GRAPH RANKING-BASED RECOMMENDER SYSTEMS

A thesis submitted for the degree of

Doctor of Philosophy

By

Mingsong Mao

December, 2015

University of Technology Sydney,

Faculty of Engineering and Information Technology

CERTIFICATE OF ORIGINAL AUTHORSHIP

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Candidate:

Production Note:
Signature removed
prior to publication.

Acknowledgement

"A teacher is one who could propagate the doctrine, impart professional knowledge, and resolve doubts." *Han Yu, 768AD*

I would like to express my earnest thanks to my supervisors, Professor *Jie Lu* and Professor *Guangquan Zhang*, who have provided tremendous support and guidance for my research in the past four years. *Jie* provided me an opportunity to study in the stimulating and interactive lab of Decision Systems & e-Service Intelligence (DeSI) of the centre for Quantum Computation and Intelligent Systems (QCIS), where I met and leant a lot from many smart people. I have benefited significantly from her unselfish help and invaluable suggestions on my research career. I also would like to thank *Guangquan* for his continuous guidance and sharp discussions during my four years study. Without their endless patience, generous support, and constant guidance, this thesis could not have been accomplished.

"In a group of three people, there is always something I can learn from."
Confucius, 551BC

I would like to thank all the people that had a positive influence on my day-to-day enjoyment during the four years study and life in Sydney. The DeSI lab members, Dr *Vahid Behbood*, Dr *Dianshuang Wu*, Dr *Mohshen Naderpour*, Ms *Hongshu Chen*, Mr *Wei Wang*, Mr *Junyu Xuan*, Mr *Fan Dong*, Mr *Jialin Han*, Mr *Peng Hao*, Mr *Anjin Liu*, Ms *Hua Zuo*, Mr *Yi Zhang*, Ms *Qian Zhang* and former visiting scholars

Dr *Yue Zheng* and Dr *Zheng Zhang* have given me support during both joyful and stressful times, to whom I will always be thankful. I also thank all the colleagues in the centre of QCIS. They are always easy to get along with and willing to share knowledge and experience, and some of them have become my lifelong friends.

I also wish to express my appreciation to the financial support I gained for my study. Special thanks go to China Scholarship Council (CSC) and University of Technology Sydney (UTS). I would like to express my since thanks to Ms. Sue Felix and John Hazelton for helping me to correct English presentation problems in my publications and this thesis.

> *"Filial piety and brotherly obedience are surely the roots of humaneness."*
>
> *Confucius, 551BC*

Last but not least, I want to thank my family for their continuous support. I especially thank my wife, *Fang Yang*, and our son, the little gentleman *Yang Mao*, for bringing lots of laughter at the most difficult period of my Ph.D. study. I would like to thank my parents and younger sister for their unconditional encouragement and support, both emotionally and financially.

Coincidentally when I am writing this, tomorrow is the second birthday of Yang Mao and this dissertation is dedicated to him as a gift.

Mingsong Mao
28th October 2015
At Singapore

Abstract

The rapid growth of web technologies and the volume of Internet users provide excellent opportunities for large-scale online applications but also have caused increasing information overloading problems whereby users find it hard to locate relevant information to exactly meet their needs efficiently by basic Internet searching functions. Recommender systems are emerging to aim to handle this issue and provide personalized suggestions of resources (items) to particular users, which have been implemented in many domains such as online shopping assistants, information retrieval tools and decision support tools. In the current era of information explosion, recommender systems are facing some new challenges. Firstly, there are increasing tree-structured taxonomy attributes as well as freeform folksonomy tags associated with items. Secondly, there are increasing explicit and implicit social relations or correlations available for web users. Thirdly, there is increasingly diverse contextual information that affects or reflects user preferences. Furthermore, the recommendation demands of users are becoming diverse and flexible. In other words, users may have changing multi-objective recommendation requests at different times.

This research aims to handle these four challenges and propose a set of recommendation approaches for different scenarios. Graph ranking theories are employed due to their ease of modelling different information entities and complex relations and their good extensibility. In different scenarios, different graphs are generated and some unique graph ranking problems are raised. Concretely, we first propose a bipartite graph random walk model for a hybrid recommender system integrating complex item content information of both tree-structured taxonomy attributes and free-form folksonomy tags. Secondly, we propose a multigraph ranking model for a multi-relational social network-based recommendation system that is able to incorporate multiple types of social relations or correlations between users. Thirdly, we propose a multipartite hypergraph ranking model for a generic full information-based recommender system that is able to handle various parities of information entities and their high-order relations. In addition, we extend the multipartite hypergraph ranking model to be able to respond to users' multi-objective recommendation requests and propose a novel multi-objective recommendation framework.

We conduct comprehensive empirical experiments with a set of real-word public datasets in different domains such as movies (Movielens), music (Last.fm), e-Commerce products (Epinions) and local business (Yelp) to test the proposed graph ranking-based recommender systems. The results demonstrate that our models can generally achieve significant improvement compared to existing approaches in terms of recommendation success rate and accuracy. By these empirical experiments, we can conclude that the proposed graph ranking models are able to handle well the indicated four key challenges of recommender systems in the current era. This work is hence of both theoretical and practical significances in the field of both graph ranking and recommender systems.

Table of Contents

Acknowledgement **i**

Abstract **iii**

Table of Contents **v**

List of Figures **xi**

List of Tables **xiii**

CHAPTER 1. Introduction **1**

1.1 Background ······ 1

1.2 Research Questions ······ 5

1.3 Research Objectives ······ 7

1.4 Significance ······ 11

1.4.1 Theoretical Significance ······ 11

1.4.2 Practical Significance ······ 12

1.5 Research Methodology and Process ······ 13

1.5.1 Research Methodology ······ 13

1.5.2 Research Process ······ 14

1.6 Thesis Structure and Publications ······ 16

1.7 Basic Notations in This Thesis ······ 17

CHAPTER 2. Literature Review 19

2.1 Classical Recommender Systems 19

2.1.1 Collaborative Filtering-based Recommender Systems 20

2.1.2 Content-based Recommender Systems 23

2.1.3 Knowledge-based Recommender Systems 24

2.1.4 Computational Intelligence-based Recommender Systems 26

2.1.5 Hybrid Recommender Systems 28

2.2 The State-of-the-Art Recommendation Techniques 29

2.2.1 Complex Content-based Recommendation Techniques 30

2.2.2 Social Network-based Recommendation Techniques 31

2.2.3 Graph Ranking-based Recommendation Techniques 36

2.2.4 Context-aware Recommendation Techniques 38

2.2.5 Multi-Criteria and Group Recommendation Techniques 40

2.3 Applications of Recommender Systems 41

2.3.1 Recommender Systems as Online Shopping Assistants 41

2.3.2 Recommender Systems as Information Retrieval Tools 43

2.3.3 Recommender Systems as Online Decision Support Tools 44

2.4 Evaluation Criteria for Recommender Systems 46

2.4.1 Recommendation Success Rate 47

2.4.2 Rating Prediction Accuracy 48

2.4.3 Classification Accuracy 49

2.4.4 Item Ranking Accuracy 50

2.5 Summary 51

CHAPTER 3. A Hybrid Recommender System via Bipartite Graph Random

Walks 53

3.1 Overview ········ 53

3.2 User-item Bipartite Random Walk Model ········ 56

3.2.1 Performing Random Walks ········ 57

3.2.2 Graph Ranking and Recommendation Making ········ 60

3.3 Representations of Item Taxonomy and Folksonomy ········ 63

3.3.1 Taxonomy Attributes ········ 63

3.3.2 Folksonomy Tags ········ 65

3.3.3 Tag-based Semantic Similarity of Attributes ········ 68

3.4 Computing Item Content Similarity ········ 70

3.4.1 Generating Weightings from Tags ········ 71

3.4.2 Subtree Matching Algorithm ········ 72

3.4.3 Overall Content similarity ········ 76

3.5 A Numeric Example ········ 77

3.5.1 Item Overall Content Similarity ········ 77

3.5.2 Recommendations based on Bipartite Random Walks ········ 79

3.6 Experiments ········ 80

3.6.1 Experiment Setup ········ 80

3.6.2 Performance Comparison ········ 82

3.6.3 Parameter Optimization ········ 84

3.7 Summary ········ 87

CHAPTER 4. A Social Network-based Recommender System via Multigraph Ranking 89

4.1 Overview ········ 89

4.2 Single Social Network Propagation ······· 91

4.2.1 Performing a Single Random Walk ······· 92

4.2.2 Performing More Random Walks ······· 94

4.3 Multi-Relational Social Networks ······· 98

4.3.1 Multigraph Generation ······· 98

4.3.2 Inter-Network Comparison ······· 100

4.4 Multigraph Ranking and Recommendations ······· 103

4.4.1 Simple Graph Ranking ······· 105

4.4.2 Multigraph Ranking Problem ······· 107

4.4.3 Making Recommendations ······· 111

4.5 Experiments ······· 112

4.5.1 Experiment Setup ······· 112

4.5.2 Performance Comparison ······· 116

4.5.3 Parameter Adjusting ······· 121

4.6 Summary ······· 123

CHAPTER 5. A Unified Recommender System via Multipartite Hypergraph Ranking 125

5.1 Overview ······· 125

5.2 Hypergraph and Hypergraph Ranking ······· 127

5.2.1 Notations of Hypergraphs ······· 127

5.2.2 Hypergraph Ranking ······· 129

5.2.3 Hypergraph Random Walks ······· 130

5.3 The Data Model and Multipartite Hypergraph ······· 131

5.3.1 Information Entities ······· 132

5.3.2 Possible Relations between Different Objects 134

5.3.3 Multipartite Hypergraph Construction 137

5.4 Balanced Hypergraph Ranking and Recommendations 139

5.4.1 Balanced Hypergraph Ranking Model 139

5.4.2 Comparison with Traditional Models 141

5.4.3 Query Vector and Recommendation 143

5.5 Experiments 144

5.5.1 Data Sampling 144

5.5.2 Compared Approaches 146

5.5.3 Evaluation Metrics 147

5.5.4 Performance Comparison 148

5.6 Summary 151

CHAPTER 6. A Multi-Objective Recommender System via Hypergraph Ranking 153

6.1 Overview 153

6.2 Multi-objective Request Analysis 155

6.3 Multi-objective Recommendation Framework 157

6.4 FoodGo: A Multi-objective Restaurant Recommender System 159

6.4.1 Multi-objective Request Collection 159

6.4.2 Background Model Training 161

6.4.3 Recommendation Display 161

6.5 Empirical Evaluation 162

6.5.1 Generating Multi-objective Requests for Testing 162

6.5.2 Evaluation Method 163

6.5.3 Performance Comparison ··········164

6.6 Summary ··········164

CHAPTER 7. Conclusions And Future Study 167

7.1 Conclusions ··········167

7.2 Future Studies ··········171

References 173

List of Figures

Figure 1-1 This research raises and solves three unique graph ranking problems..... 11

Figure 1-2 The proposed research methodology of this thesis 13

Figure 1-3 The structure and contents of this thesis 15

Figure 3-1 Random walking on a user-item bipartite graph 55

Figure 3-2 An example of movie taxonomy tree in MovieLens 64

Figure 3-3 An example of book taxonomy tree in Amazon...................................... 64

Figure 3-4 The number of associated tags per item (Movielens dataset) 67

Figure 3-5 The number of associated items per tag (Movielens dataset) 67

Figure 3-6 The proposed top-down subtree matching manner 74

Figure 3-7 The complete tree matching result between item 1 and item 2 78

Figure 3-8 The constructed user-item bipartite graph... 79

Figure 3-9 Comparison of the coverage rates under different sparsity levels............ 83

Figure 3-10 Adjusting the model parameter for different data groups 86

Figure 4-1 Modelling multi-relational social networks using union graph vs. multigraph 90

Figure 4-2 Inter-network comparisons with Last.fm dataset 101

Figure 4-3 Ranking on a multi-relational network with a two-moons pattern......... 105

Figure 4-4 Comparison of recommendation coverage (Epinions dataset)..............117

Figure 4-5 Comparison of recommendation coverage (Last.fm dataset)................117

Figure 4-6 Comparison of RMSE on the Epinions dataset.....................................119

Figure 4-7 Comparison of RMSE on the Last.fm dataset.......................................120

Figure 4-8 Performance variation with different parameter settings (Epinions dataset) 122

Figure 4-9 Performance variation with different parameter settings (Last.fm dataset) 122

Figure 5-1 Modelling high-order relations using hypergraph vs. ordinary graph....126

Figure 5-2 The proposed User-Item-Attribute-Context data model.........................132

Figure 5-3 An example of restaurant attributes in terms of five aspects.................133

Figure 5-4 An example of catering group inclusion relations of restaurants...........135

Figure 5-5 An example of neighbourhood relations of restaurants..........................135

Figure 5-6 The user-comment-restaurant high-order relations...............................137

Figure 5-7 The bias of traditional hypergraph ranking model................................142

Figure 5-8 Recall-Precision curves for all compared approaches............................149

Figure 5-9 The performance in terms of MAP under different levels of top-N.......149

Figure 5-10 The performance in terms of MAP under different level of α...........150

Figure 6-1 Examples of single and multi-objective recommendation requests.......154

Figure 6-2 Decomposing multi-objective requests to query vertices.......................155

Figure 6-3 Multi-objective recommendation framework..158

Figure 6-4 A screenshot of the multi-objective requesting page of FoodGo...........160

Figure 6-5 Recommendation page of FoodGo..161

List of Tables

Table 1-1 Basic notations throughout all chapters 18

Table 2-1 Deriving implicit trust from rating behaviours 32

Table 3-1 A brief comparison of taxonomy and folksonomy 68

Table 3-2 Visiting probability at every step of random walks (only item nodes) 80

Table 3-3 The involved input information of each compared approach 82

Table 3-4 Comparison of the NDCG* scores 84

Table 4-1 An example setting of the termination parameter..................................... 96

Table 4-2 Result comparison of the full and alternative propagators 98

Table 4-3 Statistical information of the Epinions dataset 114

Table 4-4 Statistical information of the Last.fm dataset 114

Table 4-5 Involved user-user relations in two datasets 115

Table 5-1 The incidence matrix $\mathbf{H}$ and its sub-matrices 139

Table 5-2 Statistical information of the constructed hypergraph from Yelp dataset145

Table 5-3 Performance comparison with N=5 150

Table 6-1 A random multi-objective recommendation request and query set 162

Table 6-2 NDCG comparison for multi-objective recommendations 164

CHAPTER 1. INTRODUCTION

1.1 Background

In recent years, the explosive growth of web technologies as well as the volume of Internet users promotes the advent of large-scale online applications. Being exposed to millions of options in such applications, users find it hard to identify relevant information from irrelevant information so that the Web personalization technique has gained momentum as a means of challenging this information overloading problem. As a concrete example, personalization of customers aids e-Commerce sites to remember user preference and demands for purchase forecasting or targeted promotions, which will mostly facilitate customer buying decisions. As one of the most notable applications of Web personalization, the Recommender System (RS) has gained considerable attention and undergone rapid developments in recent decades in both academic researches (Bobadilla et al., 2013) and industry applications (Lu et al., 2015). A recommender system is a type of automation response framework whereby a requester (called user) will obtain a list of suggested resources (called items) with no, or at least fewer manual queries. As reviewed in the recent survey work by this author (Lu et al., 2015), recommender systems have been successfully implemented in various domains such as e-Commerce (Zhu et al., 2014), information resources (Li et al., 2010), online study (Wu et al., 2014a), government and business services (Lu et al., 2010), digital library (Bernardi et al., 2011) and tourism (Al-hassan et al., 2011), from which we can find that recommender systems are playing three important roles as follows.

- **Online shopping assistant**. Recommender systems help customers to choose the right products or services from a massive number of options in an e-Commerce site.
- **Information retrieval tool**. Recommender systems help individual users to locate valuable Web information such as news, blogs, tweets, etc.
- **Online decision support tool**. Recommender systems help users to identify the best decisions from alternative plans or solutions, such as study plans, tourist routes, etc.

To produce precise suggestions, recommendation techniques are dedicated to profiling successful user preference and demand, which may be influenced by various sorts of information. Overall, the contributed information in a recommender system includes the item-side information, the user-side information and the third-party information (Shi et al., 2014), all of which are bringing new challenges for recommender systems with the dramatic development of web techniques as well as the popularity of diverse Internet devices such as smart phones and tablets today.

First, the item-side information is becoming complex and high-dimensional. For example, a product in an e-Commerce site is usually associated with comprehensive multi-level descriptions from different perspectives telling people its attributes and functionalities. Such tree-structured item taxonomy information cannot be well modelled by flat vector models in traditional content-based recommender systems. In addition, item content information has been also greatly enriched with the folksonomies, i.e., tags that are freely assigned by users. In the era of Web 2.0, users also play as information creators but not only information consumers to assign their own free-form descriptions to items as supplementary of the standard taxonomy attributes. Clearly, combining the system-provided taxonomy and user-created folksonomy can help people to better understand the overall content information of items. Therefore, it is necessary for a comprehensive content analysis based on both tree-structured taxonomy and folksonomy information, which, however, has not been sufficiently studied in previous recommender systems. Especially, we expect to import folksonomy information to construct the semantical similarities between taxonomy attributes to avoid manual settings of human experts as in existing studies (Shambour and Lu, 2012; Wu et al., 2014b).

On the other hand, the user-side information is also becoming complex with the growth of diverse social network applications. Users' social networks have been proven to be effective to collaborative user preferences as well as the ratings similarities in traditional collaborative filtering (CF) approaches. So far, various types of social relations or correlations between users have been incorporated to enhance recommender systems separately, including both explicit social connections (Shardanand and Maes, 1995; Golbeck, 2006; Massa and Avesani, 2007; Ma et al., 2008; Jamali and Ester, 2009; Yang et al., 2012) and also implicit correlations between users (O'Donovan and Smyth, 2005; Shiratsuchi et al., 2006; Hwang and Chen, 2007; Liang et al., 2010; Lopes et al., 2010; Yuan et al., 2010; Shambour and Lu, 2012; Eirinaki et al., 2014; Katakis et al., 2014). However, there are inadequate studies in handling different types of social networks of users for recommendations. In fact, recommender system users are often connected by multiple social relations or correlations simultaneously (Kazienko et al., 2011). Consequently, how to model the multi-relational social networks of users for recommendation making needs to be investigated in this study.

In addition, there is various third-party information existing in recommender systems, referring to any other contributed resource that may affect or reflect users' acceptance of items. For example, the environmental context such as time and locations has been well considered in many context-aware recommender systems (Woerndl et al., 2009; Adomavicius and Tuzhilin, 2011). Despite environmental context, the textual comment can also be considered as third-parity information that reflects users' evaluations/preference to items. Although both environmental context (Adomavicius and Tuzhilin, 2011) and textual comments (McAuley and Leskovec, 2013) have been utilized separately in specific recommender systems, a generic framework is needed to incorporate all possible third-parity information entities and their pairwise or high-order relationships in the field of recommender systems.

Furthermore, the users' changing and flexible multi-objective recommendation demands are also an emerging challenge not to be ignored for recommender systems. Traditionally, a recommender system only responds to a single user with always the simplest demand at each time, that is, to find the items best fitting this user's past preference. In practice, however, users may have special requirements on item conditions, and there may be different accompanied people thereby their demands may change every time. A recommender system is hence required to be flexible to respond to such multi-objective recommendation requests of single or group users, which becomes another important research topic of this thesis.

In seeking to respond to the above new challenges and also opportunities in the field of recommender systems, this thesis proposes a set of recommender systems in dealing with different scenarios. Particularly, graph models are employed to easily involve different information entities as the vertices and their relations as the edges and the recommendation problem can be naturally represented as a graph ranking problem that seeks for an optimal ranking order of the candidate items (vertices) for request users. The graph-based recommender systems have been the focus of many studies in recent years due to the ease of modelling and high extensibility (Fouss et al., 2007; Jamali and Ester, 2009; Bogers, 2010). As this study involves various information resources and complex relationships that cannot be well handled by simple graphs and conventional graph ranking models, we propose and solve some advanced graph ranking problems for different recommendation scenarios, which can be seen as of significance for the areas of both graph ranking and recommender systems.

1.2 Research Questions

This study reviews four main emerging challenges in the field of recommender systems which significantly motivated the work presented in this thesis. In order to handle these challenges, six research questions are formalized as follows.

In most situations, item content information is so complex and diverse that traditional vector-based models cannot handle it well for recommendation making. The following characters of item content information are considered to be challenging recommender systems in the current era. First, items are often described with rich and tree-structured taxonomy attributes such as the multi-level product taxonomies in e-commerce sites Amazon and eBay. A second challenge is the increasingly user-contributed tags that have been emerging as a new facet of content, called "folksonomy" information of items. In accordance with these challenges, the following research questions need to be addressed in this study.

Research Question 1. How to establish the overall content similarity between items based on both the taxonomy and folksonomy information?

Research Question 2. How to fuse collaborative filtering with the obtained overall content similarities of items to improve recommendation accuracy?

Ratings have been utilized as the only resource for efficient CF approaches; however, as one of the downsides, these approaches will produce inaccurate recommendations if the rating data are too sparse. With the rapid outgrowth of online social networking tools, user social relations are gaining increasing attention as an alternative or additional facet for collaborative filtering. Social network-based recommender systems are emerging as a hot topic in this area but still need adequate studies. One of the most challenging issues is that users are often connected simultaneously with different social networks and/or other implicit correlational networks. Existing single social network-based approaches encounter difficulties in handling such multi-relational social networks of users. Accordingly, the following research questions are formulated.

Research Question 3. How to propagate the implicit relations between indirect users in a social network to reduce data sparsity?

Research Question 4. How to fuse different types of social networks simultaneously occurred in a recommender system to improve recommendation accuracy?

Today, the increasingly diverse information entities and their complex relations become a challenge for precise recommendations. Differing from ordinary pairwise relations, there are particularly high-order relations among multiple information entities. On the one hand, some many-to-many relations arise between users and/or items. For example, a group of users chose items that satisfy all members' preference. These adoption relationships are arising for a set of users and items that cannot be well modelled as ordinary pairwise relationships. On the other hand, there are increasing third-party information entities that bring high-order relations. Considering the impact of environmental context, the original two-dimension user-item acceptance relationship $user \times item \rightarrow acceptance$ becomes a three-dimension relationship $user \times context \times item \rightarrow acceptance$. Despite environment context, the textual comments containing various evaluation topics can also be seen as special information entities selected by the users as personalized evaluations to items, illustrated as $user \times comment\ topics \times item \rightarrow utility$. To utilize all contributed information entities and high-order relations, a generic full-information based recommendation framework needs to be investigated, which leads to the following question.

Research Question 5. How should the diverse information entities and their relations be effectively utilized for full-information based recommender systems?

Existing recommender systems can only respond to single-objective recommendation request, i.e., to predict generally best-fitted items for an individual user. However, users may have flexible and special conditions or demands each time. For example, people often go out for dinner not alone but with different friends or family. At each time, the participants may have different requirements for restaurant conditions or environments. Thus, a new challenge for present recommender systems is the flexible multi-objective request of users. Accordingly, we formulate the following research question.

Research Question 6. How to model and respond to users' multi-objective recommendation requests and generate flexible recommendations?

1.3 Research Objectives

In addressing the above research questions, this thesis aims to achieve the following objectives.

Research Objective 1. To propose an overall content similarity induction algorithm for items based on both taxonomy and folksonomy information.

This objective corresponds to the research questions 1 and 2. Since the taxonomy attributes provided by system managers are associated with almost every single item from the beginning when it becomes available online, while the folksonomy information may be absent if users have yet to tag this item; we focus on comparing the item taxonomy trees to induce the overall content similarity, thus a top-down multi-level subtree matching algorithm will be proposed. Folksonomy information will be utilized in the algorithm in terms of two aspects. First, tag information will be used to determine the most discussed attributes of an item. This finding is valuable to discover the main characteristics of an item and to find other truly similar items. For example, if a movie is known to people by its magnificent scenes, it is appropriate to find another movie also featuring magnificent scenes as the similar item. Second, tag information can be used to discover the semantic similarity of different attributes, which usually need to be manually tuned by human experts (Wu et al., 2014b).

Research Objective 2. To develop a hybrid recommender system integrating item taxonomy and folksonomy information.

This objective corresponds to the research question 2. Based on the induced overall content similarity network of items as well as the preference relations between users and items, a bipartite graph is naturally constructed with users and items as the vertices. We expect to develop a unique random walk model on the user-item bipartite graph either following rating relations (which is similar to the idea of CF approaches) or following the item similarity correlations (which is similar to idea of content-based recommendation approaches).

To achieve research objectives 1 and 2, a hybrid recommender system via bipartite graph random walks is proposed in Chapter 3 of this thesis, which is able to incorporate item content information including both taxonomy attributes and taxonomy tags and user rating information to alleviate the rating sparsity problem of traditional collaborative filtering algorithms.

Research Objective 3. To propose a random walk-based propagation model for single social networks.

This objective corresponds to the research question 3. Due to transitivity of social relations such as friendship and trust that have been widely adopted, social network propagation is an essential and effective research objective in social network analysis (SNA) to enrich the original sparse data. Unlike previous propagation methods that only focus on a micro perspective of predicting indirect trust between a pair of users, we expect to propose a random walk-based graph propagation model from the perspective of the whole social network. The propagation model will be effective and efficient for different types of social networks.

Research Objective 4. To develop a multi-relational social networks-based recommender system.

