tools for providing appropriate recommendations to users. In the last decade, the amount of customers, services and online information has grown rapidly, yielding the big data analysis problem for service recommender systems. Consequently, traditional service recommender systems often suffer from scalability and inefficiency problems when processing or analysing such large-scale data. Moreover, most of existing service recommender systems present the same ratings and rankings of services to different users without considering diverse users' preferences, and therefore fails to meet users' personalized requirements. In this paper,

we introduce a Keyword-Aware Service Recommendation method, named KASR, to address the above problems. It works at providing a personalized service recommendation list and guides the most appropriate services to the users effectively. Specifically, keywords are used to indicate users' requirements, and a user-based Collaborative Filtering algorithm is used to generate appropriate recommendations.

To improve its scalability and efficiency in big data environment, KASR is developed on Hadoop, a commonly-adopted distributed computing platform using the MapReduce parallel processing paradigm. Finally, extensive experiments are conducted on real-world data sets, and results demonstrate that KASR significantly improves the accuracy and scalability of service recommender systems over existing approaches.

Services shown as important tools to help users deal with services overload and provide appropriate recommendations to them. Examples of such practical applica-tions include CDs, books, web pages and various other products now use recommender systems [5], [6], [7]. Over A secret polynomial based message authentication scheme was introduced in [3] to solve the scalability problem, where the threshold here is determined by the degree of the polynomial. When the messages transmitted is below the threshold, the nodes verify the authenticity of the message through a polynomial evaluation. When the messages transmitted is above the threshold, the polynomial is recovered and the system is completely broken.

2.TERMINOLOGY AND PRELIMINARY

A. Terminology

1)Recommender Systems and Collaborative Filtering
Recommender systems developed as an independent re-search area in the mid-1990s when recommendation prob-

lems 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) approach-es 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 re-commender systems [13]. In CF based systems, users re-ceive 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. More details of user-based CF algorithm can be found in Appendix A.1.

2) Cloud Computing and Map Reduce

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/), Map Reduce of Google [15], the Dynamo of Amazon.com [16], the Dryad of Microsoft and Neptune of Ask.com [17], etc.

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].

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. More details about Ma-pReduce can be found in Appendix A.2. Nowadays, the trend "everything as a service" has been creating a Big Services era due to the foundational architec-ture of services computing. And "servicelization" is the way of offering social networking services, big data analyt-ics, and Internet services [19] [20]. Thus the cloud compu-ting tools aforementioned can be used to improve the sca-lability and efficiency of service recommendation methods in the "Big Data" environment.

3.KEYWORD-AWARE SERVICE RECOMMENDATION METHOD

Definition 1 (Keyword-Aware Service Recommendation

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 moreover to to improve the saclability and efficiency of our recommendation method.

TABLE 1

Basic symbols and notations

Symbol	Definition
K	The keyword-candidate list={k1,k2,..kn}
APK	The preference keyword set of active user
PPK	The preference keyword set of previous user
Sim(APK,PPK)	The similarity between APK and PPK
Wp	A preference weight vector
Wap	The preference weight vector of active user
Wpp	The preference weight vector of a previous user

Here Table1 summarizes the basic symbols and notations Used in this paper.

3.1) Keyword-candidate list and domain thesaurus

In this paper two methods ,two data structures, keyword candidate list and specialized domain thesaurus are introduced to help obtain users preferences

Definition 2 (Keyword-candidate list)

The keyword candidate list is a set of keywords about users preferences and multi-criteria of candidate services, which can be denoted as K{k1,k2,…Kn},n is the number of the keywords in the keyword-candidate list.

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 can not exactly match the corresponding keywords.

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		

Definition 3 (Domain thesaurus).A domain thesaurus is a reference work of the keyword candidiate list that lists words grouped together according to the similarity of keyword meaning, including related and contrasting words and antonyms[21][22].