This objective corresponds to the research question 4. To involve possibly multiple explicit social networks and implicit correlational networks of users, a multigraph model is naturally constructed in order to retain the original structural information. The search of the overall nearest neighbourhood of a target user becomes a multigraph ranking problem that has not been adequately studied. We will develop a novel multigraph ranking model in which both intra-network relations and inter-network diversities are considered to investigate users' overall closeness in a complex multi-relational environment. Consequently, the K-nearest neighbourhood (KNN) recommendation technique will be applied resorting to the multigraph ranking result.

To achieve the above research objectives 3 and 4, a social network-based recommender system via multigraph ranking is proposed in the Chapter 4 of this thesis, which is able to handle multiple explicit or implicit relations between users to enhance recommendation quality.

Research Objective 5. To propose a unified data model to handle all information entities and relations that may appear in a recommender system.

This objective corresponds to the research question 5. The possible information entities and relations that may appear in a practical recommender system will be summarized in a generic data model. With this model, a unified multipartite hypergraph will be constructed with the collected multi-party information entities as the unified vertices and their pairwise or high-order relations as the hyperedges.

Research Objective 6. To develop a full-information based recommender system using balanced hypergraph ranking.

This objective corresponds to the research question 5. With the constructed unified hypergraph, the recommendation problem is transferred to a hypergraph ranking problem which aims to rank item vertices for a particular user vertex. An enhanced hypergraph ranking algorithm, named the balanced hypergraph ranking (BHR) algorithm will be developed as the core algorithm of a full-information based recommender system.

To achieve the above research objectives 5 and 6, a unified recommender system via multipartite hypergraph ranking is proposed in the Chapter 5 of this thesis, which is able to incorporate possible information entities and high-order relationships in an online system for recommendations.

Research Objective 7. To develop a multi-objective recommender system.

This objective corresponds to the research question 7. A multi-objective recommendation request may involve constraints on multiple information entities such as user participants, item conditions and even environments. Thus we need to decompose multi-objective recommendation requests to a computable query set and query vector as the input of the expected multipartite hypergraph ranking model of Research Objective 6. We also hope to develop a demonstration system to show how multi-objective recommendation requests are generated and solved in real-world applications.

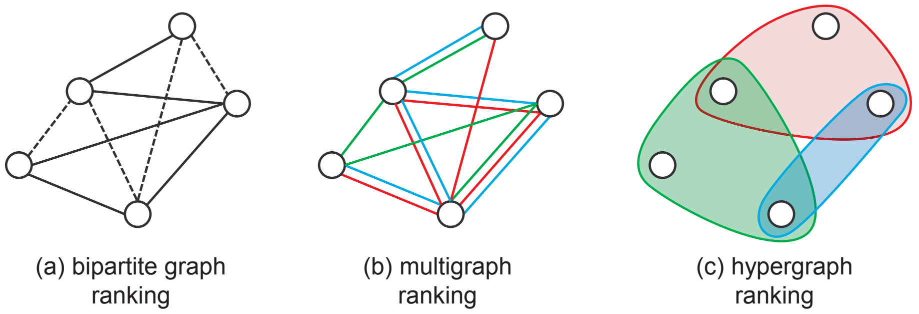
To achieve the above research objective 7, the Chapter 6 of this thesis points out the multi-objective recommendation demands of users and proposes an initial framework of multi-objective recommender system.

1.4 Significance

This research work has both theoretical and practical significances in the area of recommender systems.

1.4.1 Theoretical Significance

Theoretically, the research develops a set of recommendation algorithms and solves the three special graph ranking problems below.


(a) bipartite graph ranking

(b) multigraph ranking

(c) hypergraph ranking

Figure 1-1 This research raises and solves three unique graph ranking problems

The research objectives show that graph ranking techniques are utilized in our research as the main means to address the rising challenges of recommender systems. In particular, we propose and solve three special graph ranking problems in this research as the ordinary graph ranking methods are not applicable to model the new recommendation environments.

- **Bipartite Graph Ranking Problem**. This research solves a bipartite graph ranking problem where the graph involves different and incomparable relations between different groups of vertices. Referring to the research objective 2, a bipartite graph is established with user-item rating relations and item-item content similarity relations. Note that the two types of relations are not comparable directly because they have different natures and also they may value in different ranges. We propose a novel bipartite random walk model which is able to determine which relation is selected to follow for the current node's next move. This unique problem is illustrated in Figure 1-1(a), where the solid lines and dashed lines indicate two different relations.

- **Multigraph Ranking Problem**. This research solves a multigraph ranking problem where two vertices may be connected with multiple relations. This problem arises from the research objective 4. Figure 1-1(b) indicates that in the unique graph two users can be connected by any of the three different relations.
- **Hypergraph Ranking Problem**. This research solves a hypergraph ranking problem. A hypergraph is a generalization of an ordinary graph where edges, called hyperedges, can connect any number of vertices. Referring to the research objective 6, hyperedge model will be used in the research to model all possible information entities as the unified vertices and model their high-order relations as the hyperedges. Thus a hypergraph ranking problem is raised as a recommendation task to be solved in the research.

1.4.2 Practical Significance

Practically, the research provides guidelines of how to incorporate new emerging information resources in recommender systems to improve recommendation quality. For the item side, this research integrates both system-provided, tree-structured, taxonomy information and user-created folksonomy information with model-based CF technique to develop a hybrid recommender system. For the user side, this research incorporates user multi-relational social networks that can be extracted from any explicit social relationships or implicit correlations of users in a recommender system to develop an enhanced social filtering recommender system. For all other third-party information entities and high-order relations, this research proposes a generic data model and full-information based recommender system, which can be easily adjusted for other recommendation scenarios. Furthermore, this research develops a multi-objective recommender system that responds to the flexible and multi-objective recommendation demands of single or group users every time.

1.5 Research Methodology and Process

The overall research methodology and research process are designed as follows.

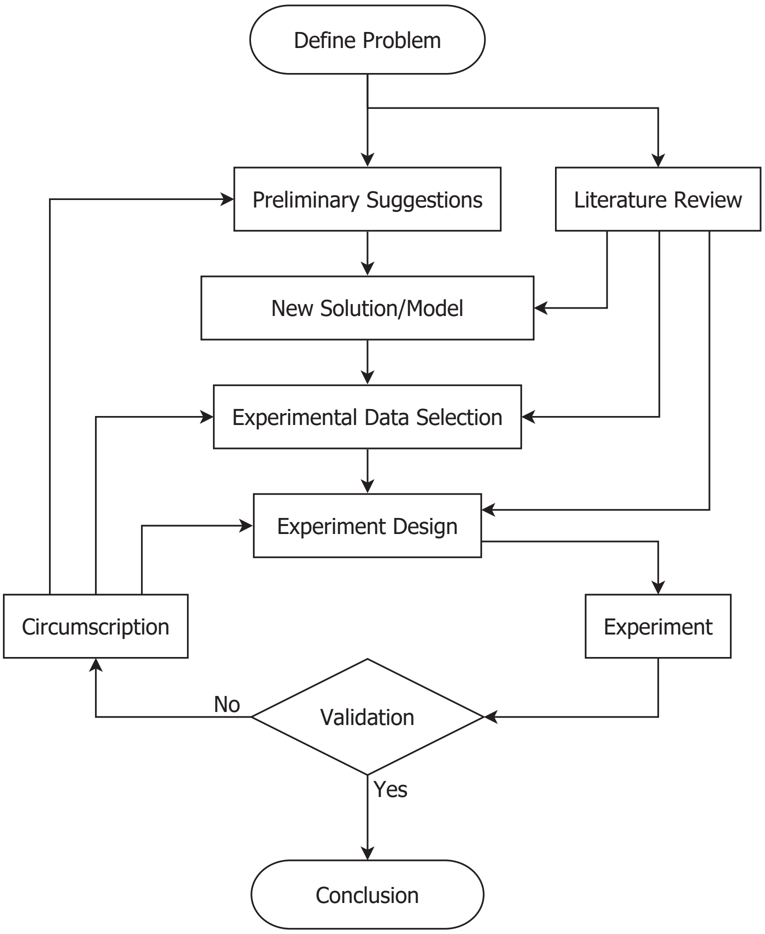


Figure 1-2 The proposed research methodology of this thesis

1.5.1 Research Methodology

Research methodology is the "collections of problem solving methods governed by a set of principles and a common philosophy for solving targeted problems" (Gallupe, 2007). This research belongs to the information system domain, for which various methodologies have been proposed and applied such as case study, field study, design research, archival research, field experiment, laboratory experiment, and survey and action research (Creswell, 2013; Yin, 2013; Vaishnavi and Kuechler, 2015).

As the research has focused on the development of new algorithms or strategies in the recommender system, and the soundness of these systems, techniques or proposed strategies must be supported by the results from the experimentations and evaluations. Therefore, the experimental approach integrated with the standard information system research cycle was chosen as the proposed research method. The process of the research approach used in this research is illustrated in Figure 1-2.

1.5.2 Research Process

This research was planned according to the methodology of design research. First, the subject of recommender systems was chosen as a very broad research topic of this research. A literature review of previous research in the topic area was conducted, and existing literature was retrieved and critically reviewed. The results of the literature review helped to identify specific research questions to be directly addressed in this research. As the research questions grew clearer and more definite, more literature closely related to the research questions was reviewed. Because the existing work in the literature lacks adequate study on the new graph ranking problems abstracted from the research questions/objectives, this research presented some unique graph/hypergraph models, developed corresponding graph/hypergraph ranking algorithms and recommender systems. Appropriate datasets are selected to evaluate the proposed a set of recommender systems. As indicated in Figure 1-2, evaluation results are fed back to previous steps so that the research outcomes are progressively improved until satisfying results are drawn from evaluations. Finally, writing up the PhD thesis is done at the end of the research.

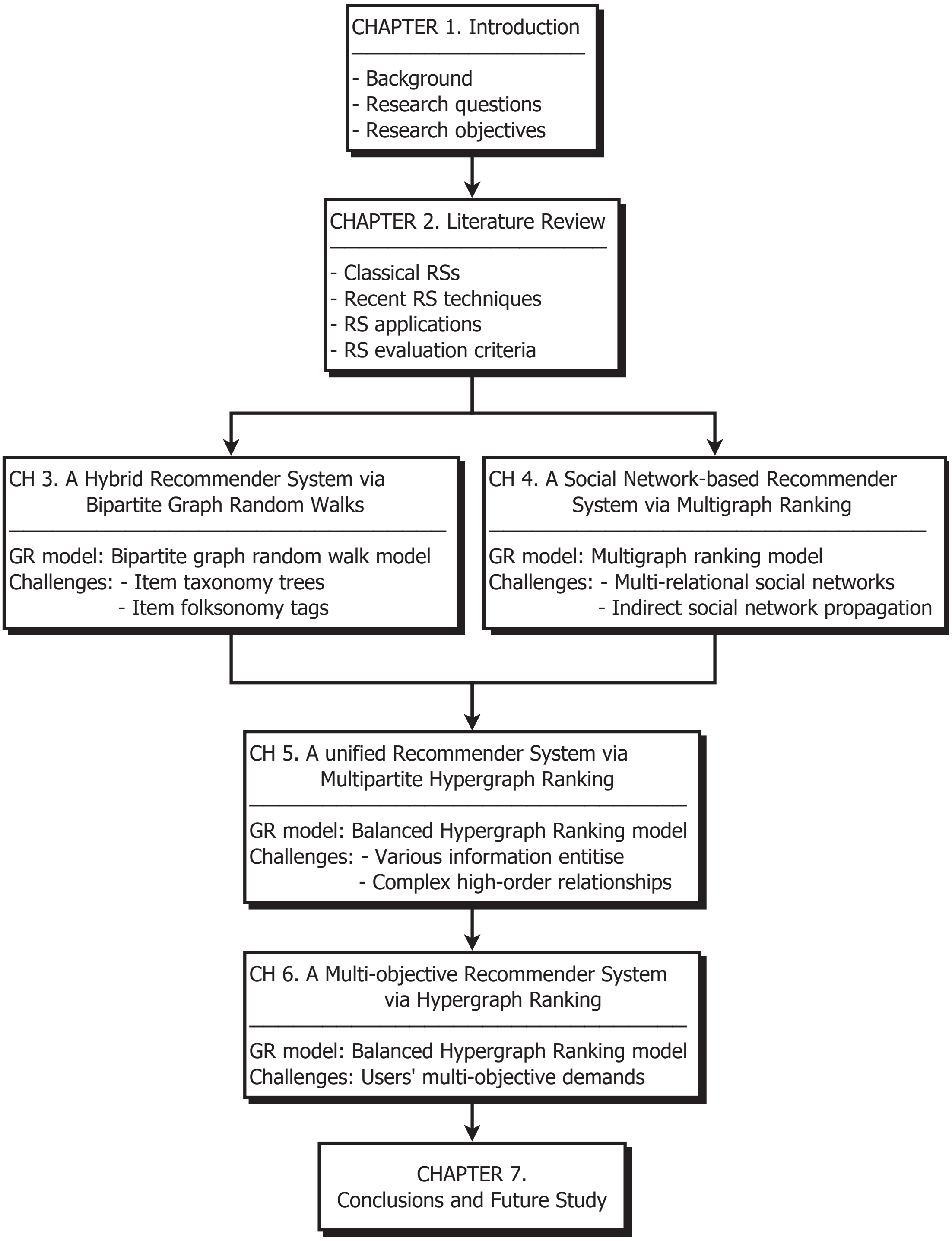

CHAPTER 1. Introduction
- Background
- Research questions
- Research objectives
CHAPTER 2. Literature Review
- Classical RSs
- Recent RS techniques
- RS applications
- RS evaluation criteria
CH 3. A Hybrid Recommender System via Bipartite Graph Random Walks
GR model: Bipartite graph random walk model
Challenges: - Item taxonomy trees
- Item folksonomy tags
CH 4. A Social Network-based Recommender System via Multigraph Ranking
GR model: Multigraph ranking model
Challenges: - Multi-relational social networks
- Indirect social network propagation
CH 5. A unified Recommender System via Multipartite Hypergraph Ranking
GR model: Balanced Hypergraph Ranking model
Challenges: - Various information entitise
- Complex high-order relationships
CH 6. A Multi-objective Recommender System via Hypergraph Ranking
GR model: Balanced Hypergraph Ranking model
Challenges: Users' multi-objective demands
CHAPTER 7. Conclusions and Future Study

Figure 1-3 The structure and contents of this thesis

1.6 Thesis Structure and Publications

This thesis contains seven chapters. Chapter 1 presents the research background, research questions, objectives, significance, research methodology and process, and the basic notations for recommendation problems. Chapter 2 presents the literature relevant to this study, including classical and the state-of-the-art recommendation techniques and applications, and evaluation criteria of recommender systems. Chapter 3 proposes a hybrid recommender system focusing on incorporating both taxonomy and folksonomy information of items and model-based CF techniques. Chapter 4 develops a multi-relational social network-based recommender system to incorporate possible different explicit social relations or implicit correlational relations of users. Chapter 5 reports a full-information based unified recommender system using a multipartite hypergraph model to handle all contributed information entities and their relations. Chapter 6 proposes a multi-objective recommendation framework based on the proposed data model and hypergraph model in Chapter 5 and develops a demonstration system in food industry. Chapter 7 presents the conclusions and further study recommendations. The structure and contents of the thesis are as indicated in Figure 1-3.

The related papers of this thesis that are being under review or published in referred international journals and conferences are listed below.

1) Mingsong Mao, Jie Lu, Guangquan Zhang, Jinlong Zhang, A Multi-objective Recommender System using Balanced Hypergraph Ranking, Decision Support System (DSS). (submitted)

2) Mingsong Mao, Guangquan Zhang, Jie Lu, Jinlong Zhang, A Hybrid Random Walk-based Recommendation Approach Incorporating Taxonomy and Folksonomy, IEEE transactions on Cybernetics (TCYB). (2nd-round review)

3) Mingsong Mao, Guangquan Zhang, Jie Lu, Jinlong Zhang, A Multigraph Ranking Model for Multi-Relational Social Network Recommendations,

IEEE transactions on Cybernetics (TCYB). (3rd-round review)

4) Mingsong Mao, Jie Lu, Guangquan Zhang, Jinlong Zhang, A Fuzzy Content Matching-based e-Commerce Recommendation Approach, in proceedings of the 2015 IEEE International Conference on Fuzzy Systems (Fuzz-IEEE 2015), Istanbul, Turkey.

5) Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, Guangquan Zhang. Recommender system application developments: A survey. Decision Support System, vol 74, pp 12–32, 2015.

6) Mingsong Mao, Jie Lu, Guangquan Zhang, Jinlong Zhang , Hybridizing social Filtering for recommender systems, in proceedings of the 8th International Conference on Intelligent System and Knowledge Engineering (ISKE 2013), Shenzhen, China.

7) Mingsong Mao, Guangquan Zhang, Jie Lu, Jinlong Zhang, A signed trust-based recommender approach for personalized government-to-business e-services, in proceedings of the 7th International Conference on Intelligent System and Knowledge Engineering (ISKE 2012), Beijing, China.

1.7 Basic Notations in This Thesis

In a recommender system, there exist a set of users who are also the recommendation requesters, and a set of items which are the recommendation objects, referring to all candidate resources or solutions in the system. The adoption relationships of users and items are represented as a user-item preference/rating matrix, which is the essential given knowledge of a recommender system. Despite special notations of the unique recommendation models in different chapters, some basic symbols and notations are formalized at first and be used throughout this thesis, as collected in the following table.

Table 1-1 Basic notations throughout all chapters

Notation	Description
$\mathcal{U}$	The user set $\{u_1, u_2, \ldots\}$ in a recommender system
$\mathcal{I}$	The item set $\{i_1, i_2, \ldots\}$ in a recommender system
$u \sim i$ or $i \sim u$	Denotes that a user u has experienced/known/purchased an item i
$\mathcal{U}_i$	The set of users who have experienced/known/purchased the item i
$\mathcal{I}_u$	The set of items that the user u has experienced/known/purchased
$r_{u,i}$ or $r(u,i)$	The rating value given by a user u to an item i, if $u \sim i$
$\hat{r}_{u,i}$ or $\hat{r}(u,i)$	The predicted rating value of a user u to an item i
$\bar{r}_u$	The average ratings issued by a user u
$\bar{r}_i$	The average ratings of an item i

CHAPTER 2. LITERATURE REVIEW

This chapter presents a comprehensive review and analysis of relevant studies in connection with this thesis, including the classical techniques in traditional recommender systems, the state-of-the-art techniques in recent recommender systems, the main recommender system applications and the most popular evaluation criteria for recommender systems.

2.1 Classical Recommender Systems

As mentioned, recommender systems advise online users on the choice of items of personal interest from a huge range of candidates. Early research in recommender systems grew out of information retrieval and filtering research (Goldberg et al., 1992), and recommender systems emerged as an independent research area in the mid-1990s when researchers started to focus on recommendation problems that explicitly rely on the ratings structure (Resnick et al., 1994).

The techniques for recommender systems vary according to prediction strategies of the preference of an individual user to a particular item, which can be classified as three basic ideas: collaborative filtering (CF), content-based (CB) and knowledge-based (KB) (Lu et al., 2015). The idea of CF techniques aims to seek for items that are preferred by other users with similar preferences (Resnick et al., 1994).

The idea of CB techniques aims to search for new items with similar attributes or functionalities to those items that are preferred by the request user in the past (Pazzani and Billsus, 2007). The idea of KB techniques aims to determine the best matched users and items according to existing knowledge rules (Burke, 2000). Each of these three ideas/strategies can be enhanced by the use of advanced computational intelligence (CI) such as fuzzy set theory and probabilistic models thereby a fourth type recommendation techniques can be concluded as CI-based (Lu et al., 2015). Furthermore, there are many hybrid recommendation techniques (Burke, 2002) that dedicate to combine two or more of CF, CB and KB ideas to overcome the drawbacks of single strategies such as the rating sparseness problem of CF.

In brief, the CI techniques provide theoretical supports for the continuous development of CF, CB and KB techniques, whereby hybrid recommender systems are constructed to combine the advantages of CF, CB and KB and to alleviate the shortcomings of a single one. Concretely, the development and the pros and cons of each recommendation technology are elaborated below.

2.1.1 Collaborative Filtering-based Recommender Systems

The collaborative filtering techniques help people to make choices based on the opinions of other people who share similar interests (Deshpande and Karypis, 2004). Resnick and Varian stated that the CF approach built on a significant assumption that “a good way to find interesting content is to find other people who have similar interests, and then recommend titles that those similar users like” (Resnick and Varian, 1997). The CF approach not only provides new suggestions and recommendations to people, but also tries to provide the right information to the right user (Shardanand and Maes, 1995). It has been demonstrated that the CF recommendation approach is the most successful and widely used approach for recommender systems (Herlocker et al., 2002; Schafer et al., 2007; Shi et al., 2014).

CF-based recommender systems have been developed and used in many fields including recommending news, articles, movies, music, products, books and web pages.

The CF-based techniques can be grouped into two general classes: memory-based and model-based (Adomavicius and Tuzhilin, 2005; Shi et al., 2014). Memory-based techniques basically are heuristics that make rating predictions based on the entire collection of previously rated items by the users, which include user-based and item-based CF approaches (Sarwar et al., 2001). The model-based CF algorithms use the existing ratings to build a model which is then used to make predictions for un-rated items. Building a model is a fundamental step for the model-based recommendation techniques. Different machine learning algorithms are used to accomplish the model building process such as the Bayesian network (Babas et al., 2013), clustering (Xue et al., 2005) and rule-based techniques (Abel et al., 2008). These algorithms mainly use a probabilistic approach to compute prediction values for un-rated items (Adomavicius and Tuzhilin, 2005; Schafer et al., 2007).

The memory-based CF technique can be divided into user-based and item-based CF approaches (Sarwar et al., 2001). The user-based CF approach recommends an active user those items which are most liked by the user's nearest neighbours. First, it analyses the user-item matrix and creates a vector for each user which contains all the user's ratings. Then, it computes the similarity between the active user's vector and the vectors of other users. Next, the most similar users to the active user are selected as the nearest neighbours. Predictions are generated using a weighted average of the neighbours' ratings to items.

The item-based CF approach recommends the items that share similar ratings with the ones the active user has rated highly in the past. First, the item-based CF approach creates a vector for each item which contains the ratings from all users to the item. For a target item to an active user, it then computes the rating similarity between the rated items of the active user and the target item. The most similar rated items to the target item are then selected as the item's nearest neighbours. Finally, the prediction for the target item is generated by taking a weighted average of the active user's ratings on the neighbour items. It is found that the item-based algorithms are able to provide the same quality of provided services as the user-based algorithm but with less online computation because the relationships between items are relatively static compared with the relationships between users (Sarwar et al. 2001).

For both user-based and item-based CF, the rating similarity measure between users or items becomes the core technique. There are a number of similarity measures, such as Pearson correlation-based similarity (Resnick et al. 1994) and cosine-based similarity (Sarwar et al., 2001). There are also many advanced similarity measurement proposed in literature for more precise neighbourhoods identification (Candillier et al., 2008), such as the constrained Pearson correlation-based (CPC) similarity (Shambour, 2012) and adjusted cosine-based similarity measures (Deshpande and Karypis, 2004).

The main advantage of using CF recommendation techniques is that they work for any type of items without the need to extract textual descriptions related to the items. It not only suggests similar items according to user interests and preferences, but also suggests new items based on the other people who have the same or similar tastes and interests to a specific user. The major limitations of CF methods are data sparseness and cold-start problems (O'Sullivan et al., 2004; Adomavicius and Tuzhilin, 2005; Bobadilla et al., 2013). The data sparseness problem occurs when the number of available items increases and the number of ratings in the rating matrix is insufficient to generate accurate predictions. When the ratings obtained are very small compared with the number of ratings that need to be predicted, a recommender system becomes unable to locate similar neighbours and fails or produces inaccurate recommendations. The cold-start (CS) problem consists of the CS user problem and the CS item problem. The CS user problem, also known as the new user problem, affects users who have a small number of ratings or none. When the number of rated items for the CS user is small, the CF-based approach cannot accurately find user neighbours using rating similarity so it fails to generate accurate recommendations. The CS item problem, also known as the new item problem, affects items that have only a small number of ratings or none. With few ratings for CS items, CF-based approaches cannot appropriately locate similar neighbours using rating similarity and will be unlikely to recommend them.

2.1.2 Content-based Recommender Systems

The task of a recommender system is to help customers to quickly identify items of interest from a large number of choices. An intuitive way to achieve this goal is to find items that have similar attributes to those previously preferred by a user; this information is usually stored as a content-based user “profile”. The basic principles of CB recommender systems are: (1) To analyse the description of the items preferred by a particular user to determine the principal common attributes (preferences) that can be used to distinguish these items. These preferences are stored in a user profile. (2) To compare each item’s attributes with the user profile so that only items that have a high degree of similarity with the user profile will be recommended (Pazzani and Billsus, 2007).

In CB recommender systems, two techniques have been used to generate recommendations. One technique generates recommendations heuristically using the traditional information retrieval methods, such as the cosine similarity measure. The other technique generates recommendations using statistical learning and machine learning methods. This technique mainly builds models that can learn users’ interests from the users’ historical data (training data). Creating a learning model of a user’s preferences is a form of classification learning. The algorithms used in classification learning, such as decision tree, naive Bayesian and K-nearest neighbours, create a probability function that has the potential to provide the probability estimation for a user's interest to an unseen item. The attained probability can be used to provide users with a sorted list of recommendations (Pazzani and Billsus, 2007). Some examples of CB recommender systems are WebWatcher (Armstrong et al., 1995) and Websail (Chen et al., 2000). In early researches, item attributes and user profiles are represented as flat vectors, and tree-structured items contents are rarely dealt with.

The advantages of the CB recommender systems are that they adopt semantic content of items and recommend to a specific user some items similar to the preferred ones. As a result, a CB recommender system would be able to recommend new items and unpopular items. Furthermore, it does not need to have information about preferences of other users in making recommendations, so it does not suffer from the rating sparseness problem. One of the main limitations of CB recommender systems is the new user problem. CB approach is not able to offer accurate recommendations to a new user since this user has few rated items to construct his/her profile. The CB approach also has the overspecialization problem. It can only recommend items to a user according to the preferred items in trained profile so it cannot recommend items outside the user's profile. Additionally, in some particular cases, it may not be desirable for a recommender system to recommend items which are too similar to users, such as different news articles that describe the same event.

2.1.3 Knowledge-based Recommender Systems

Knowledge-based (KB) recommendation techniques offer items to users based on knowledge about the users and/or items. Usually, a KB recommender system retains a functional knowledge base that describes how a particular item meets a specific user's requirement, which can be performed based on inferences about the relationship between a user's need and a possible recommendation (Burke, 2000).

Case-based reasoning (CBR) technique is a common example of KB recommendation techniques (Smyth, 2007). CBR systems rely on the idea of using the past problem solving experiences as a primary source to solve the new problem. A new problem is solved by retrieving a case whose specification is similar to the current problem and then fitting the attained solution to match the current problem. Case-based recommender systems represent items as cases and generate the recommendations by retrieving the most similar cases to the user's query or profile. Ontology-based recommender system is another classical KB recommendation technique. As a formal knowledge representation method, the ontology information represents the domain concepts and the relationships between those concepts. It has been used to express domain knowledge in recommender systems (Middleton et al., 2009). Ontology-based recommender systems classify items using ontological classes, represent user profiles in terms of ontological terms, use ontological inference to improve user profiling, and use ontological knowledge to bootstrap a recommender system and support profile visualization to improve profiling accuracy. The semantic similarity between items can be calculated based on the domain ontology (Al-hassan et al., 2011).

KB approaches are in the majority of cases applied for recommending complex products and services such as food menu for patients (Arwan et al., 2013). KB recommender systems have no cold-start problem as a case study will be conducted for every new user to get the knowledge of the user's profile or demand. A KB recommender system generates recommendations by computing the similarities between the existing cases and the user's request, so it doesn't require the user to rate or purchase many items previously. KB recommender systems still have some limitations (Burke, 2002; Leung, 2009). For instance, a KB recommender system requires extensive effort to acquire and maintain the knowledge, and to retain the information about items and users for making recommendations. It also requires more feedback and involvement from an active user in order to make an appropriate recommendation for the user.

2.1.4 Computational Intelligence-based Recommender Systems

The common computational intelligence techniques that have been applied for recommendations include Bayesian techniques, artificial neural networks, clustering techniques, genetic algorithms, fuzzy set techniques, etc.

A Bayesian classifier is a probabilistic methodology for solving classification problems. Bayesian classifiers are popular for model-based recommender systems (Amatriain et al., 2011) and are often used to derive the model for CB recommender systems. When a Bayesian network is implemented in recommender systems, each node corresponds to an item, and the states correspond to each possible vote value. In the network, there will be a set of parent items for each item which represent its best predictors. A hierarchical Bayesian network has also been introduced as a framework for combining both CB and CF approaches (Yu et al., 2004).

An artificial neural network (ANN) is an assembly of inter-connected nodes and weighted links that is inspired by the architecture of the biological brain and can be used to construct model-based recommender systems (Amatriain et al., 2011) . Hsu et al. (Hsu et al., 2007) used ANN to construct a TV recommender system by using the back-propagation neural network method to train a three-layered neural network. A hybrid recommender system combining CB and CF was proposed by Christakou et al. (Christakou et al., 2007) to generate precise recommendations for movies. The content filtering part of the system is based on a trained ANN representing individual user preferences.

Clustering entails assigning items to groups so that the items in the same group are more similar than items in different groups. Clustering can be used to reduce the computation cost for finding the k-nearest neighbours, for instance in (Amatriain et al., 2011) . Xue et al. (Xue et al., 2005) presented a typical use of clustering in recommender systems. Their method uses the clusters for smoothing the unrated data for individual users. The unrated items of an individual user in a group can be predicted by use of the rating information from a group of closely related users. Moreover, assuming that the nearest neighbour should also be in the Top N most similar clusters to the active user, only the nearest neighbours in the Top N clusters need to be selected, which enables the system to be scalable. The clustering technique is also used to address the cold start problem in recommender systems by grouping items (Shinde and Kulkarni, 2012). Ghazanfar and Prügel-Bennett used clustering algorithms to identify and solve the gray-sheep users problem (Ghazanfar and Prügel-Bennett, 2013).

Genetic algorithms (GA) are stochastic search techniques which are suitable for parameter optimization problems with an objective function subject to hard and soft constraints (Kim and Ahn, 2008). They have mainly been used in two aspects of recommender systems (Bobadilla et al., 2011, 2013): clustering (Kim and Ahn, 2008) and hybrid user models (Al-Shamri and Bharadwaj, 2008). GA-based K-means clustering is applied to a real-world online shopping market segmentation case for personalized recommender systems in (Kim and Ahn, 2008), which brings improved segmentation performance. A genetic algorithm method is presented for obtaining optimal similarity functions in (Bobadilla et al., 2011). The results show that the obtained similarity functions provide better quality and faster results than those provided by traditional metrics.

Fuzzy set theory offers a rich spectrum of methods for the management of non-stochastic uncertainty. It is well suited to handling imprecise information, the un-sharpness of classes of objects or situations, and the gradualness of preference profiles (Zenebe and Norcio, 2009). In (Yager, 2003), an item in a recommender system was represented as a fuzzy set over an assertion set. The user’s intentional preferences are represented as a basic preference module, which is the ordered weighted averaging of components that can evaluate items. Based on the representation, the preference for an item by a user can be inferred. Cao and Li used linguistic terms for domain experts to evaluate the features of consumer electronic products and allow users to use linguistic terms to express their needs for item features (Cao and Li, 2007). Porcel et al. (Porcel et al., 2009) developed a fuzzy linguistic-based recommender system combining both CB filtering and the multi-granular fuzzy linguistic modelling technique, which is useful for assessing different qualitative concepts. Zhang et al. (Zhang et al., 2013) used fuzzy set techniques to deal with linguistic ratings and calculate the fuzzy CF similarities, to provide a solution for handling uncertainty in a telecom product/service recommendation process.

2.1.5 Hybrid Recommender Systems

To achieve higher performance and overcome the drawbacks of traditional recommendation techniques, a hybrid recommendation technique that combines the best features of two or more recommendation techniques into one hybrid technique has been proposed (Burke, 2002, 2007; Albadvi and Shahbazi, 2009; Shambour, 2012; Zhang et al., 2013). According to (Burke, 2007), there are seven basic hybridization mechanisms of combinations used in recommender systems to build hybrids:

- **Weighted**: scores of each of the recommendation approaches are combined numerically to produce a single prediction (Mobasher et al., 2000; Shambour, 2012);
- **Mixed**: results from different recommendation approaches are presented together, either in a single presentation or combined in separate lists (Smyth and Cotter, 2000);
- **Switching**: one of the recommendation approaches is selected to make the prediction when certain criteria are met using decision criteria (Billsus and Pazzani, 2000);

- **Feature combination**: a single prediction algorithm is provided with features from different recommendation approaches (Basu et al., 1998);
- **Feature augmentation**: the output from one recommendation approach is fed to another (Melville et al., 2002);
- **Cascade**: one recommendation approach refines the recommendations produced by another (Lampropoulos et al., 2012);
- **Meta-level**: the entire model produced by one recommendation approach is utilized by another (Pazzani, 1999).

The most common practice in the existing hybrid recommendation techniques is to combine the CF recommendation techniques with the other recommendation techniques in an attempt to avoid cold-start, sparseness and/or scalability problems (Adomavicius and Tuzhilin, 2005; Bellogín et al., 2013).

2.2 The State-of-the-Art Recommendation Techniques

With the development of recommender systems, new opportunities and challenges are continuously emerging that have facilitated the raise of new recommendation techniques that can enhance recommendations but still need adequate studies. In particular, we enumerate some state-of-the-art recommendation techniques including complex content analysis, social network analysis, graph techniques, context-aware and multi-criteria recommendation techniques, which are highly related to the research questions and objectives of this thesis.

2.2.1 Complex Content-based Recommendation Techniques

Advanced content analysis techniques are emerging for recommender systems due to the growth of more rich and tree-structured taxonomy information and user-created folksonomy tags, as discussed separately as follows.

- **Tree-structured taxonomy information**. Item taxonomies as the major source of content information can be taken advantage of in recommender systems, since they provide a means of discovering and classifying new information about the items to recommend, user profiles and even context (Ruiz-Montiel and Aldana-Montes, 2009). Despite the flat-form attributes that can be modelled by vector models, items in many domains are associated with rich and tree-structured taxonomy attributes. Recently, tree data analyses such as tree similarity measure, tree isomorphism, and sub-tree matching haven been employed in some advanced recommender systems (Arwan et al., 2013; Biadsy et al., 2013; Wu et al., 2014b). For example, in the food menu recommender system for diabetes patients proposed by Arwan et al. (Arwan et al., 2013), items, food menus in this case, are represented as hierarchical food ontology and users (the patients) are represented as weighted nutrition demanding trees. Two patients are then comparable using a weighted tree matching method so that for new patients, personal food menus can be generated according to the diet plans of previous similar patients. Analogously, in the content-based recommend system of (Biadsy et al., 2013), a single user' preference is mapped to a weighted version of item taxonomy tree, and tree matching is used to find the correlation between users to conduct CF-liked recommendations. Wu et al. (Wu et al., 2014b) propose a fuzzy model to represent tree-structured user preferences as well as item taxonomy. The semantic (structural) comparison of user-to-user profile trees, item-to-item content trees and user-to-item preference trees are comprehensively discussed and resolved using fuzzy tree-matching techniques.

- **User-created folksonomy information**. Apart from the standard taxonomy information which is usually maintained by system managers, user-created tags are often emphasized as a kind of folksonomy (folk-taxonomy) information that will enrich item content greatly. Despite being often treated as user-item interactions to be incorporated in some model-based CF approaches (Nanopoulos, 2011; Rafailidis and Daras, 2013), folksonomy tags are more seen as a part of content information describing more attributes or functionalities of items. In (Wu et al., 2014b), tags are added as a new aspect of item attributes to enrich the original item taxonomy trees, thus it is more easy to compare the content similarity between items. Liang et al. (Liang et al., 2010) develop a heuristic recommendation approach integrating both taxonomy and folksonomy information. It measures the personalized weights (preference degree) to tags and taxonomy attributes for each user based on the user-tag-item correlations and item-attribute associations. As a result, content-based user similarities can be measured by comparing their preference to taxonomy and folksonomy, separately, which is further combined with traditional memory-based CF approach to generate recommendations.

2.2.2 Social Network-based Recommendation Techniques

The trust derived from user social networks is also believed to be effective for user collaboration as people often make choices resorting to friends' suggestions. Social networks have been emerging as important facet of resources to replace or be incorporate with CF similarities to alleviate the sparseness problem. More general, a social network between recommender system users can be established from three types of resources: (1) preference correlations derived from user preference such as rating and purchasing behaviours, (2) explicit social relations provided by social networking tools in recommender systems, or (3) implicit correlations derived from user-contributed information and feedback data.

1) User preference similarity and correlations

The rating behaviour of a user towards a particular item involves two levels of meanings. In the first, a quantified value represents how much the user likes this item; in the second, binary experience information is provided on the basis of this user's use of the item, e.g., a person has watched a movie or has purchased a book and can thus "rate" the item. As is known, quantified preferences have been widely utilized to construct the preference correlations of users, such as Pearson Correlation and Cosine similarity measurement, in conventional CF approaches. Binary experience information has also been exploited for recommender systems. For example, in the *ItemRank* model proposed by Gori and Pucci (Gori and Pucci, 2007), a correlation network of items is built using the binary user-rates-item information. Although the study only considers item correlations, we believe that a correlation network of users can also be established based on such binary information.

Previous studies also suggest that an implicit online trust relationship can be generated by the rating behaviours of users (O'Donovan and Smyth, 2005; Hwang and Chen, 2007; Yuan et al., 2010; Shambour and Lu, 2012; Eirinaki et al., 2014). Commonly, they assume that users will implicitly trust others who are found to consistently hold similar opinions, i.e., ratings. It assumes that a single user will observe and judge others' preference, and if someone's rating results are found consistently accurate, an implicit trustworthiness will then arise to this person.

Table 2-1 Deriving implicit trust from rating behaviours

Timestamps	t_1	t_2	t_3	$\rightarrow$	t
Observed Item	i_1	i_2	i_3	$\cdots$	i_t
Observer (A)	5★	1★	4★	$\cdots$	?
Observed user (B)	5★	2★	4★	$\cdots$	1★

An intuitive example is given in Table 2-1. In this case, the observer user A finds another user B always issues the same or similar ratings to their common-rated items in the past time, then at next time when user A faces an unseen item, it is very possible that A will accept the rating of B as a pre-judgement of the item. If user A further find user B's suggestion is accurate, the trustworthiness to B increases more, otherwise decreases. Different calculations of implicit trust are developed in different studies. O'Donovan et al. (2005) propose a rating-derived implicit trust measurements instead of rating similarities to conduct collaborative filtering for recommendations. Yuan et al. (2010) develop an implicit trust metric between users based on their rating similarities rather than deriving it directly from ratings. As a result, the implicit trust network is denser than the rating similarity network, so the sparseness problem is well eliminated. Shambour et al. (2012) also consider the transitivity of implicit trust. For users with no implicit trust can be calculated from ratings, the indirect trustworthiness is inferred using a trust propagation model. The enriched implicit trust network is then integrated with rating similarities for a *trust-enhanced* CF recommendation approach in their work.

2) Explicit social relational network

Increasingly, online systems involve social networking tools to enable customers to interact with each other directly, in the form of online friendships, interest groups, etc. Many researches (Golbeck, 2006; Ma et al., 2008; Massa and Avesani, 2007, 2009; Yang et al., 2012) have contributed to the incorporation of social networks to improve recommendations, especially in the case of sparse rating data. In life, people often resort to their friends for suggestions. It is therefore tempting to incorporate users' trust relationships in social networks to enhance collaborative filtering.

A few systems provide weighted trustworthiness between users so that the trust scores can be directly utilized as the weights to find neighbour users. For example, in the *FilmTrust* recommender system proposed by Golbeck and Hendler (2006), a trust rating in the range of 0 to 1 need to be indicated if a user add a new person to the trust list. The majority of social network-enabled systems, however, only provide binary relations between users such as "who follows who" or "who is linked to whom”. Based on the transitivity of social relations and the need to enrich original networks, social network propagation is often involved as an important component in social network-based recommender systems. Existing approaches mostly use graph searching techniques to infer potential connects between indirect users, where some parameters need to be tuned to control the search breadth or depth. Take the TidalTrust model in (Golbeck and Hendler, 2006) for instance. It is a modified Breadth-First-Search (BFS) model that polls indirect trustworthiness from directly trusted persons. The polling process is started from the source user to the target user via all middle users with some thresholds to control search depth. A similar model called MoleTrust is developed by Massa et al. (2007). Multi-level trust inference models have also been proposed as in (Shambour and Lu, 2012) that if no one-level trustees know the target user, the polling process is recursively conducted at deeper levels. In addition, the maximum allowed searching depth is controlled by new parameters.

It can be seen that existing social network propagation methods are easy to understand from the perspective of individual users, but they lack a global perspective of the whole network. Another limitation is that most existing social network-based recommender systems are only able to handle a single social network of users but cannot be applied for the users surrounding with multiple social relations.

3) Implicit user correlational network

Implicit correlational networks of users can be derived from broader user-contributed information such as users’ demographic information, browsing history or feedback data. Usually are usually thought to be related if they share common characteristics or have engaged in similar behaviours. Some examples are introduced below.

In the website bookmarks recommender systems of (Shiratsuchi et al., 2006), a type of “co-citation” relationship network of users is established as the basis for exploring the implicit correlations of users to see whether they have similar interests when surfing the Internet. Lopes et al. (2010) develop an academic collaboration recommender system to help scientists to find potential opportunities for cooperation and collaboration. An implicit social network of scientists is hence generated based on their relationships of "co-authoring" research papers. A voting advice application is presented in (Katakis et al., 2014), which can be used as a channel to communicate with other voters. A political correlation network of users can then be established according to their interactions; for example, whether they agree or disagree with the political options of others.

In summary, with the increasing user-contributed information in recommender systems, more diverse correlations of users can be extracted as new resource aspects for conducting or enhancing collaborative filtering, particularly for systems with sparse rating data.

4) Fusion methodologies for social network-based recommender systems

Incorporating social networks with conventional user-based CF approaches has gained much attention in the literature, and can be categorized in the following three classes.

- Post hoc combination. This means that each input resource is investigated in separate recommendation approaches, the results of which are later combined to output single result (Lampropoulos et al., 2012; Shambour and Lu, 2012).
- Neighbourhood integration. In contrast to post hoc combinations, social relations and user rating similarities can be aggregated at an earlier stage to establish a union neighbourhood to perform KNN recommendations (Bellogín et al., 2013; Kazienko et al., 2011).
- Unified framework. Unified frameworks are usually applied for the systems with complex networks between users, items and/or additional information entities.

Examples include the multipartite graph model (Jamali and Ester, 2009), hypergraph model (Tan et al., 2011) and cross domain multi-layer network model (Jiang et al., 2012).

In general, post hoc combination approaches combine the predictions of two techniques using weighting functions such as Arithmetic Mean in (Claypool et al., 1999) or Harmonic Mean in (Shambour, 2012).

Neighbourhood interaction can be undertaken in a simple way, such as by mixing different neighbourhoods into a union neighbourhood (Bellogín et al., 2013). In (Jacob et al., 2011; Kazienko et al., 2011), different social networks are aggregated to one single union network and a union neighbourhood will be built for each user for KNN-based recommendations. One of the key concerns of these approaches is the lack of inter-network comparisons. In addition, tuning the weight of every social network may result in high cost of model configuration.

Unified frameworks are usually extensions of the model-based CF approaches. For example, Jamali and Ester (2009) propose an enhanced random walks model, named TrustWalker, to integrate the trust relations of users with CF ideas. The model uses a "walk and select" manner to predict the absent rating of a pair of user and item: it first randomly walks to a "similar" user following the trust relations, and then randomly selects a "similar" item following the rating relations.

2.2.3 Graph Ranking-based Recommendation Techniques

As a recommender system returns a sorted list of items for a request user, it can be seen as to seeking for an appropriate ranking for items. If we model items and/or other information entities in a graph as vertices and their relations as edges, the recommendation problem is simulated as a graph ranking problem, which has drawn many studies in recent years (Belkin et al., 2004; Agarwal and Chakrabarti, 2007; Agarwal, 2006, 2010).

Random walk theory is one of the popular explanation of graph ranking and has been increasingly employed in recommender systems, especially to handle complex relational networks of users, items and/or other information entities (Fouss et al., 2007; Gori and Pucci, 2007; Jamali and Ester, 2009; Bogers, 2010). The concept of random walk on graph was first proposed by the PageRank algorithm (Page et al., 1999), which simulates the surfing behaviour of web users on webpages. It assumes that the probability of a user randomly jumping to a new page is determined each time by the page currently being visited. With the transition between webpages, the visiting probability distribution will reach convergence as proven in (Athreya et al., 1996) and the stationary visiting probability is used to present a particular user's interest level in a webpage. The idea of random walk has also been introduced into recommender systems, especially to handle complex relational networks of users and items. A basic random walk model if performed on a bipartite graph constituted by user nodes (V_{user}) and item nodes (V_{item}), following the rating relations (Fouss et al., 2007). In the unique model, a random walk either jumps from a user node to an item node, or jumps from an item node to a user node.

Recommendations for a particular user will be generated based on the ranks of stationary visiting probabilities of unseen items. The ItemRank model proposed by Gori and Pucci (2007) performs random walks on an item-to-item correlational network. In this model, item correlation is established by the number of shared common users, i.e., the users who have rated both items. Similarly, Yildirim and Krishnamoorthy (Yildirim and Krishnamoorthy, 2008) present an item-based random walk model using the rating similarity network of items. In contrast, Jamali and Ester (2009) integrate the additional information of user trust relations and perform random walks on user network, which is further integrated with traditional memory-based CF approaches. Not only users and items, but also other information entities such as item attributes of items have been introduced as special vertices to perform unified random walks on a multipartite graph (Cheng et al., 2007; Bogers, 2010; Lee et al., 2011). Generally, let $G = (V_{\text{user}}, V_{\text{item}}, V_{\text{content1}}, V_{\text{content2}}, \ldots, E)$ denote a multipartite graph containing users and items as well as a number of content attributes from different perspectives such as movie genres and actors, random walks will be performed by jumping between different types of vertices following the user-item rating relations or item-attribute association relations. A more complicated unified model is illustrated in (Bogers, 2010) with more parameters to control the weights of transitions between different types of vertices.

Despite the ordinary simple graphs that can only include pairwise relations, a few hypergraph models have also been newly proposed to handle high-order relations. For example, Tan et al. (2011) model the tags and the music album-tracks inclusion relations as hyperedges in a hypergraph to produce music recommendations using hypergraph ranking models (Zhou et al., 2006). Going a step further, Theodoridis et al. (2013) consider the different contributions of different parties of information entities for hypergraph ranking and introduce group sparsity constraints for hypergraph ranking-based recommendations.

In summary, graph models have advantages to incorporate various input information recourses of a recommender system by naturally modelling all related information entities including users, items and other objects as vertices and all complex relations as edges. That is to say, graph models are easily extendable for modern recommender systems with increasingly diverse information brought by the explosion of Web 2.0 applications.

2.2.4 Context-aware Recommendation Techniques

One of the most cited definitions of context is the definition of Dey et al. (Dey et al., 2001) that defines context as “any information that can be used to characterize the situation of an entity. An entity could be a person, a place, or an object that is considered relevant to the interaction between a user and an application, including the user and the application themselves.” The context information such as time, geometrical information, or the company of other people (friends, families or colleagues for example) has been recently considered in existing recommender systems, for example, the information obtained with the rapid growth of using mobile handsets (Woerndl et al., 2009). The contextual information provides additional information for recommendation making, especially for some applications in which it is not sufficient to consider only users and items, such as recommending a vacation package, personalized content on a website. It is also important to incorporate the contextual information in the recommendation process to be able to recommend items to users in specific circumstances. For example, using the temporal context, a travel recommender system might make a very different vacation recommendation in winter compared to summer (Stabb et al., 2002). The contextual information about users in technology enhanced learning environments is also incorporated into the recommendation process (Verbert et al., 2012).

In the review of Adomavicius and Tuzhilin (2011), context in the recommender system field is a multifaceted concept used across various disciplines, with each discipline adopting a certain angle and putting its “stamp” on this concept. With context awareness, the original two-dimension user-item adoption relationship $user \times item \rightarrow adoption$ becomes $user \times context \times item \rightarrow adoption$, a three-dimension relationship . To identify and incorporate the contextual information in recommender systems, three processes steps are suggested in (Adomavicius and Tuzhilin, 2011) including Contextual Pre-Filtering, Contextual Post-Filtering, and Contextual Modelling. By processing these analyses the system can detect the useful and compliable contextual information for recommendations.

2.2.5 Multi-Criteria and Group Recommendation Techniques

Some unique recommender systems have been designed to extend the functionalities of conventional recommender systems, such as multi-criteria recommendations and group recommendations.

There are some researches in recent years that model recommendation problem in multi-criteria environments and the Multi-Criteria Decision Making (MCDM) techniques are employed to help users to select items (Adomavicius et al., 2011 b; Shambour, 2012; Jannach et al., 2012). Generally, multi-criteria recommender systems allow users to issue multi-criteria ratings to evaluate items from different perspectives such as movie story, acting, direction, etc., which acknowledge that the suitability of the recommended item for a particular user will probably depend on more than one utility-related criterion that the user takes into consideration when deciding whether an item is interesting and suitable for him/her. The additional information that is provided by multi-criteria ratings can represent more complex preferences of each user and thus help to improve the quality of recommendations.

Group recommender systems are proposed to produce suggestions for a group of users when the members are unable to gather for negotiation, or their preferences are not clear in spite of meeting each other (Jameson and Smyth, 2007). Different strategies have been proposed in group recommender systems to aggregating the different choices of group members, such as in (Gorla et al., 2013; Masthoff, 2011; Quijano-Sánchez et al., 2012). Clearly, group recommender systems are able to generate corresponding suggestions for multiple participant/request users in a recommender system.

2.3 Applications of Recommender Systems

As mentioned, the three most important roles that recommender systems have played are shopping assistants, information retrieval tools, and decision supports, all of which have gained various applications in many domains. The development and functionalities of each type application are introduced as follows.

2.3.1 Recommender Systems as Online Shopping Assistants

Electronic commerce provides customers the convenience of browsing and paying products or services online, which is also accompanied with the explosively increasing volume of online items that is challenging customers to locate the right options in a reasonable time. In response to this information overloading problem, recommender systems are originally emerging as shopping assistants to predict the best choices for customers.