3.2 A keyword-aware service recommendation method

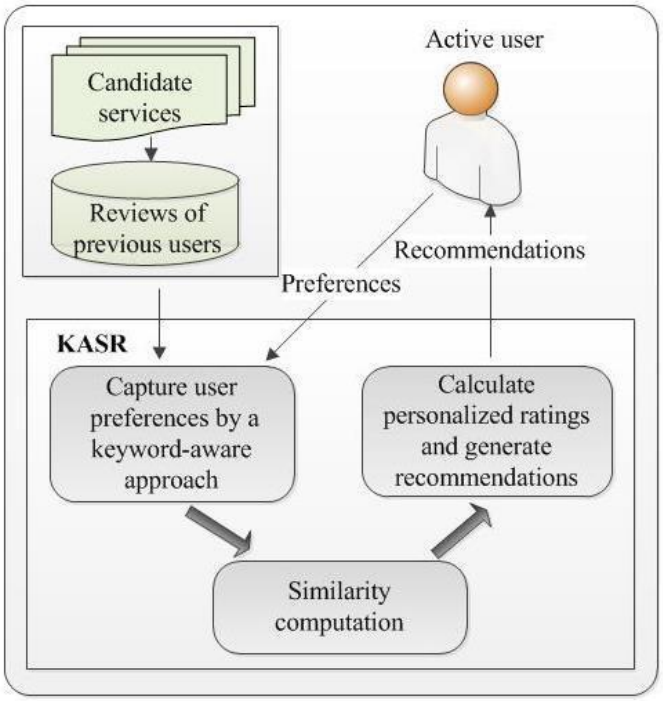
The main steps of KASR are depicted in Fig. 3, which are described in detail as follows.

(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 de



denoted as,APK={ak1,ak2,…akl}, where ak1(1≤ i≤ l) where is the *i*th 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

Measurment	General	Important	Very Important
Importance degree	1	2	3

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, where is the *i*th keyword extracted from the review, *h* is the number of ex-tracted 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 af-fecting 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 prefe-rence keyword set of the user. For example, if a review of a previous user for a hotel has the word "spa", which is cor-responding to the keyword ``Fitness" in the domain thesau-rus, 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.

Algorithm 2: SIM-ESC (Exact Similarity Computation)

Input: The preference keyword set of the active user *APK*
The preference keyword set of a previous user PPK_j
Output: The similarity of *APK* and PPK_j, $sim_{ESC}(APK, PPK_j)$
1: for each keyword k_i in the keyword-candidate list
2: if $k_i \in APK$ **then**
3: get $\vec{W}_{AP,i}$ by formula (2)
4: else $W_{AP,i} = 0$
5: end if
6: if $k_i \in PPK_j$ **then**
7: get $W_{PP_j,i}$ by formula (5)
8: else $W_{PP_j,i} = 0$
9: end if
10: end for
11: get $sim_{ESC}(APK, UPK_j)$ by formula (6)
12: return the similarity of *APK* and PPK_j, $sim_{ESC}(APK, UPK_j)$

.

4. IMPLEMENTATION ON MAPREDUCE

Many systems have provided restricted programming models and used the restrictions to parallelize the computation automatically. For example, an associative function can be computed over all pre_xes of an N element array in logN time on N processors using parallel pre_x computations. MapReduce can be considered a simpli_cation and distillation of some of these models based on our experience with large real-world computations.

More signi_cantly, we provide a fault-tolerant implementation that scales to thousands of processors. In contrast, most of the parallel processing systems have only been implemented on smaller scales and leave the details of handling machine failures to the programmer. Bulk Synchronous Programming and some MPI primitives provide higher-level abstractions that make it easier for programmers to write parallel programs. A key difference between these systems and MapReduce is that MapReduce exploits a restricted programming model to parallelize the user program automatically and to provide transparent fault-tolerance. Our locality optimization draws its inspiration from techniques such as active disks [12, 15], where computation is pushed into processing elements that are close to local disks, to reduce the amount of data sent across I/O subsystems or the network. We run on commodity processors to which a small number of disks are directly connected

instead of running directly on disk controller processors, but the general approach is similar

(1) KASR-ASC on MapReduce Fig. shows the computation flowchart of KASR-ASC on MapReduce, which consists of three steps. And Step 1 is offline executed, Step 2 and Step 3 are online executed.