For generic online shopping systems, rating is a common function reflecting the explicit user preferences. In the iTunes store, for example, customers are able to provide feedback by allocating a numeric value to evaluate purchased apps. In addition, textual comment is another way for users to express detailed evaluations for items. Ratings and textual comments are hence the two most important resources for user preference modelling that have been widely utilized in CF approaches (Shi et al., 2014) and textual mining-based recommendation approaches (Blei et al., 2003; Li et al., 2010). There are also increasingly hybrid recommender systems that incorporate both rating and comments in a unified framework for more precise recommendations (Diao et al., 2014; McAuley and Leskovec, 2013).

There are also some shopping systems where it is difficult to collect explicit user preference data, knowledge-based analyses are often employed. For example, the Wasabi Personal Shopper (WPS) (Burke, 1999) is a domain-independent database browsing tool designed for online information access, particularly for electronic product catalogues. Fuzzy techniques are also employed in content-based e-shopping recommender systems such as that Cao and Li (2007) developed a fuzzy-based recommender system for products consisting of different components. When buying a laptop, for instance, shoppers may consider the individual performance of each component, such as the CPU, motherboard, memory, etc. In this application, the weights of a shopper's needs on each component are collected and the most satisfied candidates are then generated according to a fuzzy similarity measure model. In the book recommender system of (Mooney and Roy, 2000), a naive Bayesian text classifier is used to train the data abstracted from the web to build features of books and profiles of users and find the best matched books for a target user. Certain shopping assistant systems have an interest in explaining why and how the recommendations are generated. For example, when buying expensive goods, buyers expect to be skilfully steered through the options by well-informed sales assistants who are capable of balancing the user's various requirements. To provide an equivalent virtual recommendation explanation such as "why product A is better than B", McCarthy et al. (2004) developed a shopping assistant website called Qwikshop.com on which compound critiques were used as explanations. Another issue for practical e-shopping system is the bundle promotion. In the systems developed by Garfinkel et al. (2006), a "shopbot" (shopping search engine) is implemented to consider purchasing plans for a bundle of items. This recommender system leverages bundle-based pricing and promotional deals frequently offered by online merchants to extract substantial savings. Zhu et al. (2014) considers the bundle recommendation problem as a type of Quadratic Knapsack Problem (Gallo et al., 1980) that has increased the purchase intention of users on an experiment of Walmart.

2.3.2 Recommender Systems as Information Retrieval Tools

The information resource mentioned here refers to the content uploaded by system mangers or users, such as news, blogs and tweets. Online users often share information to the Internet so that other users can access the resources that interest them. Differing from keywords-based search engines, recommender systems have the superiorities in terms of two aspects: (1) recommender systems do not require users to input keywords explicitly but track user preference at background and predict user demands automatically; (2) recommender systems provide personalized results for different users.

In most instances, textual content such as news, emails, documents and webpages is described as a list of keywords, which can be extracted from historic data, URLs and public encyclopaedias, and many recommender systems are designed on the basis of keywords analyses. Based on the inferred user-resource preference, memory-based CF techniques can be applied such as in the joke recommender system Eigentaste of (Goldberg et al., 2001). Model-based systems have also been developed, such as News Dude (Billsus and Pazzani, 2000), which builds long term preferences through Bayesian methods, and Foxtrot (Middleton et al., 2002), which uses k-nearest classification. Graph-based clustering is adopted in WinPUM (Jalali et al., 2010), in which the websites are modelled as graph vertices and user navigation patterns are classified according to session information. Recently, Nguyen et al. (2013) incorporate ontology and semantic knowledge to analyse session data to improve recommendation accurately. Apart from the keywords taken from the textual content, implicit and explicit feedback from users is also taken into account. In the webpage recommender system ArgueNet (Chesnevar and Maguitman, 2004), for example, a use is allowed to indicate which websites are trustworthy for him/her as an additional feedback.

In general, explicit scaled ratings are not provided in information resource recommender systems such that implicit user preference needs to be established from more widely implicit behaviours or relations such as user browsing history and item contents. Although some memory-based CF approaches have been applied in early years, it is more common that information resource recommendations are resolved by advanced learning algorithms such as graph and latent topic models.

2.3.3 Recommender Systems as Online Decision Support Tools

In specific domains where users do not interact with items frequently or there are no clear options pre-stored as the "items" for users to select, recommender systems become like one-time decision support systems to help users to narrow the candidate options of alternative plans. For example, an e-learning recommender system is dedicated to help each learner to make appropriate study schedules (Capuano et al., 2014) . In the follows, we enumerate some popular applications of this type recommender system in different domains.

Online learning (e-learning) recommender systems aim to assist learners to choose the appropriate courses, subjects, materials, and learning activities or their combinations. In the personalized e-learning material recommender system (PLRS) proposed by Lu (2004), a computational analysis model is developed to identify the leaner's learning requirement and then uses matching rules to generate a recommendation of learning materials. In addition, by incorporating with web usage mining and CF, Lu et al. (2004) developed a subject e-learning recommender system (SRS) to locate the right subject information for the right students according to their individual interests and needs in subject selection. The pedagogical rules are important knowledge for learning systems, which have been highlighted the applications of (Biletskiy et al., 2009; Cobos et al., 2013; Santos et al., 2014; Verbert et al., 2012). For example, Cobos et al. (2013) define an ontology to represent the pedagogical patterns and their interaction with the fundamentals of the educational process, and applies a unified hybrid model combining CB and CF techniques to produce recommendations. The ontologies of learners and learning objectives are also discussed in (Biletskiy et al., 2009), which provide a technical solution for personalized searches of learning objects on the web.

E-tourism recommender systems are designed to provide suggestions for tourists. Some systems focus on attractions and destinations, while others offer tour plans that include transportation, restaurants and accommodation. Pashtan et al. (2003) propose a context-aware tourist information system named CATIS to recommend accommodation, restaurants and attractions for tourist. The recommendations are generated by combining the user query and the user context information from the application server where a context manager module is dynamically collecting the context information such as time and locations. A personalized sightseeing planning system (PSiS) is developed by Lucas et al. (2013) to aid tourists to find a personalized tour plan in the city of Oporto, Portugal. Both CF and CB ideas are incorporated in this hybrid system as well as some computational intelligence techniques such as fuzzy logics for user profiling. SigTur/E-Destination (Moreno et al., 2013) is designed to provide personalized recommendations of touristic activities. Various types of information, such as user demographic details, destination context, geographical aspects and explicit or implicit requirements, are integrated to build user demand to be compared with the tourism ontology to identify the appropriate leisure activities. In the cost-aware travel tour recommendation proposed by Ge et al. (2014) , two latent factor models to recommend travel packages by considering both not only the tourist’s interests but also the fanatical and time cost of travel.

There are also many specific domain-based recommender systems that help people for decision making. For example, Cornelis et al. (2007) propose an one-time recommendation framework for special items of which there is only one single instance (like an event). An application in trade exhibition recommendation for e-government services is illustrated in their work where fuzzy logic is applied to estimate the similarities of new coming events with previous events. Arwan et al. (2013) develop a diabetic food recommender system to assist the nutrition experts to choose appropriate daily foods menu for diabetes patients for controlling blood sugar levels. In the system, domain knowledge based on food ontology including nutrition and caloric information is combined with a semantic search method to generate and identify candidate food menus.

The above reviews show that KB techniques are mainly applied in this type of recommender systems, which usually have insufficient historical data for CF or CB algorithms. The architecture of this type of recommender systems usually consists of three parts: (1) determine user needs from user background information, historical behaviours and context information that may affect the choices; (2) collecting taxonomies or ontologies to generate or represent the possible options (items); (3) acquiring related matching rules or algorithm to evaluate the matching degree between users and candidate options.

2.4 Evaluation Criteria for Recommender Systems

A number of evaluation measures have been used to assess the recommendation quality of recommender systems, which can be classified into the following four categories (Bobadilla et al., 2013).

- **Recommendation success rate**. It evaluates the ability of a recommender system to complete all recommendation tasks in experiments.
- **Rating prediction accuracy**. It evaluates weather a single predicted rating is accurate.
- **Classification accuracy**. It evaluates whether the recommended items are the

right items that the request user prefers.

- **Item ranking accuracy**. It evaluate the ranking order of recommended items is accurate.

The most used metrics for each type evaluation are detailed as follows.

2.4.1 Recommendation Success Rate

As a result of the data sparseness and cold-start problems, it is possible that a recommender system fails to predict the rating or matching rate of a single user to a particular item. In experiments, a recommender system is required to complete the recommendation tasks hidden in test set. Therefore, the ability of completing all recommendation tasks indicates the success rate of this recommender system, which is usually represented by the recommendation coverage metric as defined as follows.

Given the test set $\mathcal{T}$ containing pairs of $\langle user, item \rangle$ and their actual ratings (or preference scores), a recommender system is implemented to predict a score such as rating or matching rate for each of such pairs. The term coverage is computed as the completion percentage of all prediction tasks in $\mathcal{T}$:

$$coverage = \frac{\sum_{\langle u,i \rangle \in \mathcal{T}} \mathcal{P}(u,i)}{|\mathcal{T}|} \times 100\%, \tag{2.1}$$

where $\mathcal{P}(u,i)$ is a Boolean showing whether the recommender system can successfully predict a rating or matching rate for the user u and the item i.

In literature, the metric *coverage* has been widely used to evaluate the ability of a recommender system for alleviating the sparseness and cold-start problems (Massa and Avesani, 2007; Bobadilla et al., 2013).

2.4.2 Rating Prediction Accuracy

Varied by recommendation strategies, recommender systems can be classified to two categories: rating prediction-based and ranking-based. The prediction-based recommender systems predict the missing rating of a user to an unseen item, and all candidate items are then ranked according to the acquired predictions. The ranking-based recommender systems do not predict the rating values of users to items but directly output a sorted list of items based on unique utility scores. In general, KNN-based recommender systems such as memory-based CF and some social network-based approaches are prediction-based (Resnick et al., 1994; Massa and Avesani, 2009; Shambour and Lu, 2011). The majority of model-based CF or advanced approaches are ranking-based (Gori and Pucci, 2007; Tan et al., 2011; Shi et al., 2014).

For a prediction-based recommender system, it is necessary to evaluate whether the prediction ratings are accurate. If a recommender system successfully predicts a rating for a pair of $\langle u, i \rangle \in \mathcal{T}$, that is, $\mathcal{P}(u, i) = 1$, and the predicted rating is denoted as $\hat{r}_{u,i}$, the deviation/error between $\hat{r}_{u,i}$ and the actual rating $r_{u,i}$ in the test set is used to evaluate the prediction accuracy of this recommender system. Two widely used metrics MAE and RMSE are introduced below.

The MAE (*Mean Absolute Error*) is defined in the following equation:

$$\mathrm{MAE} = \frac{\sum\limits_{\langle u,i \rangle \in \mathcal{T}, \mathcal{P}(u,i)=1} |\hat{r}_{u,i} - r_{u,i}|}{\sum\limits_{\langle u,i \rangle \in \mathcal{T}} \mathcal{P}(u,i)}. \tag{2.2}$$

The RMSE (*Root Mean Squared Error*) is defined as follows:

$$\text{RMSE} = \sqrt{\frac{\sum\limits_{\langle u,i\rangle \in \mathcal{T}, \mathcal{P}(u,i)=1} \left(\hat{r}_{u,i} - r_{u,i}\right)^2}{\sum\limits_{\langle u,i\rangle \in \mathcal{T}} \mathcal{P}(u,i)}}. \tag{2.3}$$

Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the RMSE is most useful when large errors are particularly undesirable.

2.4.3 Classification Accuracy

In experiments, a recommender system can be understood as a classifier to guess the preferred or not-preferred items for a particular user in the test set. Thus the classification evaluation metrics Precision and Recall rates are imported to evaluate recommendation performance, as defined as follows.

Recommendation Precision is defined as:

$$Precision = \frac{Number\ of\ correctly\ recommended\ items}{Number\ of\ recommended\ items} \tag{2.4}$$

Recommendation Recall is defined as:

$$Recall = \frac{Number\ of\ correctly\ recommended\ items}{Number\ of\ actual\ preferred\ items} \tag{2.5}$$

In addition, F1 metric is the harmonic mean of Precision and Recall:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2.6}$$

Average Precision (AP) is the average of precisions computed at the point of each correctly recommended item in the recommendation list:

$$AP = \frac{\sum_{i=1}^{N} Precision@i \times corr_i}{Number\ of\ correctly\ recommended\ items} \tag{2.7}$$

where $\mathrm{Precision}@i$ is the measured precision at ranking position i, i.e., the precision level when a model recommends i items, and $\mathrm{corr}_i = 1$ if the restaurant at position i is correctly recommended, otherwise $\mathrm{corr}_i = 0$. MAP is the mean of AP scores over all users.

2.4.4 Item Ranking Accuracy

A recommender system returns the request user a sorted list of items as the final result. Thus we can compare the ranking order of the recommended items with the actual ranking order in test set as an evaluation of the recommendation performance.

The metric NDCG (*Normalized Discounted Cumulative Gain*) is a widely adopted measure of ranking quality. First, the *Discounted Cumulative Gain* (DCG) accumulated at a particular rank position p is defined as:

$$\mathrm{DCG}@p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)} \tag{2.8}$$

where rel_i denotes the ranking score at position i.

The NDCG is then calculated by:

$$\mathrm{NDCG}@p = \frac{\mathrm{DCG}@p}{\mathrm{IDCG}@p} \tag{2.9}$$

where IDCG is the DCG of the “ideal ranking order”, i.e., the ranking order based on the actual ratings in test set.

2.5 Summary

This chapter reviews some highly related backgrounds which are essential for recommender system studies. We first review the classical ideas for recommender systems. This helps us to understand the main objects and constraints of recommender systems and grasp the main frameworks or ideas of recommendation making. Second, we review the advanced state-of-the-art recommendation techniques which give us a glance of the most recent developments in the study of the field of recommender systems. Next, we also conduct a comprehensive survey of the practical applications of recommender systems. The various applications indicate recommender systems have been implemented widely in our daily life as shopping assistants, information retrieval tools and decision support tools. The last, we elaborate the main evaluation criteria for recommendation approaches. As this thesis proposes a set of recommender systems in different scenarios, comprehensive evaluations from different criteria such as success rate, prediction accuracy, classification accuracy and ranking accuracy are needed.

CHAPTER 3. A HYBRID RECOMMENDER SYSTEM VIA BIPARTITE GRAPH RANDOM WALKS

3.1 Overview

As reported in literature, hybrid recommender systems are widely applied in online applications to combine the advantages of different recommendation techniques and overcome the drawbacks of single techniques such as the rating sparseness problem of CF. Item content information is often incorporated with rating-based CF approaches in hybrid recommender systems. The following two facets of content information are increasingly attracting researches and applications in hybrid recommender systems. The first is the standard taxonomy of items. Generally, taxonomy is provided and maintained by system managers or sellers such as in e-Commerce sites, consisting of tree-structured descriptions from different perspectives for consumers to best know the details of the attributes and functionalities of items. On the other hand, the second aspect of content information is the user-created tag information, called "folksonomy" (folk-taxonomy), of items. In the era of Web2.0, users are able to freely create their own descriptions (taxonomies) for items in the form of single words or short phases, namely, the tags, as additional remarks for items beyond the standard taxonomy attributes.

In literature (Arwan et al., 2013; Belém et al., 2014), taxonomy and folksonomy have been utilized in a number of recommender systems separately, but there have been few studies that incorporate both of them for a comprehensive content analysis and to enhance recommendation quality (Liang et al., 2010). Intuitively, combining the system-provided taxonomy and the user-created folksonomy can help people to better understand and identify the overall content information of items. This motivates the first task of this chapter: combining tree-structured taxonomy information and free folksonomy information for item content analysis.

The second part of this chapter is to propose a hybrid recommendation algorithm incorporating the ideas of both CF and CB approaches. We chose graph models to handle diverse relational networks between users and items (Gori and Pucci, 2007). In this research, we incorporate item taxonomy and folksonomy to establish an overall content similarity relationship between items (the first research task), which is further combined with the user-item preference (rating) relations to construct a user-item bipartite graph. We then propose a unique random walk model on this bipartite graph to produce hybrid recommendations to suggest users the appropriate items.

The motivated bipartite random walk model is illustrated in Figure 3-1 using an example of a user browsing in a movie recommender system. At first, the overall content similarities between items (movies) are supposed to have been obtained by comparing both the taxonomy and folksonomy information of movies. Combining the item-item content similarity and user-item rating relations, a bipartite graph is constructed with users and items as the two groups of vertices. Thus, the behaviour of a particular user browsing movies can be simulated as a random walk process on the bipartite graph. In our example, let user A be the starting user and movie 3 be the target item. There are two directions of moving from user A to movie 3 in the unique bipartite graph:

- Reaching the target item via an item-item content similarity relation, e.g., the router 1 in the figure;
- Reaching the target item via a user-item rating relation, e.g., the router 2 in the figure.

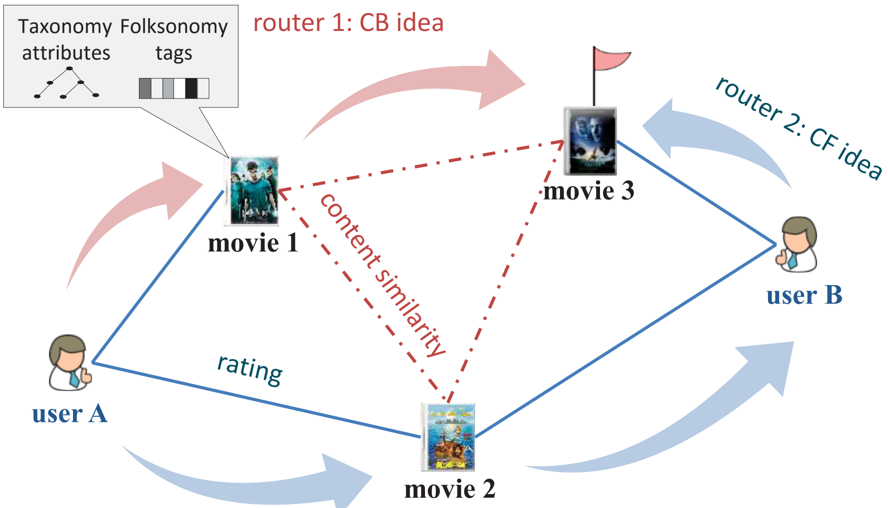


Figure 3-1 Random walking on a user-item bipartite graph

In this example, the router 1 essentially indicates a CB recommendation idea that seeks out the items similar to the ones that are known being preferred. In contrast, the router 2 indicates a CF idea that finds the items preferred by a neighbour user. Therefore, this model can be seen as a hybrid recommendation approach combining both the ideas of CB and CF.

In brief, the aims and also the contributions of this chapter consist in two parts: (1) to design a unique random walk model on a user-item bipartite graph with their user-item rating relations and item-item content similarity relations; (2) to propose a tree matching algorithm to compare the tree-structured taxonomy and folksonomy information of items. We will elaborate how folksonomy tags are utilized as an additional resource for semantical analysis of taxonomy attributes and how item content information is compared using a top-down tree matching manner.

The remainder of this chapter is organized as follows. Section 3.2 first elaborates on the random walking strategy on the proposed bipartite graph of users and items via their rating relations and content correlations. Section 3.3 proposes an overall content similarity induction algorithm by integrating item taxonomy and folksonomy as input information. Section 3.4 is a numeric example illustrating how our approach is implemented in real cases. In Section 3.5, empirical experiments are conducted with Movielens data under different sparsity levels. Our model is compared to a number of single and hybrid approaches. The summary of this chapter is given in the last section.

3.2 User-item Bipartite Random Walk Model

We consider a common scenario of recommendations with such given information: users have given numeric ratings (1 to 5, for example) to evaluate the items they have known; and every item is associated with its taxonomy attributes (provided by system) and tags (created by users). A bipartite network involving two groups of nodes, the users and items, is constructed to perform random walks. Clearly, a user node and an item node are connected via rating behaviours, indicating the degree of preference of the user to the item. We also believe that a content similarity relation can be established between items by comparing their content information including both taxonomy attributes and folksonomy tags. Thus, for two items i and j, their content similarity is derived with a comparison function of their attributes and tags:

$$sim_{i,j} = \mathcal{F}\Big(\{attributes^i, tags^i\}, \{attributes^j, tags^j\}\Big) \tag{3.1}$$

At first, the induction of the above formula is left to discuss later in the following section; for now, we just assume a content similarity network of items has been obtained as given information to construct the user-item bipartite graph.

Note that we do not consider the directed connections between users such as social connection or rating similarity because it is behind the goal of this chapter to propose an item taxonomy and folksonomy-based hybrid recommendation model. We will exploit how different user-user networks are utilized to enhance recommendations in the next chapter.

3.2.1 Performing Random Walks

We perform random walks on the user-item graph starting from an active user for whom the recommendation is to be made. The idea is that after a long-term walking to convergence, the most visited items are considered as highly relevant to the starting user.

Walking through the path of a pair of $\langle user, item \rangle$ or $\langle item, item \rangle$ is assuming that the two nodes are "relevant". Therefore, negative connections such as negative ratings or low item similarities should be pruned to avoid harmful information. In our study, two thresholds θ_r and θ_c are introduced. The ratings less than θ_r and the item similarities less than θ_c will be omitted for performing random walks.

The visiting position at a certain step is at either a user node or an item node. Thus we have the following two situations for concrete walking manners.

1) Current position is a user node

If a user node $u \in \mathcal{U}$ is the current visiting position at time t, we assume that there are two options for the next move:

Option 1. With probability α, the runner continues to walk, and randomly moves to an item node preferred by the current user u.

Option 2. With probability $1-\alpha$, the runner jumps back to the starting node to restart the walk.

Note that the random walk with restarting models have been widely adopted in existing studies by importing a decay parameter $\alpha \in (0,1)$ (Gori and Pucci, 2007; Shi et al., 2014)[1].

In the case of the first option, user u selects one of his/her preferred items to move. The probability of selecting a particular item $i \sim u$ is:

$$p(i|u) = \frac{r_{u,i}}{\sum_{j \sim u} r_{u,j}} \tag{3.2}$$

2) Current position is an item node

On the other hand, if the runner is currently at an item node $i \in \mathcal{I}$, the options of next move are as follows.

Option 1. With probability α, the walk is continued.

Since an item node may connect to both user nodes via the rating relations and also item nodes via the content correlations, there are further two options of the next move considering whether to choose a user node or an item node to visit. We introduce a switch variable s with Bernoulli distribution that if $s = 0$ we walk to a user node, otherwise $s = 1$ we walk to a similar item node. Thus we have:

- If $s = 0$, the runner randomly moves to a user that prefers this item;
- If $s = 1$, the runner randomly moves to a similar item.

Option 2. With probability $1 - \alpha$, restart the random walk from the starting user.

Formally, if the runner is at an item node i and choses to move to a related user

[1] Generally, parameter α is optimized at 0.8 to 0.85.

node, the probability of moving to a certain user $u \sim i$ (here the symbol $\sim$ denotes the relationship of "preferring") is:

$$p(u|i, s=0) = \frac{r_{u,i}}{\sum_{v \sim i} r_{v,i}} \tag{3.3}$$

Otherwise if the runner choses to move to another item node, the probability of moving to a similar item $j \sim i$ (here the symbol $\sim$ denotes the relationship of "similar to") is:

$$p(j|i, s=1) = \frac{sim_{i,j}}{\sum_{k \sim i} sim_{i,k}} \tag{3.4}$$

Combining the equations (3.2) to (3.4), the visiting probability of the bipartite graph vertices is updated as follows.

- For a user node $u \in \mathcal{U}$ that only connects with item nodes via the rating relations, the probability of being visited at the next step is:

$$p^{(t+1)}(u) = \sum_{i \sim u} p(u|i, s=0) p^{(t)}(i) p(s=0) \tag{3.5}$$

- For an item node $i \in \mathcal{I}$ that may connect to both item and user nodes, the visiting probability of next step is:

$$p^{(t+1)}(i) = \sum_{j \sim i} p(i|j, s=1) p^{(t)}(j) p(s=1) + \sum_{u \sim i} p(i|u) p^{(t)}(u), \tag{3.6}$$

where, $p^{(t)}(u)$ and $p^{(t)}(i)$ denote the visiting probabilities of a user u and an item i at the current time/step t respectively; $p^{(t+1)}(u)$ and $p^{(t+1)}(i)$ are their updated probabilities at the next time/step $t+1$.

3.2.2 Graph Ranking and Recommendation Making

Random walks are widely applied to rank graph vertices in accordance with their stationary visiting probability. In our model, we consider the ranking orders of only item nodes since our goal is to identify the most related items as recommendations. Thus we combine Equations (3.5) and (3.6) and obtain the updating formula for only item nodes in the following form:

$$\begin{aligned} p^{(t+1)}(i) &= \sum_{j \sim i} p(i|j, s=1) p^{(t)}(j) p(s=1) \\ &\quad + \sum_{u \sim i} p(i|u) \sum_{j \sim i} p(u|j, s=0) p^{(t-1)}(j) p(s=0) \end{aligned} \tag{3.7}$$

Like all other random walk models, we can present our model in matrix-vector notations. First, the rating matrix $\mathbf{R}$ of size $|\mathcal{U}| \times |\mathcal{I}|$ is imported. We also define a matrix $\mathbf{S}$ of size $|\mathcal{I}| \times |\mathcal{I}|$ denoting the content similarity matrix of items, in which each element is the similarity of two items, that is, $S_{ij} = sim_{i,j}$. Here we allow asymmetric similarity relationship between items. In particular, we let $S_{ii} = 0$ on the diagonal.

For the rating matrix/network, the vertex degrees δ of a user node $u \in \mathcal{U}$ and an item node $i \in \mathcal{I}$ are defined respectively as follows.

$$\delta(u) = \sum_{i \in \mathcal{I}_u} r_{u,i} \tag{3.8}$$

$$\delta(i) = \sum_{u \in \mathcal{U}_i} r_{u,i} \tag{3.9}$$

Two diagonal matrices $\mathbf{D}_u^{|\mathcal{U}| \times |\mathcal{U}|}$ and $\mathbf{D}_i^{|\mathcal{I}| \times |\mathcal{I}|}$ are then defined with the rating vertex degrees of users and items on the diagonal, respectively.

In addition, we also define the vertex degree π of an item $i \in \mathcal{I}$ in the content similarity network of items as follows.

$$\pi(i) = \sum_{j=1}^{|\mathcal{I}|} S_{ij} \tag{3.10}$$

The corresponding diagonal matrix is denoted as $\mathbf{\Pi}_i^{|\mathcal{I}|\times|\mathcal{I}|}$.

Simply, we assume $p(s=1)=\beta \in (0,1)$ thus $p(s=0)=1-\beta$, and define a vector $\mathbf{p}^{(t)} = [p^{(t)}(i_1), p^{(t)}(i_2), \ldots, p^{(t)}(i_{|\mathcal{I}|})]^T$ denoting the probability distribution of all item nodes at a certain time t. Equation (3.7) can then be used to produce the updating formula of visiting probabilities of all item nodes as follows:

$$\mathbf{p}^{(t+1)} = \beta \mathbf{S}^T \mathbf{\Pi}_i^{-1} \mathbf{p}^{(t)} + (1-\beta)\mathbf{R}^T \mathbf{D}_u^{-1} \mathbf{R} \mathbf{D}_i^{-1} \mathbf{p}^{(t-1)} \tag{3.11}$$

As mentioned, it has a probability of $1-\alpha$ to jump back to the starting node at every step such that the distribution will be restored to $\mathbf{p}^{(0)}$ at the starting time. In our model, however, the starting node is only the single active user such that the initial probability of every item node is zero, i.e., $\mathbf{p}^{(0)} = \mathbf{0}$. Due to this reason, we use $\mathbf{q} = \mathbf{p}^{(1)}$ as the restarting condition instead of $\mathbf{p}^{(0)}$. According to Equation (3.11), we have:

$$q_i = \begin{cases} r_{u_0,i}/\delta(u_0), & \text{if } i \sim u_0 \\ 0, & \text{otherwise} \end{cases}, \tag{3.12}$$

where u_0 denotes the starting user, i.e., the request user for whom we make recommendations.

Summing up the above discussions, the stationary probability distribution of items can be obtained by recursively applying the following equation:

$$\mathbf{p}^{(t+1)} = \alpha\Big(\beta\mathbf{A}\mathbf{p}^{(t)} + (1-\beta)\mathbf{B}\mathbf{p}^{(t-1)}\Big) + (1-\alpha)\mathbf{q}, \tag{3.13}$$

where two matrices are defined as follows.

$$\begin{cases}\mathbf{A} = \mathbf{S}^T\mathbf{\Pi}_i^{-1} \\ \mathbf{B} = \mathbf{R}^T\mathbf{D}_u^{-1}\mathbf{R}\mathbf{D}_i^{-1}\end{cases} \tag{3.14}$$

The stationary distribution $\mathbf{p}^*$ will be obtained when Equation (3.13) reaches convergence (proven in (Athreya et al., 1996)).

Let $\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} = \mathbf{p}^{(t-1)} = \mathbf{p}^*$, the following result can be obtained via some algebraic operations:

$$\begin{aligned}\mathbf{p}^* &= (1-\alpha)\Big[\mathbf{I} - \alpha\Big(\beta\mathbf{A} - (1-\beta)\mathbf{B}\Big)\Big]^{-1}\mathbf{q} \\ &\propto \Big(\mathbf{I} - \alpha\beta\mathbf{A} - \alpha(1-\beta)\mathbf{B}\Big)^{-1}\mathbf{q}\end{aligned} \tag{3.15}$$

In the above equation, the positive constant $(1-\alpha)$ is omitted as it does not affect the ranking order of $\mathbf{p}^*$.

Next, all candidate items can be ranked according to the stationary visiting probabilities, and the top-N items are selected as recommendations to the active user. So far, the whole recommendation process of our random walk-based recommendation model is completed for a single request user, as illustrated in Figure 3-1.

3.3 Representations of Item Taxonomy and Folksonomy

The item content similarity correlations constitute an important component in the proposed random walk model in the last section. We have emphasized that the items are usually associated with tree-structured taxonomy attributes, and also possibly a number of folksonomy tags. This section uncovers the difference and relationship between taxonomy attributes and folksonomy tags, and propose a tag-derive semantic analysis for item contents.

3.3.1 Taxonomy Attributes

The item taxonomy information usually consists of various attributes from different perspectives, which construct the top-level branches of item taxonomy trees. For example, a movie in MovieLens.com[1] is associated with descriptions from its *genres*, *directors*, *actors*, etc.; a book selling in Amazon.com[2] contains descriptions of its *category*, *author*, *language*, etc.; while a restaurant in Yelp.com[3] is described in the aspects of *category*, *trading hours*, *cuisine*, etc. We denote these special top-level concepts as the *concept layer* of item taxonomy trees. Every concept attribute dominates a subtree with single or multiple levels of attributes that describe the functionalities of items within this aspect. We call all other attributes except the concept attributes the *attribute layer*, and it is further identified as *level 1 attribute layer*, *level 2 attribute layer*, etc., according to the taxonomy depth. Figure 3-2 and Figure 3-3 show two example taxonomy trees of a movie and a book. It can be found that the depth of the subtree under each concept attribute can be different. Taking the book taxonomy for instance, the branch of *category* has three level attributes, while

[1] https://movielens.org/

[2] http://www.amazon.com/b/ref=txtb_surl_textbooks/

[3] http://www.yelp.com.au/c/sydney/restaurants

the branch of *author* has only one level attributes.

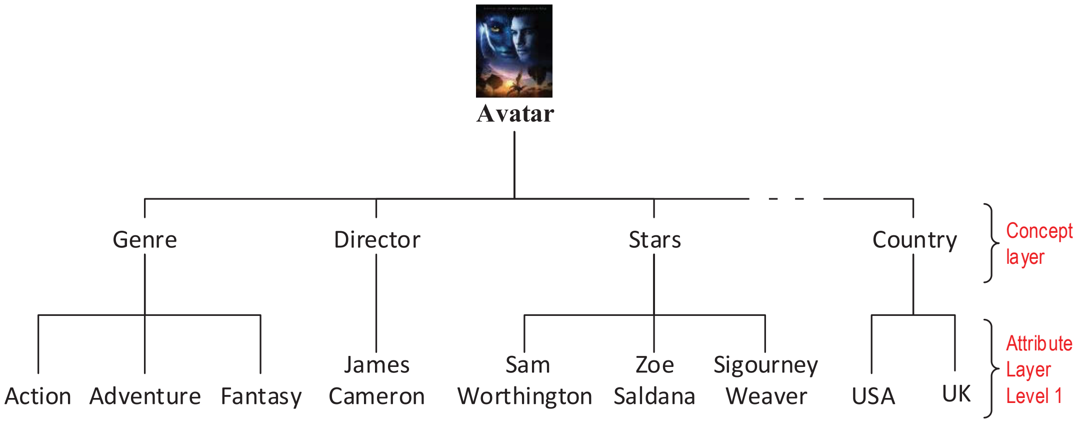


Figure 3-2 An example of movie taxonomy tree in MovieLens

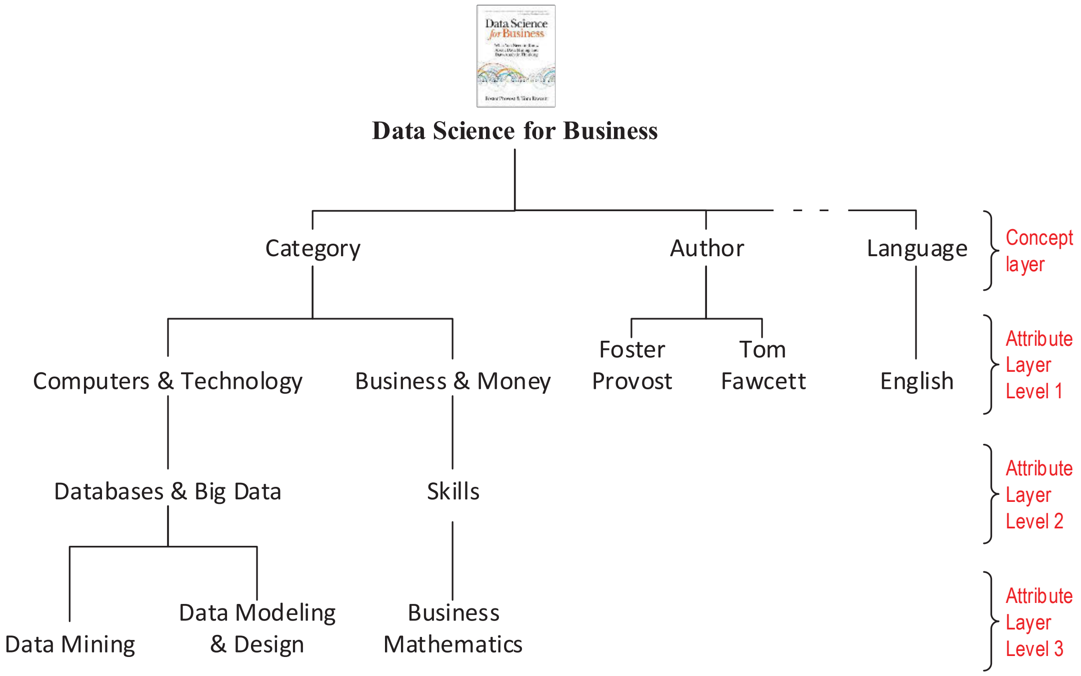


Figure 3-3 An example of book taxonomy tree in Amazon

With all above illustrations, the definition of taxonomy trees and related characteristics are given as below.

Definition 3-1 (Taxonomy Tree). The taxonomy tree of a single item is defined as a directed graph $\Gamma = \{\mathcal{A}, \hookrightarrow\}$ with no cycles, where $\mathcal{A} = \{a_1, a_2, \ldots\}$ is a finite set of taxonomy attributes, and $\hookrightarrow$ is the "parent-child" relationship. For two nodes $a_1, a_2 \in \mathcal{A}$, if $a_1 \hookrightarrow a_2$, then we say a_2 is a child attribute of a_1 and a_1 is the parent attribute of a_2. Note that a node can have multiple (or zero) child nodes, but it can only have one unique parent node. A virtual distinguished root node $root(\Gamma)$ is annotated to represent this item itself, and for any other node $a \in \mathcal{A}$, there exists one and only one path in Γ from $root(\Gamma)$ to a.

Clearly, the direct child nodes of the root node are the concept nodes constricting the concept layer. Suppose that we collect K concept attributes in total of entire item taxonomy trees as the global concept attributes set $\mathcal{C} = \{c_1, c_2, \ldots, c_K\}$. Due to information incompleteness, a single item taxonomy tree may not include all concept attributes. For consistence, if a concept attribute $c \in \mathcal{C}$ does not appear in a single item taxonomy tree, we add this concept node to the tree and let it has no child nodes. Consequently, every single taxonomy tree contains all K concept attributes at the first level.

3.3.2 Folksonomy Tags

Despite the standard taxonomy information, many online systems enable users to freely create their own descriptions for items in the form of single words or short phrases, which are called folksonomy tags. Tags have enriched content information such that other users can better identify items. As well as creating new tags, users also can select existed tags in the systems to classify items. As a result, an item can be assigned with a same tag repeatedly, and the count of co-occurrence of a pair of $\langle item, tag \rangle$ represents the strength of how much this item is relevant to the tag.

We denote the whole tag set as $\mathcal{T} = \{t_1, t_2, \ldots\}$. After pruning some trivial tags that only occurs one time, the metric *tf-idf* (*Term Frequency-Inverse Document Frequency*) (Salton and McGill, 1986) is widely used to represent the item-tag relevance, defined as follows.

$$\textit{tf-idf}(i,t) = \textit{tf}(i,t) \times \log \frac{|\mathcal{I}|}{|\mathcal{I}_t|} \tag{3.16}$$

Here, $\textit{tf}(i,t)$ is the count of a tag t being assigned to an item i by different users, $\mathcal{I}_t$ denotes the set of items that contain tag t. With this setting, the tags appearing in majority of items are penalized as they are thought unable to discriminate items very well.

Compared with taxonomy attributes, folksonomy information is relatively incomplete if users have yet to create sufficient tags for items. We conduct an empirical analysis on the latest Movielens dataset[1]. In this dataset, about thirty percent of items have no tags and for the rest 70% items, we present the item-tag frequency distributions in Figure 3-4. From this figure, we can find the long tail of tag distribution that a large number of items only have a few tags. We also present the item frequency associated for tags in Figure 3-5, which shows that there are also many tags only occurring in a small number of items.

[1] http://grouplens.org/datasets/movielens/

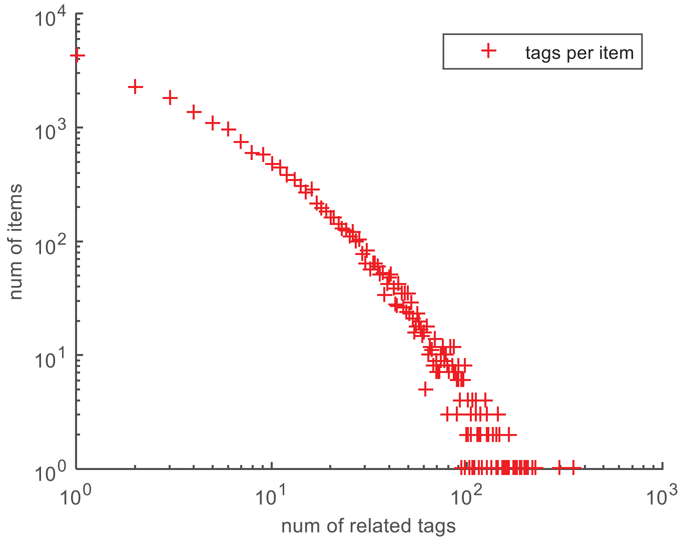


Figure 3-4 The number of associated tags per item (Movielens dataset)

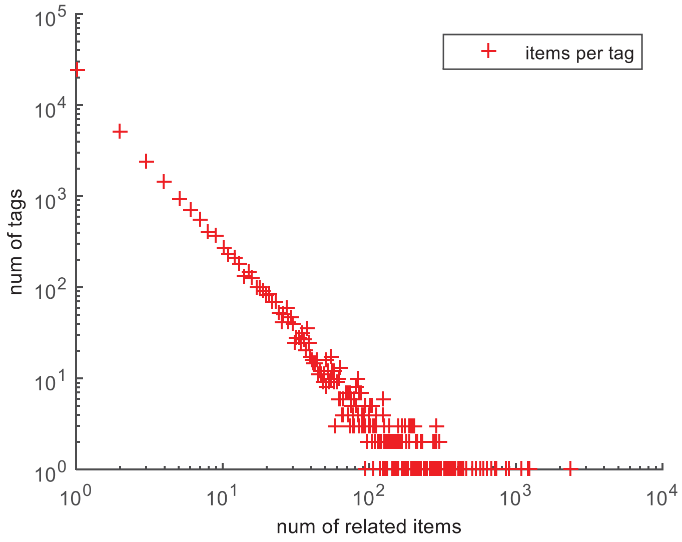


Figure 3-5 The number of associated items per tag (Movielens dataset)

To summarize, taxonomy attributes represent the standard description given by system managers. It is usually inflexible and of fixed and relatively small dimension. In contrast, user-created tags are usually arbitrary, flexible, and of incremental and large dimension. Table 3-1 shows a brief comparison of taxonomy attributes and folksonomy tags.

Table 3-1 A brief comparison of taxonomy and folksonomy

	Taxonomy attributes	Folksonomy tags
Creator:	system managers	users
Standardization:	standard	free
Information completeness:	complete	incomplete
Dimension:	fixed, small	incremental, large

3.3.3 Tag-based Semantic Similarity of Attributes

We can link a pair of an attribute $a \in \mathcal{A}$ and a tag $t \in \mathcal{T}$ via the common items that are associated with both of them, as follows:

$$f(a,t) = \sum_{i:a \in \Gamma_i} \textit{tf-idf}(i,t), \tag{3.17}$$

where Γ_i is the taxonomy tree of a single item $i \in \mathcal{I}$.

A tag, represented by a single word or short phrases, can be seen as an additional description of item content from a particular taxonomy perspective such as movie genres or actors, which is corresponding to a unique concept attribute of the item taxonomy tree. For example, when a user issues a tag "*future*" to a sci-fi movie, this user is very likely using his/her own words to describe the genre of this movie. Thus, the tag "*future*" is actually a user-created attribute from the perspective of *genre*. In our study, we call this relationship as the "domination" of tags, that is, a tag t is dominated by a concept attribute $c \in \mathcal{C}$ if this tag is issued as a description from the perspective of c, notated as $t \prec c$. To identify the domination relationships, we define the follows.

$$t \prec \underset{c \in \mathcal{C}}{\operatorname{argmax}} \max_{a : c \hookrightarrow a} f(a, t) \tag{3.18}$$

Equivalently, the above formula defines that a tag is dominated by the concept that contains the most relevant attributes of this tag. It can be explained by such an example: based on the tag-attribute correlation metric of Equation (3.17), if a tag "*future*" is found most associated with a taxonomy attribute "*sci-fi*", which is a child node of the concept attribute *genre*, then we think this tag is dominated by the concept attribute *genre*, meaning that this tag is a description about the *genre*, rather than other perspectives such as the *directors*, etc.

Figuring out the dominated concepts of tags is helping to find the main characteristics of particular items. For example, summing up the dominated concepts of all related tags of a movie, we can have a glance of that which aspect, such as *genre*, *director* or *cast*, is most discussed or emphasized by people. This finding can be used to conduct more precise comparison between item contents, as discussed later in Section 3.4.

In literature, taxonomy attributes are related and compared based on their semantic meanings. In the tree-based recommender system developed by (Wu et al., 2014b), for example, the so-called "conceptual similarity" is evaluated manually for every pair of attributes by domain experts. The main concern is that extra human resources are needed to manually identify and compare the semantic meanings of multiple levels of attributes. In this chapter, however, we incorporate folksonomy information to derive semantic similarity of taxonomy attributes to avoid manual settings. The following definition is hence given.

Definition 3-2 (Tag-based Semantic similarity). The tag-derived semantic similarity of two taxonomy attributes $a_1, a_2 \notin \mathcal{C}$ (not the concept attributes) is computed by the following equation.

$$ss(a_1, a_2) = \frac{\sum_{t \in \mathcal{T}} f(a_1, t) f(a_2, t)}{\sqrt{\sum_{t \in \mathcal{T}} f(a_1, t)} \sqrt{\sum_{t \in \mathcal{T}} f(a_2, t)}} \tag{3.19}$$

Here the *Cosine* correlation is used to produce a decimal value $ss(a_1, a_2) \in [0, 1]$ as the semantic similarity of the two attributes a_1 and a_2. Equation (3.19) can find the hidden correlations between attributes from different perspectives. For example, we may find the director *James Cameron* is found much related to the genre *Adventure* as these two attributes are often associated with same tags.

3.4 Computing Item Content Similarity

Since tagging information of a particular item may be absent, we compare the taxonomy trees of two items to induce their content similarities. With our data model, the item taxonomy tree contains K top-level subtrees corresponding to several concept attributes $c_1, c_2, \ldots, c_K$. Thus we first compare the subtree under each concept attribute separately, and then aggregate the results as the final output.

For a host item i_{host} and a guest item i_{guest} with two taxonomy trees, denoting $M_{[c_k]}(\Gamma_{host}, \Gamma_{guest})$ the matching result of the subtrees under a particular concept attribute $c_k, k = 1, 2, \ldots, K$, the following aggregating equation is given to compute the overall content similarity $sim(i_{host}, i_{guest})$ between the two items.

$$sim(i_{host}, i_{guest}) = \frac{1}{\sum_{k=1}^{K} w_i} \sum_{k=1}^{K} w_k M_{[c_k]}(\Gamma_{host}, \Gamma_{guest}), \tag{3.20}$$

where $w_k > 0$ is the corresponding weighting of each concept attribute $c_k \in \mathcal{C}$.

3.4.1 Generating Weightings from Tags

To aggregate the comparing result under each concept, a weighting value w_i is introduced to indicate the importance of the attributes under a particular concept c_i tanking place in the comparison of all attributes. Meanwhile, the tag domination information mentioned in last section is imported to automatically generate the weighting set. Intuitively, if people always concern and discuss a particular aspect of items, this hot aspect of content information should be emphasized when comparing item taxonomy attributes. Still taking movie items for instance, if the *actor* information is more concerned by people than the *director* information, it is appropriate to increase the weight of the *actor* information when comparing movies.

Summarizing the tagging information of the entire items, a global weighting set w_{global} of each concept $c \in \mathcal{C}$ is given as follows.

$$w_{\text{global}}(c) = \frac{\sum_{t \prec c} \sum_{i \in \mathcal{I}} tf(i,t)}{\sum_{c \in \mathcal{C}} \sum_{t \prec c} \sum_{i \in \mathcal{I}} tf(i,t)} \tag{3.21}$$

The global weighting set does not consider the distinguish characters of a particular item. For a single item with sufficient tag information, we can propose a private weighting set for this single item, as follows.

$$w_{[i_{host}]}(c) = \frac{\sum_{t \prec c} tf(i_{host},t)}{\sum_{c \in \mathcal{C}} \sum_{t \prec c} tf(i_{host},t)} \tag{3.22}$$

With private weightings, the content similarity computed by Equation (3.20) becomes an *asymmetric* relationship between items.

The choice of private or global weightings can be determined in the following two cases: if the host item has insufficient tag information, global weighting w_{global} is used; otherwise if the host tag has rich tagging information (e.g., more than 50 tags are assigned to this item as our setting in this study), private weighting w_{private} is used.

3.4.2 Subtree Matching Algorithm

A top-down matching manner is proposed to compare the subtrees under a particular concept attribute of the taxonomy trees of two items, which is illustrated in Figure 3-1. It can be found that there are two tasks at each level of matching:

- **Task 1**: To connect every shared common node for the next-level matching process.
- **Task 2**: To determine the best-matched non-common nodes. Deeper-level comparisons are not conducted for these nodes.

The above two tasks are performed at every level of comparison until no common nodes are found. At the current level l, we compute the overall difference of this level comparison as follows.

$$\delta_l = \frac{\lambda_l \sum (1 - ss(\textit{paired noncommon nodes}))}{|\textit{commnon nodes}| + |\textit{paired noncommon nodes}|} \tag{3.23}$$

Here a positive parameter $\lambda \in [0, 1]$ is set to be the weight of each level comparison. It is appropriate to set a decreasing value of λ along with increasing matching levels, e.g., we let $\lambda_l = 1/l$. In Figure 3-6(a), the first level comparison of subtrees under concept c_1, the attribute 1 is the shared common node, while node 2 and node 3 are the paired non-common nodes, so we have $|\textit{common nodes}| = 1$ and $|\textit{paired noncommon nodes}| = 1$ in this level comparison.

Concretely, the best matched pairs of non-common nodes are determined using the following scheme. In this level comparison, suppose the shared parent node of the two subtrees is par, and the child nodes in the two subtrees are denoted as

$child_{[\Gamma_1]}(par)$ and $child_{[\Gamma_2]}(par)$, respectively, then we have the non-common nodes in tree Γ_1 can be denoted as $X_1 = child_{[\Gamma_1]}(par) - child_{[\Gamma_2]}(par)$, similarly, the non-common nodes in tree Γ_2 are $X_2 = child_{[\Gamma_2]}(par) - child_{[\Gamma_1]}(par)$. Clearly, we have $X_1 \cap X_2 = \emptyset$. At each time, we select one node from each side of X_1 and X_2 and assess the matching degree between the two selected nodes based on the semantic similarity proposed in Section 3.3.3. The two nodes with the highest semantic similarity among are matched and the semantic similarity is returned as the matching degree. Recursively, other two most similar nodes are selected from the rest unmatched nodes as a new pair, until no more match can be made, which happens when the semantic similarities of all possible pairs of nodes are zero or there is no more unmatched nodes in either X_1 or X_2. Thus, the maximum number of pairs of non-common nodes is $\min\{|X_1|, |X_2|\}$.