Step 1: The first step is to process the reviews for candidate services by previous users into their preference keyword sets and compute the average ratings for each candidate service. Map-I: Map on *i* such that the tuples with the same *i* are shuffled to the same node in the form of. Reduce-I: Take as the input and emit for each input of Map-I. The output of Reduce-I will be used as the input of Map-II to calculate the similarity.

Step 2: The second step is to compute the similarity be-tween the active user and previous users. Map-II: Map on *i*, and tuples with the same *i* are shuf-fled to the same node in form of. Reduce-II: Take <*APK*> and as the input, then emit.

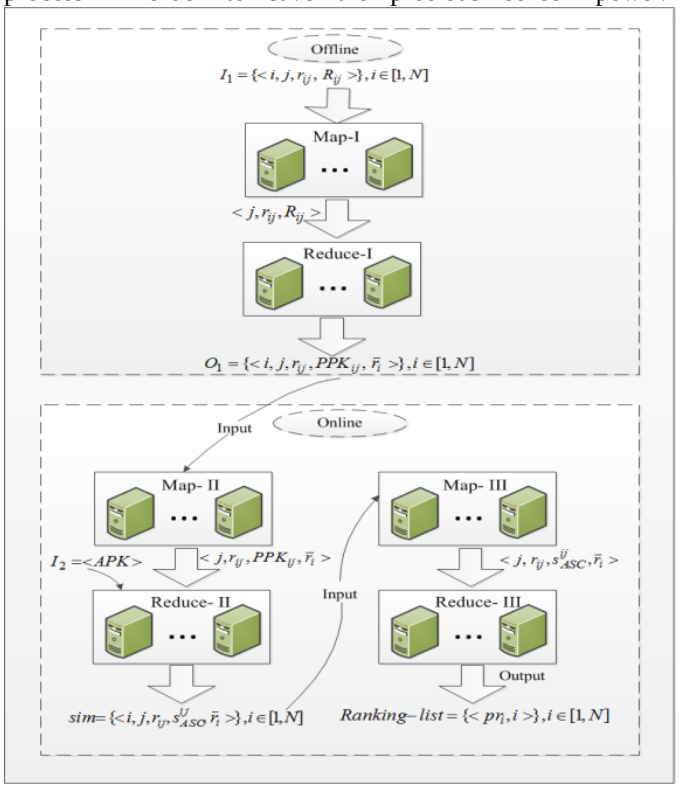
Step 3: The third step aims to calculate the personalized rating of each candidate service and present a personalized recommendation list to the active user. Based on the output of this step, the recommendation can be obtained. Map-III:

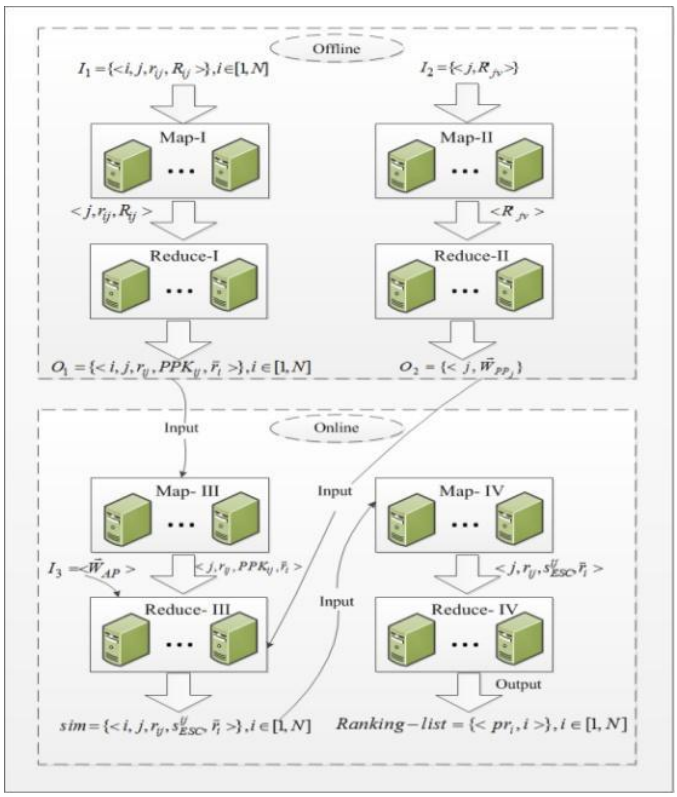
Map on *i* so that the tuples with the same *i* are shuffled to the same node in form of. Reduce-III: Take as the input, and emit, where *pri* is the persona-lized rating of the active user to service *i*. The tuples of the output are ordered by the services id *i*, which is just the personalized service recommendation list to the active user compromised nodes to be identified and captured. When any node identified as compromised, the SS can then remove its public key from its public key list.

It can also broadcast node's short identity to the entire sensor domain so that any sensor node that uses the stored public key for an AS selection can update its key list. Once the public key of a node has been removed from the public key.

(2) KASR-ESC on MapReduce

Fig. shows the comparision flowchart of KASR-ESC public key list, and/or broadcasted, any message with AS containing the compromised node should be dropped The MapReduce programming model has been successfully used at Google for many different purposes. We attribute this success to several reasons. First, the model is easy to use, even for programmers without experience with parallel and distributed systems, since it hides the details of parallelization, fault-tolerance, locality optimization, and load balancing. withoutt any process in order to save the precious sensor power.





5. EXPERIMENT SETUP AND DATASETS

Technically, our experiments are conducted in a Hadoop platform. And to evaluate the accuracy and scalability of KASR, two kinds of dataset are adopted in the experiments: a real dataset and a synthetic dataset. Due to the space limit, more details about the experiment settings and dataset can be found in Appendix C.2 and Appendix C.3, respectively.

Experiment Evaluation

Two groups of experiments are conducted to evaluate the accuracy and scalability of KASR. In the first one, we compare KASR with UPCC and IPCC in MAE, MAP and DCG to evaluate the accuracy of KASR. The other is to explore the scalability of KASR. **5.2.1 Accuracy evaluation**

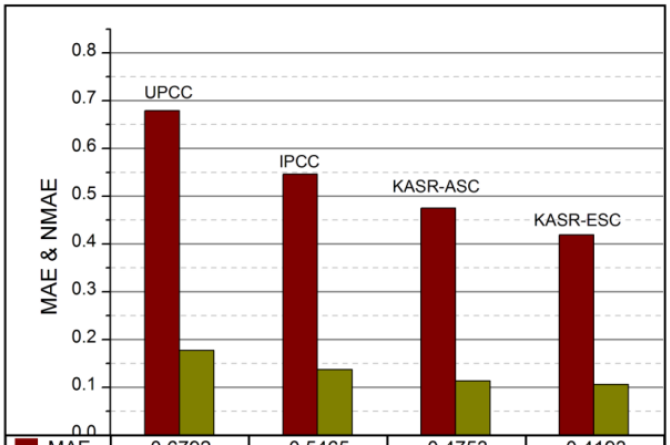
Comparison of UPCC, IPCC, KASR-ASC and KASR-ESC in MAE

and NMAE vaules of KASR-ASC and KASR-ESC are much lower than UPCC and IPCC (e.g., the MAE and NMAE values

MAE is a statistical accuracy metric often used in CF methods to measure the prediction quality.