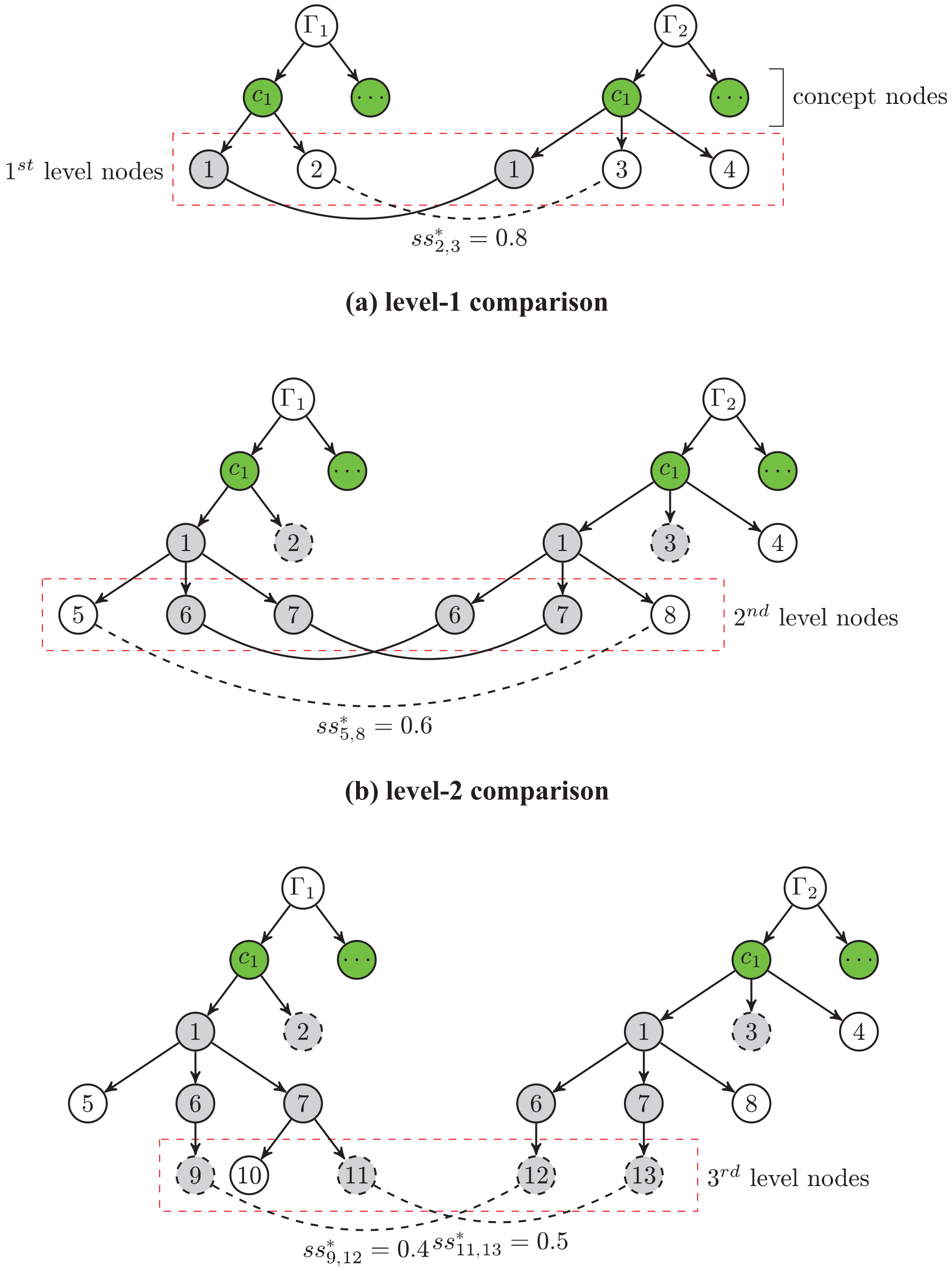
The above discussion of finding the best matched nodes at each time can be expressed as follows.

$$\langle x_1^*, x_2^* \rangle \leftarrow \operatorname*{argmax}_{x_1 \in X_1, x_2 \in X_2} ss(x_1, x_2) \tag{3.24}$$

In Figure 3-6(c), two pairs of nodes $\langle 11, 13 \rangle$ and $\langle 9, 12 \rangle$ are successfully matched as they have relatively higher semantic similarities. Once all level comparisons are completed, the matching result of the two subtrees under the k-th concept attribute c_k can be summarized as follows.

$$M_{[c_k]}(\Gamma_1, \Gamma_2) = \prod_l (1 - \delta_l) \tag{3.25}$$

A recursive function named **subtreeMatch** is proposed to conduct the multi-level comparison process, as shown in Algorithm 3-1.



(a) level-1 comparison

(b) level-2 comparison

(c) level-3 comparison

Figure 3-6 The proposed top-down subtree matching manner

Algorithm 3-1 The subtree matching function

```
Function $M_{[par]} = \mathbf{subTreeMatch}(par, \Gamma_1, \Gamma_2, l)$
Preparation: get current parent node: $par$;
             get current matching level: $l$;
$N_1 \leftarrow child_{[\Gamma_1]}(par)$; //the child nodes in tree $\Gamma_1$
$N_2 \leftarrow child_{[\Gamma_2]}(par)$; //the child nodes in tree $\Gamma_2$
if $N_1 = \emptyset$ or $N_2 = \emptyset$ then
    set $M_{[par]} = 1$; // stop comparison
else
    $N_{co} \leftarrow N_1 \cap N_2$; //shared common nodes
    $X_1 \leftarrow N_1 - N_2$; //uncommon nodes in $\Gamma_1$
    $X_2 \leftarrow N_2 - N_1$; //uncommon nodes in $\Gamma_2$
    initialize $\Delta = 0$; $\eta = 0$;
    while $X_1 \neq \emptyset$ and $X_2 \neq \emptyset$
        update $\langle x_1^*, x_2^* \rangle \leftarrow \underset{x_1 \in X_1, x_2 \in X_2}{\text{argmax}} \; ss(x_1, x_2)$;
        update $\Delta \leftarrow \Delta + (1 - ss(x_1^*, x_2^*))$;
        update $\eta \leftarrow \eta + 1$;
        delete $x_1^*$ from $X_1$;
        delete $x_2^*$ from $X_2$;
    end while
    get $\delta_l \leftarrow \lambda_l \cdot \frac{\Delta}{\eta + |N_{co}|}$; //Equation (3.23)
    if $N_{co} = \emptyset$ then
        set $M_{[child]} = 1$; //no common nodes, stop deeper-level matching
    else
        initialize $TM_{[child]} = 0$;
        for every common node $cnode \in N_{co}$
            update $TM_{[child]} \leftarrow TM_{[child]} + \mathbf{subTreeMatch}(cnode, \Gamma_1, \Gamma_2, l+1)$
        end for
        get $M_{[child]} \leftarrow \frac{TM_{[child]}}{|N_{co}|}$;
    end if
    get $M_{[par]} \leftarrow (1 - \delta_l) M_{[child]}$; //Equation (3.25)
end if
Return $M_{[par]}$;
```

Consequently, each level comparison of the example in Figure 3-6 is obtained as follows. The level-1 difference is $\delta_1 = 0.1$; level-2 difference is $\delta_2 = 0.167$; level-3 difference is $\delta_3 = 0.183$, and the overall semantic similarity from the perspective of the concept attribute c_1 is:

$$M_{[c1]}(\Gamma_1, \Gamma_2) = (1 - \delta_1)(1 - \delta_2)(1 - \delta_3) = 0.61.$$

This result indicates that the overall matching degree of the subtrees of Γ_1 and Γ_2 under the concept c_1 is 0.61, which can be seen as the content similarity of the two corresponding items in terms of the particular aspect of c_1.

3.4.3 Overall Content similarity

As indicated in Equation (3.20), the overall content similarity of two items is induced by aggregating the marching results of all taxonomy subtrees under all concept attributes with private or global weightings. A content similarity induction algorithm is hence proposed as below.

So far, we can establish the overall content similarity correlations between the entire items based on both taxonomy and folksonomy information. This result will be imported as an important component into the hybrid random walk model proposed in Section 3.2 to generate recommendations for users.

Algorithm 3-2 Content Similarity Induction

```
Algorithm Content Similarity Induction
Data:   two items $i_1$, $i_2$;
Return: content similarity $sim(i_1, i_2)$;
get taxonomy tree $\Gamma_1$ of $i_1$; get taxonomy tree $\Gamma_2$ of $i_2$;
for $k = 1$ to $K$
    get current concept attribute $c_k$;
    $sc_{[c_k]}(\Gamma_1, \Gamma_2) \leftarrow \mathbf{subTreeMatch}(c_k, \Gamma_1, \Gamma_2, 1)$;
end for
get tag set $\mathcal{T}_{host} \leftarrow \mathcal{T}_{i_1}$;
if $|\mathcal{T}_{host}| < \min_{\text{tag}}$ then
    for $k = 1$ to $K$
        $w(c_k) \leftarrow$ Equation (3.21);
    end for
else
    for $k = 1$ to $K$
        $w(c_k) \leftarrow$ Equation (3.22);
    end for
end if
Return $sim(i_1, i_2) \leftarrow$ Equation (3.20);
```

3.5 A Numeric Example

In this section, we use a numeric example to illustrate how the proposed random walk-based recommendation approach is implemented step by step based on diverse input information such as item attributes, tags and ratings. In this example, we are given two users and four items. The users have rated some items and the items are associated with standard tree-structured taxonomy attributes and free tags.

3.5.1 Item Overall Content Similarity

At first, we establish the semantic similarities of taxonomy attributes from the tag information based on Equation (3.19), as well as a global weighting set for different concept attributes by Equation (3.21). We then apply Algorithm 3-2 to compute the overall similarity of each (directed) pair of items.

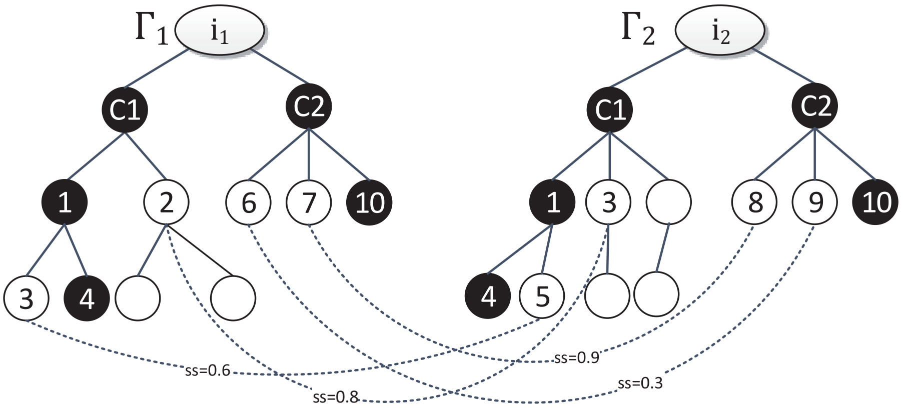


Figure 3-7 The complete tree matching result between item 1 and item 2

The full taxonomy trees of items i_1 and i_2 are illustrated in Figure 3-7 as an example. Assuming that item i_1 has sufficient tags and its private weightings of concept attributes are computed as $w(c_1) = 0.4$ and $w(c_2) = 0.6$ based on Equation (3.22), while m_2 has few tags so the global weights $w(c_1) = 0.7$ and $w(c_2) = 0.3$ computed by Equation (3.21) are used. The asymmetric content similarities between them are then aggregated as follows, referring to Equation (3.20).

$$\begin{aligned} sim(i_1, i_2) =& 0.4 \times (1 - \frac{1-0.8}{2\times 1}) \times (1 - \frac{1-0.6}{2\times 2}) \quad +0.6 \times (1 - \frac{1-0.9+1-0.3}{3\times 1}) \\ =& 0.764; \\ sim(i_2, i_1) =& 0.7 \times (1 - \frac{1-0.8}{2\times 1}) \times (1 - \frac{1-0.6}{2\times 2}) \quad +0.3 \times (1 - \frac{1-0.9+1-0.3}{3\times 1}) \\ =& 0.787. \end{aligned}$$

Once the overall content similarity of each pair of the four items is obtained, a similarity network is constructed to connect the four items, as illustrated by the dashed lines in Figure 3-8.

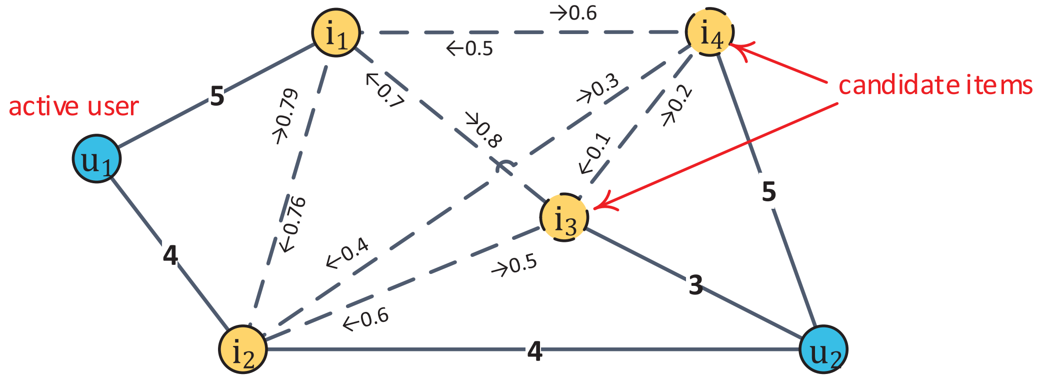


Figure 3-8 The constructed user-item bipartite graph

3.5.2 Recommendations based on Bipartite Random Walks

By incorporating both of the user-item rating relations and the content similarity network of items, a bipartite graph consisting of user and item nodes is constructed as in Figure 3-8. In our example, the user u_1 is assumed to be the request user for whom we want to recommend an unknown item from i_3 and i_4. The random walk model proposed in last subsection is performed and the visiting probability distribution of item nodes is updated at every step by applying Equation (3.11). With the initial parameter setting of $\alpha = 0.8$ and $\beta = 0.5$, the updated distribution vector $\mathbf{p}$ at every step is collected in Table 3-2., which shows that $\mathbf{p}$ reaches convergence after 50 steps of random walking. Comparing the visiting probabilities of the target items i_3 and i_4, we find that i_4 acquires higher visiting probability when walking depths is longer than 10. The converged visiting probabilities are $p(i_3) = 0.149$ and $p(i_4) = 0.152$ finally. Thus the item i_4 should be more likely accepted by the request user u_1 than the item i_3. We select i_4 as the recommendation for the request user u_1.

Table 3-2 Visiting probability at every step of random walks (only item nodes)

step/time	0	1	2	3	4	5	10	20	>50
$p(i_1)$	0.556	0.110	0.359	0.233	0.322	0.284	0.330	0.349	0.355
$p(i_2)$	0.444	0.098	0.319	0.233	0.292	0.277	0.316	0.337	0.344
$p(i_3)$	0	0.173	0.056	0.130	0.097	0.125	0.133	0.146	0.149
$p(i_4)$	0	0.119	0.066	0.124	0.097	0.125	0.134	0.148	0.152
sum	1	0.5	0.8	0.72	0.808	0.811	0.913	0.980	≈ 1

To summarise, the proposed hybrid recommendation approach is completed for this numeric case by using the unique bipartite graph random walk model in Section 3.2 and the content similarity induction algorithm proposed in Section 3.4.

3.6 Experiments

Empirical experiments are conducted to compare the performance of our proposed approach with other benchmark models. The "Movielens-HetRec 2011" dataset (Cantador et al., 2011) is used for our experiments. This dataset is an extension of the standard MovieLens10M dataset by importing rich content attributes of movies in different aspects such as genres, actors, directors, etc.

3.6.1 Experiment Setup

There are in total 2113 users and 10109 movies in the whole dataset. The movies are associated with taxonomy attributes from five perspectives of their *genre*, *director*, *actor*, *production country* and *location*. Accordingly, there are five concept attributes in the top level of the movie taxonomy tree. With necessary data cleaning, there are in total 46720 folksonomy tags assigned to these movies, with an average per movie of 7.1 tags.

The selected dataset has been widely applied for testing CF approaches due to the dense rating data. There are about 855,000 rating records, giving an average per user of 405 ratings. With so rich rating data, the sparseness problem of CF is not evident, so we dilute the rating data and test recommendation approaches under different sparsity levels. The dilution is conducted as follows: the whole rating data is selected as Group 1, then from Group 1, we randomly select half of the ratings to build a Group 2 dataset, and then select half of Group 2 to build Group 3, and so on. We conduct this procedure five times to generate six data groups. The sparsity level of each group is 96%, 98%, 99%, 99.5%, 99.7% and 99.9%, respectively. We conduct separate experiments with each of the six groups, and apply five-folder validations.

Some related studies are selected for comparison. First, pure **CF** approach (Resnick et al., 1994) is implemented. Besides, the **ItemRank** model proposed by Gori et al. (2007) is selected as a pure random walk-based model. A multipartite random walk model inspired by (Cheng et al., 2007) is also implemented, denoted as **MultiRW** for short. Moreover, two hybrid recommendation approaches, the tree-based algorithm of (Wu et al., 2014b) and the combined model of (Liang et al., 2010), that both incorporate taxonomy and folksonomy information are compared. We name them **TreeSim** and **TFCB**, respectively. Our hybrid recommendation approach is denoted as **TFRW** (Taxonomy and Folksonomy-integrated Random Walk model) for short.

From the perspective of information fusion, the selected data set contains three types of input information: user-item ratings, item taxonomy attributes and item folksonomy tags. Table 3-3 distinguishes which information is utilized as input resource for each compared single or hybrid recommendation approach.

Table 3-3 The involved input information of each compared approach

Information	CF	ItemRank	MultiRW	TreeSim	TFCB	TFRW(ours)
Rating	√	√	√	√	√	√
Taxonomy			√	√	√	√
Folksonomy				√	√	√

We select two metrics to evaluate the recommendation performance. First, the recommendation coverage is used to compare the success rate a recommendation approach. Second, the NDCG metric is used to compare the ranking accuracy of the generated recommendation list of each approach. Error-based metrics like MAE and RMSE are not applicable because not all approaches output rating predictions.

We implement each approach on all the six data groups with different rating sparsity levels. In particular, the restarting parameter α for random walk models, ItemRank, MultiRW and TFRW, are all set to be 0.8. The parameter β for TFRW is set to 0.5 initially, such that we have equivalent choices of walking via a rating relation or a content similarity relation in the bipartite graph, referring to the two routes in the example of Figure 3-1.

3.6.2 Performance Comparison

Figure 3-9 demonstrates the coverage variation of each approach with increasingly rating sparsity. This result shows that our approach TFRW maintains the best and stable coverage rate compared to all other approaches. In general, recommendation coverage of a particular approach falls when ratings become sparser. For the first group dataset with the richest ratings (averagely 405 ratings per user), it can be seen that every approach performs perfectly with high coverage of almost 100%. When the rating data becomes sparser, single approaches CF and ItemRank lose their coverage rates sharply, and hybrid approaches are also impacted at different degrees.

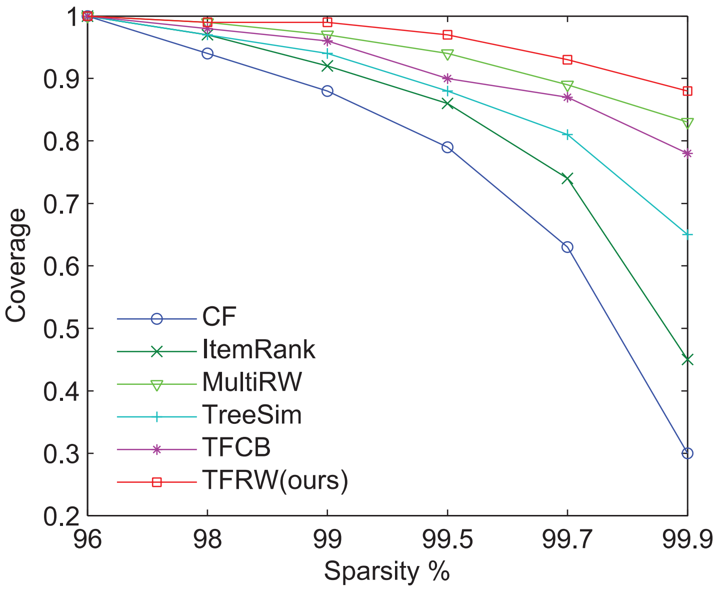


Figure 3-9 Comparison of the coverage rates under different sparsity levels

We also have the following two findings from Figure 3-9. First, hybrid models, MultiRW, TreeSim, Combine and TFRW, can improve recommendation coverage compared to single resource-based approaches CF and ItemRank. This demonstrates the success of the incorporation of content information to alleviate the sparseness problem of rating data. Second, we find that the random walk-based approaches TFRW and MultiRW generally perform higher coverage rates than KNN-based approaches such as TreeSim and TFCB. The reason may be that there are insufficient data in sparse environment for KNN approaches to identify precise neighbour users or items.

The comparison NDCG measurements is presented in Table 3-4, which evidently shows that the proposed approach TFRW reaches the best performance in most data groups, especially for the sparser groups. Among other compared approaches, CF acquires high accuracy when ratings are as rich as in the first two groups, while it loses the superiority quickly in sparser data groups. The other single model ItemRank also suffers the sparseness problem heavily in the sparse groups. In contrast, hybrid approaches are able to maintain relatively higher ranking accuracy by incorporating item taxonomy and/or folksonomy information when rating data are sparse, referring to the performance of MultiRW, TreeSim, TFCB and TFRW. In particular, the superiority of our approach compared to other two hybrid models TreeSim and TFCB with exactly the same input resources indicates that the proposed random walk model is more effective to incorporate item taxonomy and folksonomy for improving recommendations.

Table 3-4 Comparison of the NDCG* scores

Group	Sparsity	CF	ItemRank	MultiRW	TreeSim	TFCB	TFRW(ours)
#1	96%	**0.737**	0.633	0.659	0.727	0.711	0.733
#2	98%	0.647	0.565	0.555	0.597	0.577	**0.659**
#3	99%	0.463	0.467	0.507	0.491	0.487	**0.583**
#4	99.5%	0.505	0.419	0.519	0.479	0.459	**0.617**
#5	99.7%	0.356	0.286	0.376	0.434	0.406	**0.542**
#6	99.9%	0.128	0.118	0.276	0.326	0.316	**0.436**

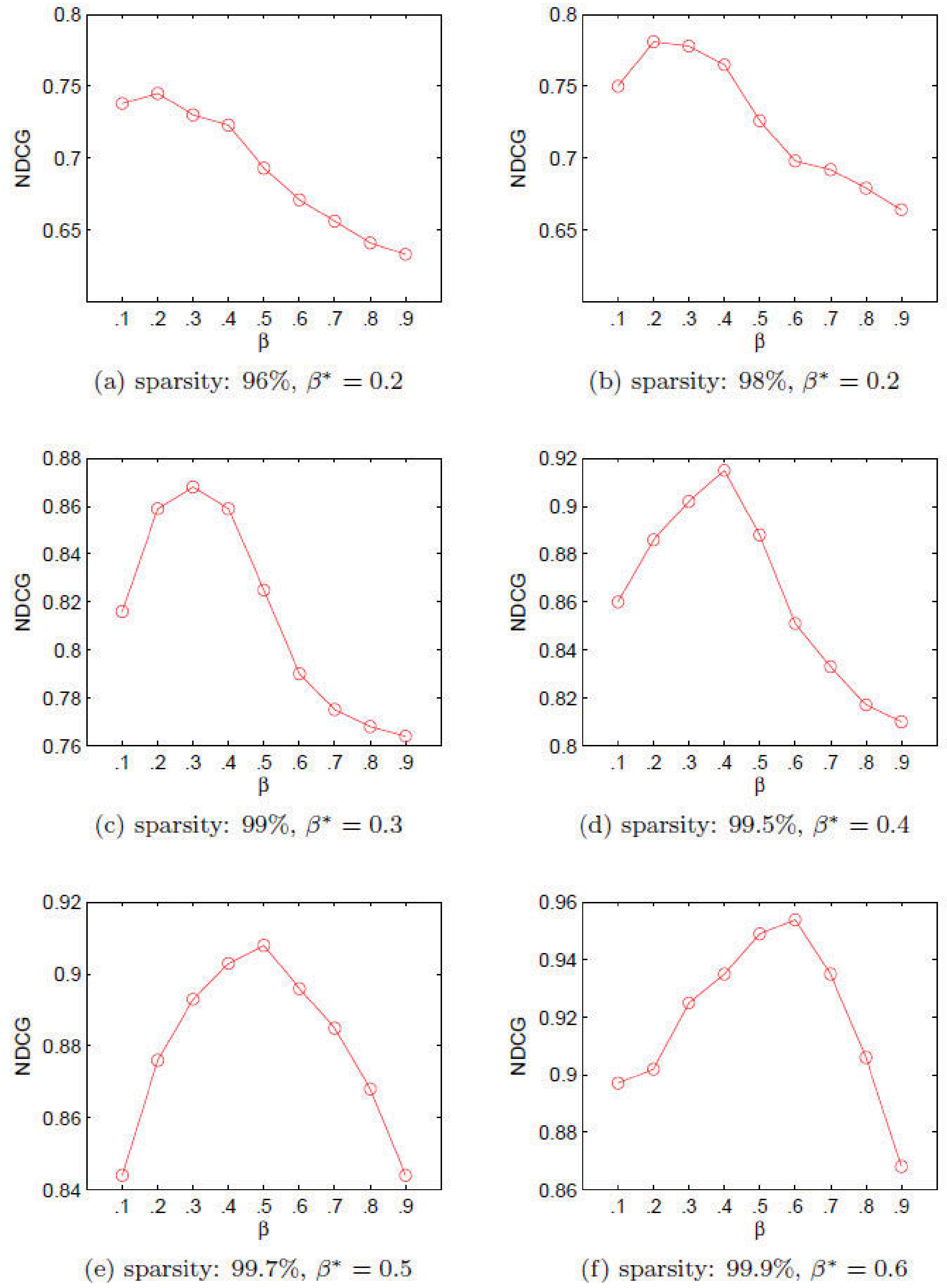
*NDCG@10**, bold** typeset indicates the best result under the current sparsity level.

Summarizing all above comparisons, we can conclude that the proposed hybrid recommendation approach is effective in improving recommendation performance in terms of both the success rates and ranking accuracy, especially for sparse environments with no sufficient explicit ratings.

3.6.3 Parameter Optimization

In the proposed random walk model, a special variable $s \sim \mathrm{Bernoulli}(\beta)$ is introduced to determine whether the runner that currently locates at an item node should move to a user node following the user-item preference relations (can be seen as a CF characteristic) or move to another item node following the content similarity correlations (can be seen as a content-based characteristic), referring to Equation (3.3) and Equation (3.4). The Equation (3.11) indicates that a higher value of β means that the runner has a higher probability to move to item nodes via item content similarity correlations such that the content-based characteristic is more emphasized in the hybrid recommendation framework. Inversely, a lower value of β indicates that the CF side idea is more considered. In our experiment, we tune the value of parameter β from 0.1 to 0.9 to seek for the best performance for different data groups.

Figure 3-10 illustrates the ranking accuracy varying with the values of the parameter β for different data groups. It can be found that the optimized value of β shifts with data sparsity. First, for the data groups with denser ratings (for example, the first two groups), highest performance is achieved when β is relatively small (e.g., 0.2). This demonstrates that the proposed hybrid recommendation approach should emphasize the CF idea when rating data is sufficient enough to sketch precise user profile. In contrast, in sparse environments such as in the sixth data group, β should be tuned higher to consider more of the item content information such as the taxonomy attributes and folksonomy tags for recommendation making.



(a) sparsity: 96%, $\beta^* = 0.2$

(b) sparsity: 98%, $\beta^* = 0.2$

(c) sparsity: 99%, $\beta^* = 0.3$

(d) sparsity: 99.5%, $\beta^* = 0.4$

(e) sparsity: 99.7%, $\beta^* = 0.5$

(f) sparsity: 99.9%, $\beta^* = 0.6$

Figure 3-10 Adjusting the model parameter for different data groups

3.7 Summary

This chapter proposes a hybrid recommendation approach by performing graph ranking on a user-item bipartite graph. This approach incorporates various input information including user-item ratings, item taxonomy attributes and folksonomy tags. This work is constituted by two main components. First, we develop a random walk model on a bipartite graph involving both users and items as the vertices and their connections in terms of user-item ratings and item-item similarity correlations as the edges. Second, we present a tree matching model to incorporate both taxonomy and folksonomy information of items to establish content similarity correlations between items.

In the proposed random walk model, the runners surf between users and items either following the user-item rating relations or the item-item similarity correlations. The former route is similar to the idea of CF approaches, while the latter one is similar to the idea of content-based recommendation approaches; thus the random walk model can be seen as a hybrid recommendation approach that incorporates CF and CB ideas. In addition, a random variable s is introduced to switch walking between the two options. Therefore, our model is considered to provide a natural parameter to adjust the balance between the CF characteristic and CB characteristic.

To incorporating increasingly diverse content information in terms of both taxonomy attributes and folksonomy tags for our hybrid recommendation approach, we proposed a tree matching algorithm to produce more precise similarity measurement of items. Differing with other studies, the tag information plays an important role in our content analysis. In this study, we utilize the tag information to identify the semantical similarity between taxonomy attributes, which was often manually evaluated by human experts in previous studies. Another advantage of using tags to estimate attribute similarity is that we can discover the possible stringing connections between two attribute that are seemingly unrelated at all in the taxonomy tree. This study also use tags to sketch the most “featured” attributes of items, which can be used for producing more precise similarity measurement between items. Still taking movie items for instance, if a particular movie is found most discussed by people about its cast members, it is appropriate to elevate the weight of movie actors when seeking other similar movies. In brief, the tag information is well utilized in the study of this chapter to (1) generate the semantical similarity of attributes and (2) identify the distinctive attributes of items, both of which are the essences of the proposed item overall content similarity induction algorithm.

A numerical case is given as an example to show all the steps of implementing our bipartite random walk model for recommendations. The approach has also been well tested and compared with existing single or hybrid approaches on a public dataset of Movielens. The results support the advantages of our model in alleviating the sparseness problem with robust and high performance. Particularly, the parameter β of our bipartite random walk model, which adjusts the balance of CF idea and CB idea in the hybrid recommendation approach, has been finely tuned with different data groups. The results provide a general guideline of leveraging the CF and content-based characteristics for hybrid recommendation approaches in different situations.

CHAPTER 4. A SOCIAL NETWORK-BASED RECOMMENDER SYSTEM VIA MULTIGRAPH RANKING

4.1 Overview

With the recent outgrowth of social networking tools, user-user correlations are increasingly emerging as another facet of resource for amalgamating peoples' opinions and naturally being incorporated to overcome the rating sparseness problem of CF-based recommender systems. In recent years, various types of user-user connections have been incorporated for making recommendations separately, including both the explicit social relations (Yang et al., 2012) and also more implicit user correlations (Shambour and Lu, 2012). Despite the success of each single approach, a new challenge is that people are often surrounded in multiple types of relations simultaneously. This complex relationship of users is usually referred to in the area of (social) network analysis as multi-relational/layer/dimensional networks (Salehi et al., 2014). In recommender systems, however, there are few studies that seek to handle such multiple user networks for enhancing recommendations (Kazienko et al., 2011). Motivated by the need to attempt this, this chapter presents a multigraph model to involve users and their various types of relations, and develops a multigraph ranking model to determine potentially interested items for users.

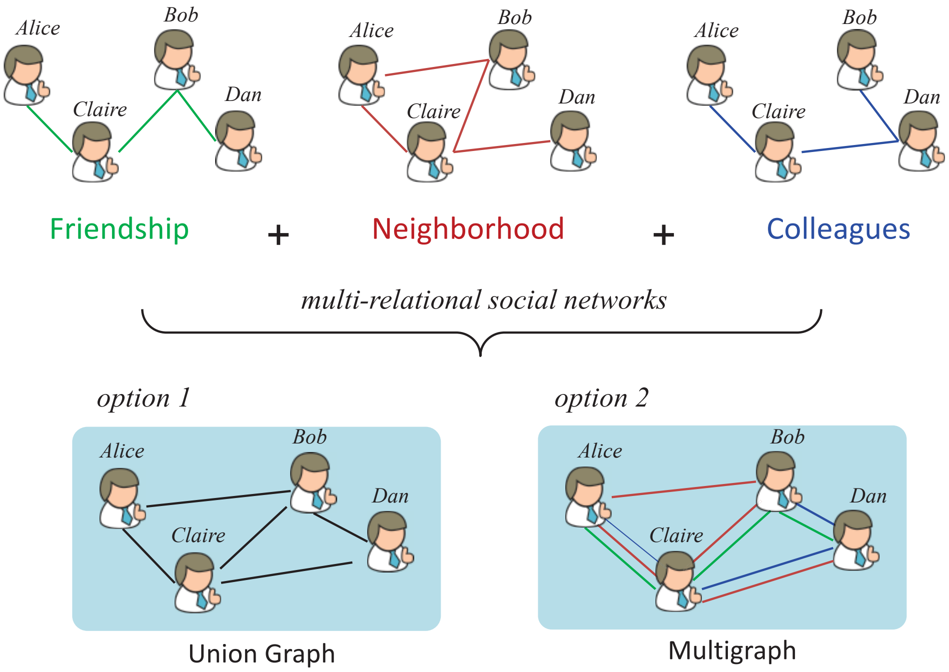


Figure 4-1 Modelling multi-relational social networks using union graph vs. multigraph

We present the multi-relational social networks of users using an example in Figure 4-1, which illustrates the generation of a multigraph for four persons who are connected by three types of relationships: friendship, neighbourhood and colleagues. A union (simple) graph is also constructed by aggregating the relations of each pair of persons into a union relation as the comparison. Compared to the union graph, the multigraph model preserves the structural information of each single network with no information loss (Gjoka et al., 2011).

This chapter proposes a multigraph ranking model to identify the nearest neighbour users in a multi-relational social network to produce KNN-based recommendations like conventional CF-based recommender systems. This work has two main components and innovations as follows. (1) It presents a random walk-based social network propagation model to infer indirect relations in a single social network. (2)

Based on the propagated single social networks, a multigraph ranking model is proposed to establish the overall nearest neighbours of a particular user. In other words, the social network propagation model is used as a preliminary process to enrich the data of original user relations, while the multigraph ranking model undertakes the next step to identify the overall closeness for users.

The rest of this chapter is organized as follows. A random walk-based social network propagation model is first presented in Section 4.2. In Section 4.3, a multigraph model constructed by different user networks is defined. In particular, we propose an inter-network measurement to assess the structural diversity between single social networks, which can be used for pre-screening the diverse input resources. Next, a multigraph ranking model is developed in Section 4.4, where the regularization framework of simple graph ranking theory is employed and improved. In Section 4.5, empirical experiments are conducted with two real-world datasets including the product review data of Epinions.com and the social music sharing data of Last.fm. The results demonstrate the superiority of our approach in terms of both recommendation coverage and accuracy. We then summarize our findings in the last section.

4.2 Single Social Network Propagation

The transitivity of social relations such as "friendship" and "trust" has been adopted as a main topic in existing social network-based recommendation approaches (Golbeck, 2006; Massa and Avesani, 2007; Shambour and Lu, 2012). In particular, we apply the social transitivity for not only the explicit social relations but also for the general user-to-user correlations, e.g., the rating similarity correlation of users. It is well known that rating similarity of two users is computed from the common ratings to their both rated items such that the result will be unavailable or inaccurate if they share none or insufficient common items. However, we can infer the "indirect" similarity of them via a third user if both of them are found having similar ratings to the third user. Based on this assumption, this section proposes a random walk-based propagation model for each collected user relational network constructed from explicit social relations or implicit correlations, as reviewed in Section 2.2.2.

Formally, a single social network of the entire users in a recommender system is represented by a graph defined as follows.

Definition 4-1 (Single social network). In a recommender system, a social network of users is denoted as a graph $G=\{\mathcal{U},E\}$, where $\mathcal{U}$ is the population of users handled as the graph vertices and $E\subseteq\mathcal{U}\times\mathcal{U}$ is a set of edges where a directed pair of users $[u_1,u_2]\in E$ indicates there is a connection from u_1 to u_2 in the graph.

As reviewed, user-to-user networks may have different forms such as directed or undirected, weighted or binary relations. Uniformly, an undirected relation can be decomposed to two one-way directed relations. Also, the binary social networks can be treated as special weighted networks with same weightings for all edges. For a weighted network, the edges are natively associated with a weighting function $w:E\to[0,1]$ after normalization. For a binary network, a weighting function is assumed to be $w:E\to 1$ that gives a constant weight to every visible edge.

Correspondingly, we denote a *weighting adjacency matrix* $\mathbf{W}$ of size $|\mathcal{U}|\times|\mathcal{U}|$ for a social graph $G=\{\mathcal{U},E\}$ with each element $\mathbf{W}_{ij}=w([u_i,u_j])$ if the directed path $[u_i,u_j]\in E$ and 0 otherwise. In our study, we always use $[u_i,u_j]$ to denote a directed vertex-to-vertex path from the vertex u_i to another vertex u_j.

4.2.1 Performing a Single Random Walk

To infer the missing edges in a single graph, we use a "walk and select" manner following the work of Jamali and Ester (Jamali and Ester, 2009) to perform random walks on this graph. This unique model is elaborated in the following steps.

Given a source user $u_{\mathbf{sou}}$ and a target user $u_{\mathbf{tar}}$, for whom we know $[u_{\mathbf{sou}}, u_{\mathbf{tar}}] \notin E$, i.e., there is no edge from $u_{\mathbf{sou}}$ to $u_{\mathbf{tar}}$ in the original graph. To infer the indirect relation and estimate an appropriate value of the weighting $\tilde{w}([u_{\mathbf{sou}}, u_{\mathbf{tar}}])$, a single random walk is performed starting from the source user $u_{\mathbf{sou}}$. Assuming a runner randomly moves from one node to another node each time in the graph, and at a time t it has moved to a certain user u_i. At this time, the runner can choose to keep moving to another node or otherwise to terminate this walk. This process refers to the *walking* part in our model. If the runner decides to terminate the walk, the value of $w([u_i, u_{\mathbf{tar}}])$ is just returned as the prediction of $\tilde{w}([u_{\mathbf{sou}}, u_{\mathbf{tar}}])$, and this process is called the *selection* part. To summarize, the walking mana performs the search of similar users in the social network while the selection manner polls the suggestions of the similar user's closeness to the target user.

Concretely, the two options at time t when the runner is located at user u_i are as follows.

Option 1. With probability ϕ_t, the random walk is terminated, and $w([u_i, u_{\mathbf{tar}}])$ is returned as the result of this single walk. If u_i is neither linked to $u_{\mathbf{tar}}$ such that $w([u_i, u_{\mathbf{tar}}])$ is not available, zero is returned.

Option 2. With probability $1-\phi_t$, the walk is continued, and another node connected by u_i will be randomly reached at the next step.

Here, the notation ϕ_t is the termination probability with regarding to the walking time/step t. Similar to all other random walk models, the transition probability of moving from the current user u_i to another user u_j is given by:

$$p(u_j|u_i) = \frac{W_{ij}}{d(u_i)}, \tag{4.1}$$

where $d(u_i)$ denotes the vertex degree of user u_i, which is defined as follows.

$$d(u_i) = \sum_{j=1}^{|\mathcal{U}|} W_{ij} \tag{4.2}$$

A diagonal matrix $\mathbf{D}^{|\mathcal{U}|\times|\mathcal{U}|}$ can be constructed with $\mathbf{D}_{ii} = d(u_i)$ on the every diagonal place. We define a row vector $\mathbf{p}^{(t)}$ as the probability distribution of all users at time t, that is, the i-th element $p_i^{(t)}$ is the probability of that the user u_i is being visited at time t. The transition matrix of the walking process is the same of traditional random walk models as follows.

$$\mathbf{T} = \mathbf{D}^{-1}\mathbf{W} \tag{4.3}$$

If the random walk keeps moving from the current time t to the next time $t+1$, the distribution vector will be updated once as follows.

$$\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} \times \mathbf{T} \tag{4.4}$$

Note that the above equation has a different form as in traditional random walk models because we use a row vector $\mathbf{p}$ instead of a column vector to record the probability distribution.

So far, we have completed a single random walk and returned a suggestion of the indirect edge from the source user to the target user. For precise predictions, we can start a number of random walks separately from the source user and aggregate the returned result as the final results, which is detailed in the following section.

4.2.2 Performing More Random Walks

More random walks are started from the source user to seek more suggestions of the prediction of $\tilde{w}([u_{\mathbf{sou}}, u_{\mathbf{tar}}])$. We define a new variable s as the total walking length of a terminated single walk, for which we can obtain a distribution vector $\mathbf{p}^{(s)}$ by recursively applying the updating Equation (4.4). We also denote a column vector $W_{:\mathbf{tar}}$, i.e., the corresponding column in the weighting matrix $\mathbf{W}$ for the garget user $u_{\mathbf{tar}}$. Clearly, $W_{:\mathbf{tar}}$ actually represents the weightings of the in-linked edges for the target user. The expectation of returned result of a single random walk terminated at time s should be as follows.

$$\tilde{w}([v_{\mathbf{sou}}, v_{\mathbf{tar}}])|s = \mathbf{p}^{(s)} W_{:\mathbf{tar}} \tag{4.5}$$

Aggregating all random walks that starts from the source user, the global expectation of returned values will be:

$$\begin{aligned}\tilde{w}([v_{\mathbf{sou}}, v_{\mathbf{tar}}]) =& p(s=1)\mathbf{p}^{(1)} W_{:\mathbf{tar}} + p(s=2)\mathbf{p}^{(2)} W_{:\mathbf{tar}} \\ &+ p(s=3)\mathbf{p}^{(3)} W_{:\mathbf{tar}} + \dots\end{aligned} \tag{4.6}$$

At beginning, the starting distribution is $\mathbf{q} = \mathbf{p}^{(0)}$. As all random walks start from the particular source user $u_{\mathbf{sou}}$, $\mathbf{q}$ actually has only one positive element $q(v_{\mathbf{sou}}) = 1$ and all others are zeros. Combining Equation (4.4) and (4.6), we have the following simplification.

$$\begin{aligned}\tilde{w}([v_{\mathbf{sou}}, v_{\mathbf{tar}}]) =& \phi_1 \mathbf{q} \mathbf{T} W_{:\mathbf{tar}} + (1-\phi_1)\phi_2 \mathbf{q} \mathbf{T}^2 W_{:\mathbf{tar}} \\ &+ (1-\phi_1)(1-\phi_2)\phi_3 \mathbf{q} \mathbf{T}^3 W_{:\mathbf{tar}} + \dots \\ =& \sum_{t=1}^{\infty} \phi_t \prod_{i=1}^{t-1} (1-\phi_i) \mathbf{q} \mathbf{T}^t W_{:\mathbf{tar}} \\ =& \sum_{t=1}^{\infty} \psi(t) \mathbf{q} \mathbf{T}^t W_{:\mathbf{tar}}\end{aligned} \tag{4.7}$$

In the above equation, a new notation $\psi(t)$ is introduced to denote the probability of that a single random walk is terminated at time t, computed as follows.

$$\psi(t) = p(s = t|\phi) = \phi_t \prod_{i=1}^{t-1}(1 - \phi_i) \tag{4.8}$$

The Formula (4.7) predicts the indirect connections between two particular users. From the perspective of the whole network, we obtain an inferred weighting matrix $\tilde{\mathbf{W}}$, where $\tilde{W}_{ij}$ is the predicted weighting value of the edge from the user u_i to the user u_j.

$$\tilde{\mathbf{W}} = \sum_{t=1}^{\infty} \psi(t)\mathbf{T}^t\mathbf{W} \tag{4.9}$$

We can prevent walking that is too long term by adjusting the termination parameter ϕ. Based on the idea of "six degrees of separation" (Jamali and Ester, 2009; Milgram, 1967), most users will be reachable with a walk that is at most six steps in length. Hence, if a walk has reached six steps, we force it to terminate, that is, let $\phi_6 = 1$. Thus the Equation (4.9) can be replaced by the following form.

$$\tilde{\mathbf{W}} = \sum_{t=1}^{6} \psi(t)\mathbf{T}^t\mathbf{W} \tag{4.10}$$

Table 4-1 An example setting of the termination parameter

t	1	2	3	4	5	6
ϕ_t	0.5	0.6	0.7	0.8	0.9	1
$p(s = t\|\phi)$	0.5	0.3	0.14	0.048	0.0108	0.0012
$p(s \leq t\|\phi)$	0.5	0.8	0.94	0.988	0.9988	1

Furthermore, it is appropriate to assume the termination probability becomes higher when the random walk goes to deeper levels, that is, the parameter ϕ increases with time t. Simply, we let ϕ_t increase from 0.5 to 1 in the first six steps, as shown in Table 4-1, in which the distribution of walking length ψ is also computed. We find that most (80%) random walks will stop at the first two steps. Based on the Pareto Principle (also known as the 80-20 rule), a fast and approximate solution of (4.10) is obtained based only on the first two steps, as follows.

$$\tilde{\mathbf{W}} = 0.5\mathbf{TW} + 0.3\mathbf{T}^2\mathbf{W} \tag{4.11}$$

Note that we only need to predict the indirect relations, so the final enriched weighting matrix $\hat{\mathbf{W}}$ after propagation is:

$$\hat{\mathbf{W}} = \mathbf{W} + (\mathbf{J} - \mathbf{H}) \circ \tilde{\mathbf{W}} \tag{4.12}$$

where $\circ$ is entry-wise production, $\mathbf{J}$ is a matrix with all element equalling to one with size $|\mathcal{U}| \times |\mathcal{U}|$ and $\mathbf{H}$ is the adjacent matrix of the graph with $H_{ij} = 1$ if there is an edge from the i-th user u_i to the j-th user u_j and $H_{ij} = 0$ otherwise.

We conduct an empirical analysis to evaluate the precision of the alternative calculation of (4.11) on a real-world dataset of the Last.fm (Cantador et al., 2011). Tow-relational networks of users are collected from the dataset. One is the explicit *Friendship* network of users. The other is a *preference similarity* network derived from the listening counts of users. Each network is propagated using the full propagation equation (4.10) and the alternative propagation equation (4.11), respectively. The experimental results are compared in Table 4-2. First, for the similarity network that already has denser original relations, the density improvements of the two propagators are very close (79.2% vs 79.1%) and the mean deviation of the inferred relations is very small (0.047). Moreover, the average strength of the omitted edges of the alternative propagator (the ones inferred by the full propagator but ignored by the alternative propagator) is only 0.006. For the friendship network, though the improvements of the two propagators in network density vary (77% vs 39%), the mean deviation and omitted values are both trivial (0.006 and 0.001, respectively). These comparisons demonstrate that the approximation Equation (4.11) is a good alternative to Equation (4.10) for simplifying the calculation.

Table 4-2 Result comparison of the full and alternative propagators

User relations	Similarity	Friendship
Original graph density	22.4%	0.6%
Density after propagation (full)	79.2%	77%
Density after propagation (alternative)	79.1%	39%
Mean deviation of both-inferred edges	0.047	0.006
Mean value of omitted (unpredicted) edges	0.006	0.001

collected from Last.fm dataset

In summary, the proposed social network propagation will be applied to every collected single network of users as a pre-processing step to enrich the input data of our multi-relational social network-based recommender system. For convenience of expression, we still use the original notations like $\mathbf{W}$ instead of $\hat{\mathbf{W}}$ in the reminder of this chapter unless specified.

4.3 Multi-Relational Social Networks

Let us assume that multiple social networks have been initialized for users from a range of resources such as user behaviours, preference, social interactions or implicit feedback information. This section discusses how a multigraph model is generated and measured for the perpetration of recommendations.

4.3.1 Multigraph Generation

Formally, let graphs $G_1 = \{\mathcal{U}, E_1, w_1\}$, $G_2 = \{\mathcal{U}, E_2, w_2\}$, …, $G_Z = \{\mathcal{U}, E_Z, w_Z\}$ denote the collected and propagated Z different social networks on a common user set $\mathcal{U} = \{u_1, u_2, \ldots\}$, which is the population of users in a recommender system, and $E_1, E_2, \ldots, E_Z \subseteq \mathcal{U} \times \mathcal{U}$ are respectively the edge sets of each graph. Each graph G_k is associated with a weighting function w_k and correspondingly a weighting matrix $\mathbf{W}^{|\mathcal{U}| \times |\mathcal{U}|}$ as denoted in Section 4.2.

A simple way to handle different social networks is to aggregate different user relations to build a union graph, which can be defined as follows.

Definition 4-2 (Union Graph). A union graph $G' = \{V, E', w'\}$ is also an ordinary simple graph on the vertex set $\mathcal{U}$, in which the edges are given by the union set of all single edge sets, i.e., $E' = \bigcup_{k=1}^{Z} E_k$. Correspondingly, a union weighting function $w' : E' \rightarrow [0,1]$ is associated to aggregate the available edge weightings of single graphs:

$$w'(e) = \mathcal{F}\Big(w_1(e), w_2(e), \ldots, w_Z(e)\Big), \forall e \in E', \tag{4.13}$$

where $\mathcal{F}$ is an aggregating function such as linear averaging in the study of Jacob et al. (2011).

In contrast, a multigraph that retains the original structures of single graphs is defined as follows (Gjoka et al., 2011) .

Definition 4-3 (Multigraph). A multigraph $G = \{\mathcal{U}, E\}$ is a special graph on V, where the edge set $E = \biguplus_{k=1}^{Z} E_k$ is given by the multiset of the edge sets of all single graph G_1 to G_Z.

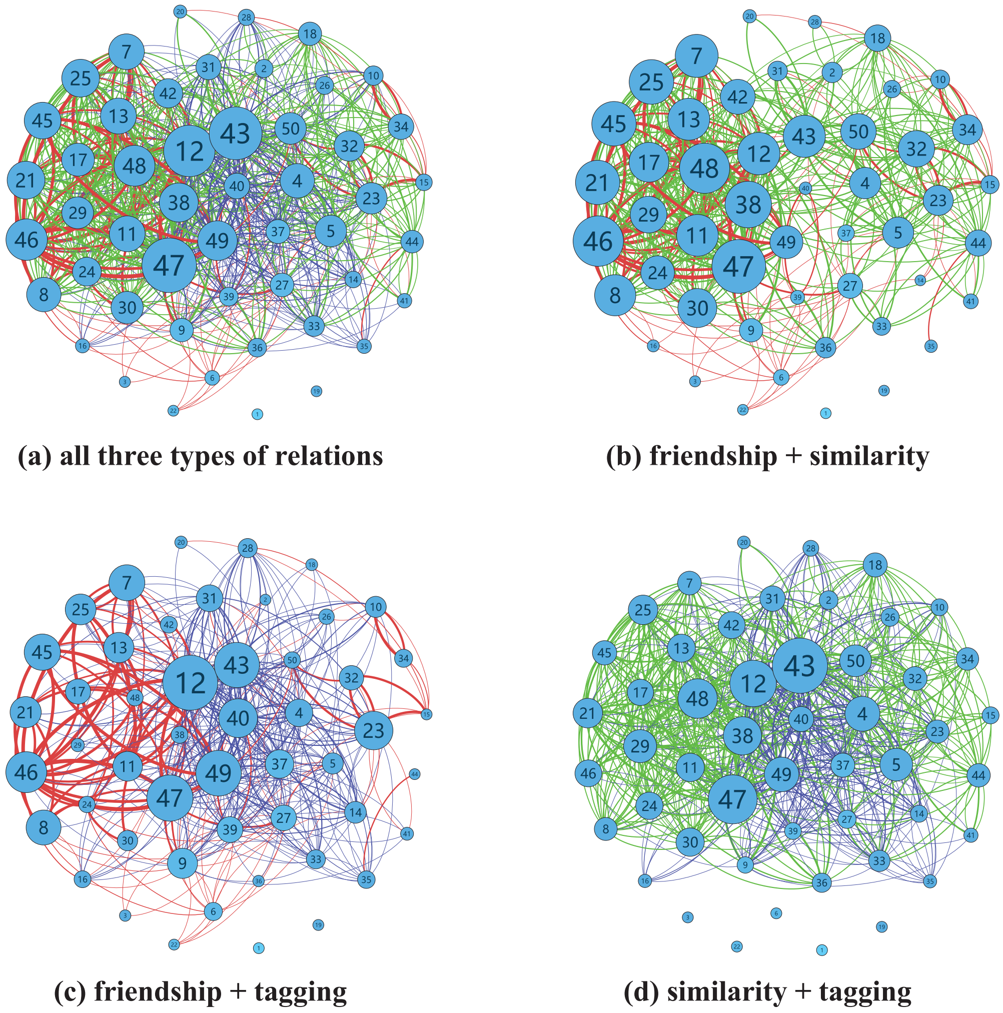
As shown in the above definition, a multigraph can be seen as to overlap all single graphs rather than to merge them to one simple graph. In the following, we use the subscript symbols i, j to index vertices (users) and k, l to index single graphs.