Fig. 5 shows the MAE and NMAE values of UPCC, IPCC, KASR-ASC and KASR-ESC. It could be found that the MAE of KASR-ASC are respectively 30.02% ((0.6792-0.4753) / 0.6792 = 30.02%) and 35.73% lower than UPCC. And the Norma-lized Mean Absolute Error (NMAE) metric is also used to measure the prediction accuracy.

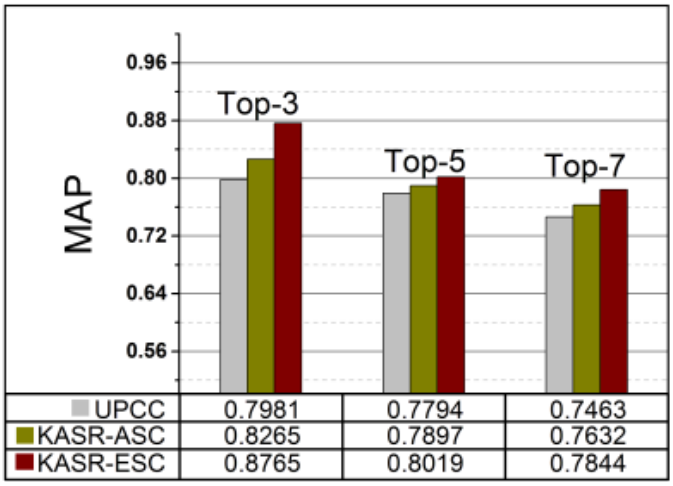
And the MAE and NMAE values of KASR-ESC are The lower the MAE or NMAE presents the more accurate predictions.



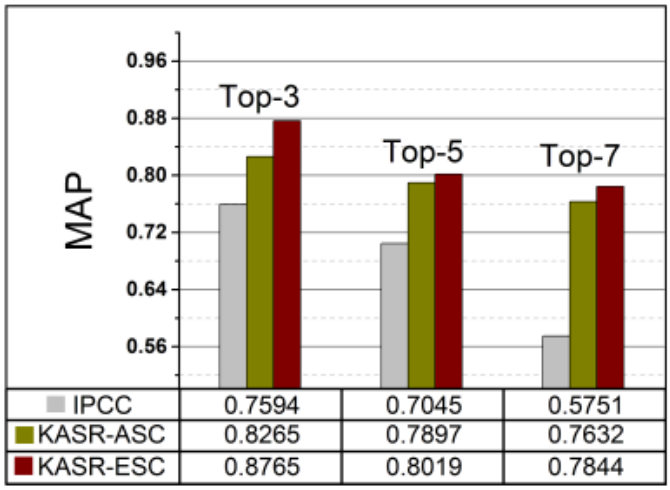
MAE	0.6792	0.5465	0.4753	0.4193
NMAE	0.1769	0.1373	0.1137	0.1059

Comparison of UPCC, IPCC, KASR-ASC and KASR-ESC in MAP and DCG

To evaluate the quality of Top-K service recommendation list MAP and DCG are used as performance evaluation metricsThe services in higher position, especially the nfirst position, should be more satisfying than the services in lower posi-tion of the returned result list.. And the higher MAP or DCG presents the higher quality of the predicted. In most service recommender systems, users tend to be recommended the top services of the returned result list.



UPCC	0.7981	0.7794	0.7463
KASR-ASC	0.8265	0.7897	0.7632
KASR-ESC	0.8765	0.8019	0.7844



IPCC	0.7594	0.7045	0.5751
KASR-ASC	0.8265	0.7897	0.7632
KASR-ESC	0.8765	0.8019	0.7844

6.CONCLUSION

In this paper, we have proposed recommendation method, named KASR. In KASR, key-words are used to indicate users' ratings&rankings, and a user-based Collaborative Filtering algorithm is used to provide services. More specifically, a keyword-candidate list and domain thesaurus are provided to help obtain users' reqiurements. The active user gives his/her requirement by selecting the keywords from the keyword-candidate set, and the ratings and rankings of the previous users can be collected from their reviews for services ac-cording to the keyword-candidate list and domain thesau-rus. Our method aims at presenting a personalized service recommendation list and recommending the most appro-priate service(s) to the users. Moreover, to improve the working capacity and efficiency of KASR in “Big Data” environment, we have developed it on a MapReduce framework in Hadoop platform. Finally, positive and negative preferences of the users from their reviews to make the predictions more accurate.

REFERENCES

[1] F. Ye, H. Lou, S. Lu, and L. Zhang, "Statistical en-route filtering of injected false data in sensor networks," in IEEE INFOCOM, March 2004.

[2] S. Zhu, S. Setia, S. Jajodia, and P. Ning, "An interleaved hop-by-hop authentication scheme for filtering false data in sensor networks," in IEEE Symposium on Security and Privacy, 2004.

[3] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences,"in Advances in Cryptology - Crypto'92, . [4] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Communications. of the Assoc. of Comp. Mach., vol. 21, no. 2, pp. 120–126, 1978.

[5] T. A. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions on Information Theory, vol. 31, no. 4, pp. 469–472, 1985.

[6] H. Wang, S. Sheng, C. Tan, and Q. Li, "Comparing symmetric-key and public-key based security schemes in sensor networks: A case study of user access control," in IEEE ICDCS, Beijing, China, 2008, pp. 11–18.

[7] D. Pointcheval and J. Stern, "Security proofs for signature schemes," in Advances in Cryptology - EUROCRYPT, ser. Lecture Notes in Computer Science Volume 1070, 1996, pp. 387–398.

[8] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," Communications of the ACM, vol. 24, no. 2, pp. 84–88, February 1981.

[9] "The dinning cryptographer problem: Unconditional sender and recipient untraceability," Journal of Cryptology, vol. 1, no. 1, pp. 65–75,1988.

[10] A. Pfitzmann and M. Hansen, "Anonymity, unlinkability, unobservabil- ity, pseudonymity, and identity management a proposal for terminol- ogy," http://dud.inf.tu-dresden.de/literatur/Anon Terminology v0.31.pdf, Feb. 15 2008.

[11] A. Pfitzmann and M. Waidner, "Networks without user observability–design options." in Advances in Cryptology - EUROCRYPT, ser. Lecture Notes in Computer Science Volume 219, 1985, pp. 245–253.

[12] M. Reiter and A. Rubin, "Crowds: anonymity for web transaction," ACM Transactions on Information and System Security, vol. 1, no. 1, pp. 66–92,1998.

[13] M. Waidner, "Unconditional sender and recipient untraceability in spite of active attacks," in Advances in Cryptology - EUROCRYPT, ser. Lecture Notes in Computer Science Volume 434, 1989, pp. 302–319.

[14] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," Journal of Cryptology, vol. 13, no. 3, pp. 361–396, 2000.

[15] L. Harn and Y. Xu, "Design of generalized ElGamal type digital signature schemes based on discret logarithm," Electronics Letters, vol. 30, no. 24, pp. 2025–2026, 1994.

[16] K. Nyberg and R. A. Rueppel, "Message recovery for signature schemes based on the discrete logarithm problem," in Advances in Cryptology - EUROCRYPT, ser. Lecture Notes in Computer Science Volume 950, 1995, pp. 182–193.

[17] R. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in Advances in Cryptology–ASIACRYPT, ser. Lecture Notes in Computer Science, vol 2248/2001. Springer Berlin / Heidelberg[18] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in CCS'93, 1993, pp. 62–73.

[19] BlueKrypt, "Cryptographic key length recommendation," http://www.keylength.com/en/3/.

[20] W. Zhang, N. Subramanian, and G. Wang, "Lightweight and compromise resilient message authentication in sensor networks," in IEEE INFOCOM, Phoenix, AZ., April 15-17 2008.

[21] A. Perrig, R. Canetti, J. Tygar, and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," in IEEE Symposium on Security and Privacy, May 2000., 2001.