4.3.2 Inter-Network Comparison

Compared to the union graphs that simply merge different types of user relations, the multigraph model retains the original structures of single networks, which helps to identify the structural difference between single networks. In the field of network study such as network sampling, the inter-network comparisons have acquired much study (Marceau et al., 2011; Zhao et al., 2014). We introduce the **Average Similarity of Neighbours (ASN)** as a measurement of the structural similarity of two networks, defined as $\mathrm{ASN}(A,B)=\sum_i K_{AB}(i)/\sum_i K_A(i)+K_B(i)-K_{AB}(i)$, where $K_A(i)$ (respectively $K_B(i)$) is the number of neighbours of node v_i in single graph A (respectively B) and $K_{AB}(i)$ is the number of common neighbours of the i -th node in both graphs A and B. This metric was not originally proposed for weighted graphs, thus we modify it to the following form.

$$\mathrm{ASN}(A,B)=\frac{\sum_i \mathrm{DEG}_{AB}(i)}{\sum_i \mathrm{DEG}_A(i)+\mathrm{DEG}_B(i)}, \tag{4.14}$$

where $\mathrm{DEG}_A(i)$ and $\mathrm{DEG}_B(i)$ are respectively the out degrees of the i-th node in graph A and graph B and $\mathrm{DEG}_{AB}(i)$ denotes the summation of the out degrees of this node to the common neighbours in both graphs. Clearly, we have that $\mathrm{DEG}_{AB}(i)\leq \mathrm{DEG}_A(i)+\mathrm{DEG}_B(i)$ and the equation holds only if the node has the exactly same out-linked neighbours in both graphs.



(a) all three types of relations

(b) friendship + similarity

(c) friendship + tagging

(d) similarity + tagging

(red: friendship; blue: preference similarity; green: co-tagging relations)

Figure 4-2 Inter-network comparisons with Last.fm dataset

For recommender systems, we put emphasis on the structural "diversity" rather than the "similarity" measurement of different user networks, given as follows.

$$\delta(A,B) = 1 - \frac{\sum_i \mathrm{DEG}_{AB}(i)}{\sum_i \mathrm{DEG}_A(i) + \mathrm{DEG}_B(i)}. \tag{4.15}$$

The proposed inter-network diversity measurement can be used to pre-screen the various input networks. For example, we adopt only one of the two social networks if they have very small diversity (e.g. $\delta < 0.1$) for the following two reasons:

- First, the two networks have very similar structure such that the sparseness problem is not further alleviated if both are incorporated;
- Second, incorporating duplicate networks with too similar structure is equivalent to reusing a same information resource, which renders it unfair for other input resources.

Figure 4-2 visually illustrates the network-to-network comparisons. We import the complex relationships of the first 50 users in the Last.fm dataset mentioned in the last section. Three relational networks between users are initialized from different resources including the explicit friendships, the preference similarities, and the co-tagging relations. The overall multigraph structure is presented in Figure 4-2(a) by overlapping all three kinds of relations together. We compare each pair from the three single networks to measure their structural diversity in the subfigures (b), (c) and (d), respectively. Intuitively, Figure 4-2(b) shows that the friendship and the similarity networks share a small part of edges. Figure 4-2(c) presents the friendship network and tagging network together and illustrates that these two networks are also well distinguished. Figure 4-2(d) indicates the similarity and tagging networks have more common edges compared to the former two figures. We calculate the inter-network diversity measurement of Equation (4.15) for each pair of the three networks and obtain the following results.

$$\begin{cases} \delta(\text{friendship}, \text{similarity}) = 0.82 \\ \delta(\text{friendship}, \text{tagging}) = 0.83 \\ \delta(\text{similarity}, \text{tagging}) = 0.42 \end{cases} \tag{4.16}$$

The result supports our intuitive observations of that the friendship and similarity networks are different in structure as well as the friendship and the tagging networks, while the diversity between the similarity and the tagging networks is lower, in other words, these two networks are more similar in the structure.

It should be noticed that the proposed inter-network diversity measurement does only evaluate the difference in structural rather than the quantitative weights of edges. In other words, “high diversity” here indicates that two networks have complementary structures, i.e., they share a small part of the common edges, while it does not mean that their edge weights differ with each other. Admittedly, we would like to utilize more complementary input information to alleviate the sparseness problem of a single user network.

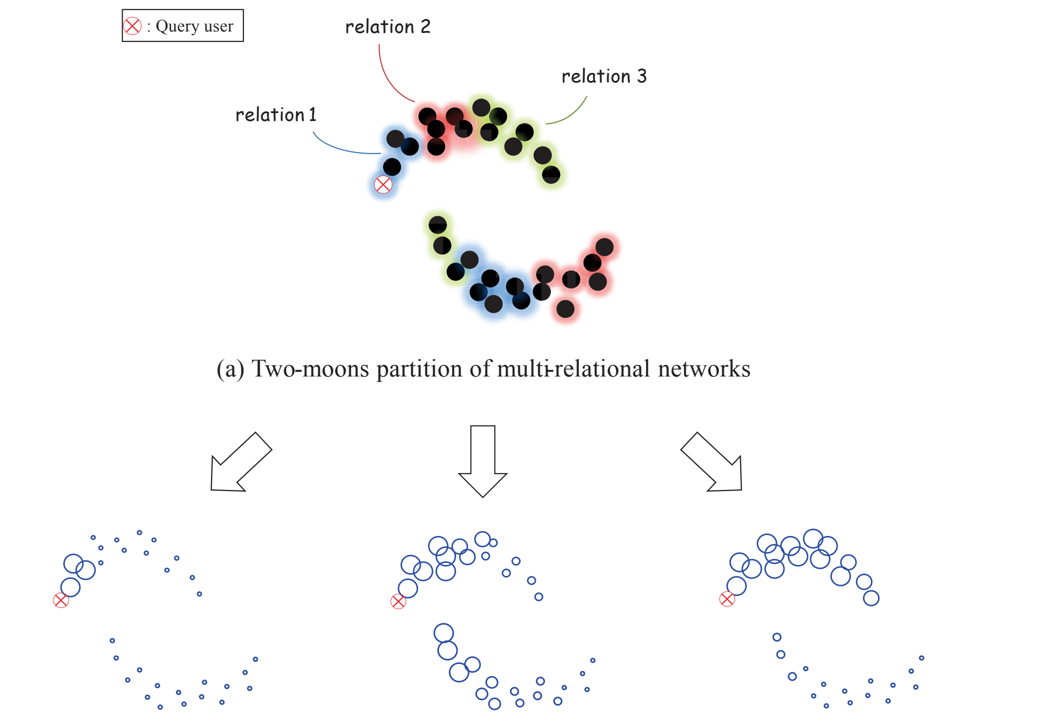
In this section, we present a multigraph model that can involve multi-relational social networks of users. A novel measurement of the inter-network structure diversity is proposed and illustrated using a real-world dataset. In the next section, we focus on how to identify the overall closeness of users from the multigraph structure to improve recommendation precision.

4.4 Multigraph Ranking and Recommendations

One of the key tasks of CF-based recommender systems is to identify the closed neighbour users who are thought to have similar preference. There are various of user relationships that have been imported as clues to evaluate the closeness between users such as the rating-based similarity/trust (O’Donovan and Smyth, 2005; Resnick et al., 1994), explicit social connections (Massa and Avesani, 2009) and implicit correlations (Lopes et al., 2010). Differing from the most existing studies that only apply to a single type of social relations, this section proposes a multigraph ranking model to be able to identify the nearest neighbours from multiple social networks.

To illustrate the need and the advantage of multigraps, we start this section with a two-moons ranking problem shown in Figure 4-3(a). In this example, the users are connected with three types of relations, namely, similarity, friendship and tagging correlations as in the previous example of the Last.fm dataset. User nodes are placed in a geometric figure with regarding to the average strength of the three types of relations to the query user. In addition, those users who are strongly connected with a particular type of relations are marked with a unique surrounding colour such as blue for similarity, red for friendship and green for tagging correlations, thereby we can find several small groups of users surrounded with different colours in the figure. Putting all these small groups together, we obtain a two-moons pattern of the whole structure indicating that two large partitions are generated by different adjacent small groups of users. Thus we call this problem as a two-moons multi-relational social network ranking problem. The goal of our study is to identify the overall nearest neighbours for the particular query user marked in the figure. In the following subfigures (b) to (d), three different ranking ideas are compared. Note that the marker sizes are proportional to the ranking scores of a particular ranker.

The conventional single network-based approaches will find the nearest neighbours who are connected to the query user with a particular type of relations. For example, Figure 4-3(b) presents the expected ranking result of CF approach using the preference similarity of users. Due to data sparsity, CF method can only find a small number of neighbours, which is thereby not a successful ranker as many users are failed to assess. Figure 4-3(c) is the ranking result of a union graph-based ranker that use the averaged Euclidian distance. As this ranker compares the average closeness of the candidate users to the query user independently that the connections between the candidates are ignored. As a result, the two-moons pattern is not able to be recognized by this kind of ranker. Figure 4-3(d) shows the ideal ranking result that we expect to obtain by proposing a multigraph ranking model. This ranker can identify that the users in a same partition constituted by adjacent small groups with different types of relations.



(a) Two-moons partition of multi-relational networks

(b) Ranking by CF similarity (c) Union graph-based ranking (d) Multigraph-based ideal ranking

Figure 4-3 Ranking on a multi-relational network with a two-moons pattern

In the following, a brief preliminary study of simple graph ranking is needed to grasp the key idea of ranking graph data, which we then expand to the multigraph environment.

4.4.1 Simple Graph Ranking

The graph ranking problem has been studied in considerable depth in recent years that some solutions have been readily adopted in the area of network sampling (Belkin et al., 2004; Zhou and Schölkopf, 2004; Agarwal, 2006; Mao et al., 2014). This problem is given by a weighted graph $G=(V,E,w)$, where $V=\{v_1,\ldots,v_n\}$ is a set of vertices, $E\subseteq V\times V$ a set of edges, and $w:E\rightarrow[0,1]$ the weighting function, together with an input *query vector* $\mathbf{y}\in\mathbb{R}^n$, in which the i-th element y_i denotes the initial query score of the node v_i. The query vector can be seen as a given (input) ranking function $y:V\rightarrow\mathbb{R}$ on the vertex space such that $y(v_i)=y_i$.

The ranking problem can then be thought of as seeking a new function $f : V \to \mathbb{R}$ that is smooth and simultaneously close to the given function y. The graph ranking problem is usually formalized to an optimization problem to minimize the following cost function.

$$\min_{f:V\to\mathbb{R}} Q(f) = \left\{ \mathcal{S}(f) + \mu\hat{R}(f;y) \right\} \tag{4.17}$$

In the above cost function, the first term $\mathcal{S}(f)$ in the right-hand side measures the smoothness of the ranking function f; the second term $\hat{R}(f;y)$ measures the empirical error of f compared with y. A trade-off parameter $\mu > 0$ is imported to balance the two terms.

The ranking error $\hat{R}(f;y)$ is usually computed using the ℓ_2 norm variance $\|\cdot\|$ and written in matrix-vector form as follows.

$$\hat{R}(f;y) = \|f - y\|^2 = (\mathbf{f} - \mathbf{y})^T(\mathbf{f} - \mathbf{y}) \tag{4.18}$$

A good ranking function f shall not vary greatly across two vertices that are "closely related". By importing the weighting matrix $\mathbf{W}$ and degree matrix $\mathbf{D}$ of graph G, the regularization framework first proposed in (Zhou and Schölkopf, 2004) gives the smoothness function as:

$$\mathcal{S}(f) = \mathbf{f}^T(\mathbf{I} - \mathbf{A})\mathbf{f}, \tag{4.19}$$

where a new matrix $\mathbf{A} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ is defined. Consequently, requiring the gradient of $Q(\mathbf{f})$ to vanish gives us the following result.

$$\left.\frac{\partial Q}{\partial \mathbf{f}}\right|_{\mathbf{f}=\mathbf{f}^*} = (\mathbf{I} - \mathbf{A})\mathbf{f}^* + \mu(\mathbf{f}^* - \mathbf{y}) = 0 \tag{4.20}$$

In defining a decimal parameter $\alpha = 1/(\mu+1) \in (0,1)$, the optimized ranking result $\mathbf{f}^*$ is obtained by solving Equation (4.20).

$$\begin{aligned} \mathbf{f}^* &= (1-\alpha)(\mathbf{I}-\alpha\mathbf{A})^{-1}\mathbf{y} \\ &\propto (\mathbf{I}-\alpha\mathbf{A})^{-1}\mathbf{y} \end{aligned} \tag{4.21}$$

The positive constant $1-\alpha$ can be omitted in the above equation as it does not affect the ranking order.

4.4.2 Multigraph Ranking Problem

We have the same goal for multigraph ranking as for the single graph ranking problem: to seek for a good ranking function $f: V \to \mathbb{R}$ that is smooth and simultaneously close to a given query $\mathbf{y}$. The problem can also be formalized to minimize a cost function in the form of Equation (4.21). It is clear that the second term $\hat{R}(f;y)$ is not changed in the multigraph environment, so we only have to elaborate a new smoothness function $\mathcal{S}(f)$ for the multigraph ranking problem.

In a multigraph constructed from several single graphs, the overall relationship of a pair of vertices is represented by a set of edges with regarding to different types of relations. We still use the notations defined in Section 4.3, which supposes that a multigraph G has been constructed for users with Z single social networks. The relationship between two users can be represented by a column vector containing the strength of closeness (edge weight) in each single network, as follows.

$$[u_i, u_j] \leftarrow \begin{pmatrix} w_{ij}^1 \\ w_{ij}^2 \\ \vdots \\ w_{ij}^Z \end{pmatrix} \tag{4.22}$$

Here we use $w_{ij}^k \in [0,1]$ as a simplified expression of $w_k([u_i,u_j])$, i.e., the closeness of the two users in the k-th graph. Note that $w_{ij}^{k}=0$ means there is no connection from u_i to u_j in the k-th single graph.

An *edge function* is defined as a function that can map a directed pair of vertices to a real value (Jacob et al., 2011). In a multigraph, a directed pair of vertices can be seen to have a virtual edge and the edge function gives an initial estimation for the strength of this virtual edge. To utilize both intra-network relations and inter-network comparisons, we define a virtual edge function $\varpi : V \times V \rightarrow \mathbb{R}$ for the user vertices in a multigraph as follows:

$$\varpi([v_i, v_j]; \Delta) = \frac{1}{2}\sum_{k,l}^{Z} w_{ij}^{k} w_{ij}^{l} \Delta_{kl}, \tag{4.23}$$

where Δ_{kl} is the inter-network diversity of the k-th and the l-th single graphs, referring to the definition in Section 4.3.2. Comparing the structural diversity of each pair of single graphs, we obtain a $Z \times Z$ diversity matrix Δ as follows.

$$\Delta_{kl} = \begin{cases} \delta(G_k, G_l), & k \neq l \\ 1, & k = l \end{cases} \tag{4.24}$$

This setting considers the unique environment for recommender systems, where we assume that the relationship of two particular users should be emphasized if they are connected by many and diverse types of relations. With Equation (4.23), we define a $|\mathcal{U}| \times |\mathcal{U}|$ matrix $\mathbf{\Pi}$ with $\Pi_{ij} = \varpi([v_i, v_j])$, which can be built based on the matrices $\mathbf{W}_1, \ldots, \mathbf{W}_Z$ and Δ, as shown below.

$$\mathbf{\Pi} = \frac{1}{2}\sum_{k,l=1}^{Z} \Delta_{kl} \mathbf{W}_k \circ \mathbf{W}_l \tag{4.25}$$

Here $\circ$ is entry-wise multiplication operator. Further, we can define the virtual out-degree $\mathfrak{d}^+(u_i)$ and in-degree $\mathfrak{d}^-(u_i)$ of a user vertex u_i as follows.

$$\mathfrak{d}^+(u_i) = \sum_{j \neq i} \varpi([u_i, u_j]), \tag{4.26}$$

$$\mathfrak{d}^-(u_i) = \sum_{j \neq i} \varpi([u_j, u_i]). \tag{4.27}$$

Two diagonal matrices $\mathfrak{D}_+$ and $\mathfrak{D}_-$ are denoted with $(\mathfrak{D}_+)_{ii} = \mathfrak{d}^+(u_i)$ and $(\mathfrak{D}_-)_{ii} = \mathfrak{d}^-(u_i)$ on the diagonals.

According to the regularization framework, the smoothness of a ranking function consists of the *edge derivation* crossing every pair of users, which we define as follows for the virtual edge in a multigraph.

For an out-linked virtual edge:

$$\left.\frac{\partial f}{\partial[u_i, u_j]}\right|_{u_i} = \sqrt{\frac{\varpi([u_i, u_j])}{\mathfrak{d}^+(u_i)}} f(u_i) - \sqrt{\frac{\varpi([u_i, u_j])}{\mathfrak{d}^-(u_j)}} f(u_j) \tag{4.28}$$

For an in-linked virtual edge:

$$\left.\frac{\partial f}{\partial[u_j, u_i]}\right|_{u_i} = \sqrt{\frac{\varpi([u_j, u_i])}{\mathfrak{d}^-(u_i)}} f(u_i) - \sqrt{\frac{\varpi([u_j, u_i])}{\mathfrak{d}^+(u_j)}} f(u_j) \tag{4.29}$$

Clearly, we have $\left.\frac{\partial f}{\partial[u_i,u_j]}\right|_{u_j} = -\left.\frac{\partial f}{\partial[u_i,u_j]}\right|_{u_i}$. Next, the *local variation* at each vertex is defined as below.

$$\|\nabla_{u_i} f\| = \sqrt{\frac{1}{2}\left[\sum_{j \neq i}\left(\left.\frac{\partial f}{\partial[u_i, u_j]}\right|_{u_i}\right)^2 + \sum_{j \neq i}\left(\left.\frac{\partial f}{\partial[u_j, u_i]}\right|_{u_i}\right)^2\right]} \tag{4.30}$$

The summation of the local variations at all users reflects the smoothness level of the ranking function f. Following the study of Zhou and Schölkopf (2004), the smoothness function is given by:

$$\mathcal{S}(f) = \frac{1}{2}\sum_{u \in \mathcal{U}} \|\nabla_u f\|^2. \tag{4.31}$$

The above equation can be simplified in the following steps.

$$\begin{aligned}
&\mathcal{S}(f) \\
&= \frac{1}{2}\sum_{i=1}^{n} \|\nabla_{u_i} f\|^2 \\
&= \frac{1}{4}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\partial f}{\partial[u_i,u_j]}\bigg|_{u_i}\right)^2 + \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\partial f}{\partial[u_j,u_i]}\bigg|_{u_i}\right)^2\right] \\
&= \frac{1}{4}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\partial f}{\partial[u_i,u_j]}\bigg|_{u_i}\right)^2 + \sum_{j=1}^{n}\sum_{i=1}^{n}\left(-\frac{\partial f}{\partial[u_j,u_i]}\bigg|_{u_j}\right)^2\right] \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\partial f}{\partial[u_i,u_j]}\bigg|_{u_i}\right)^2 \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\varpi_{ij}}{\mathfrak{d}_i^+} f_i^2 + \frac{\varpi_{ij}}{\mathfrak{d}_j^-} f_j^2 - 2\frac{\varpi_{ij} f_i f_j}{\sqrt{\mathfrak{d}_i^+ \mathfrak{d}_j^-}}\right) \\
&= \frac{1}{2}\sum_{i=1} f_i^2 + \frac{1}{2}\sum_{j=1} f_j^2 - \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\varpi_{ij} f_i f_j}{\sqrt{\mathfrak{d}_i^+ \mathfrak{d}_j^-}} \\
&= \mathbf{f}^T\mathbf{f} - \mathbf{f}^T \mathfrak{D}_+^{-1/2} \boldsymbol{\Pi} \mathfrak{D}_-^{-1/2} \mathbf{f} \\
&= \mathbf{f}^T(\mathbf{I} - \mathbf{S})\mathbf{f}
\end{aligned} \tag{4.32}$$

In the above calculation, a matrix $\mathbf{S}$ is defined as follows.

$$\mathbf{S} = \mathfrak{D}_+^{-1/2} \boldsymbol{\Pi} \mathfrak{D}_-^{-1/2} \tag{4.33}$$

The multigraph ranking problem can thus be formalized to the following optimization.

$$\min_{f:V\to\mathbb{R}} Q(f) = \mathbf{f}^T(\mathbf{I}-\mathbf{S})\mathbf{f} + \mu(\mathbf{f}-\mathbf{y})^T(\mathbf{f}-\mathbf{y}) \tag{4.34}$$

The new cost function has a similar form with the one of single graph ranking, so we can obtain the optimized ranking vector $\mathbf{f_m}$ for multigraph ranking following the same calculations of Equation (4.21), as follows.

$$\mathbf{f_m} = (\mathbf{I}-\alpha\mathbf{S})^{-1}\mathbf{y} \tag{4.35}$$

The same, $\alpha \in (0,1)$ is the model parameter balancing the ranking smoothness and the consistency with input query.

4.4.3 Making Recommendations

To generate recommendations for a particular (active) user denoted as u_a, we hope to identify the top-K most closed neighbour users using the proposed multigraph ranking model. At beginning, a column vector $\mathbf{y} \in \mathbb{R}^{|\mathcal{U}|}$ is given as the input query for multigraph ranking. The edge function defined in Equation (4.23) can be natively imported to generate an initial query vector, such as follows.

$$y(u_i) = \begin{cases} Z^2/2 & \text{if } u_i = u_a \\ \varpi([u_a, u_i]) & \text{if } u_i \neq u_a \end{cases} \tag{4.36}$$

It is easy to find that $\varpi \leq Z^2/2$ from Equation (4.23).

With the query vector $\mathbf{y}$, we can find the optimized ranking vector $\mathbf{f}$ by solving Equation (4.35). The top-K users who acquires the highest ranking scores are then selected as the nearest neighbours for the active user, denoted as $\mathrm{Neib}(u_a)$. Next, the standard CF prediction formula is used to predict the rating of the active user u_a to an unseen item i (Resnick et al., 1994).

$$\hat{r}(u_a, i) = \bar{r}_{u_a} + \frac{\sum\limits_{u \in \mathrm{Neib}(u_a)} f(u)(r_{u,i} - \bar{r}_u)}{\sum\limits_{u \in \mathrm{Neib}(u_a)} f(u)} \tag{4.37}$$

Ultimately, several (e.g. 5 or 10) of the top unseen items with the highest predicted ratings are recommended to the active user u_a.

Now we can summarize the whole process of the proposed recommendation approach. At first, we collect various explicit or implicit social relations for users in a recommender system and build multiple single social networks. Next, the proposed random walk-based social network propagation model is employed to enrich the original data of each social network. After the propagation of all social networks, a multigraph is constructed to represent the multi-relational environment for users. Now given a particular active user as the requester for recommendations, the proposed multigraph ranking model is implemented to identify this user's closed neighbour users and the CF-like rating predictions are made for the unseen items. Ultimately, the items with highest predictions are selected as a list of recommendations to present to the active user.

4.5 Experiments

In this section, empirical experiments are conducted with two real-world datasets to compare the performance of our approach with several existing social network-based recommender systems.

4.5.1 Experiment Setup

The two public datasets selected for experiments are from Epinions.com and Last.fm, which are the few publicly available datasets that contain both ratings and social networks thereby having been widely used for evaluating social network-based recommender systems (Massa and Avesani, 2007; Jamali and Ester, 2009; Yang et al., 2012; Bellogín et al., 2013). Epinions.com is a general product review site on which customers can rate and review different domains of products including cars, books, movies, software, etc. The ratings range from 1 (the worst) to 5 (the best). In addition, there is a "web of trust" network recording users' connections of "who trusts whose opinions" as the system enable every user to build a trust list and add people who share similar opinions into it. We use this version[1] initially crawled by Massa and Avesani (2007) for our experiments. The statistical information is presented in Table 4-3, which indicates the rating data are very sparse (sparsity 99.99%). As a result of the high rating sparsity, it has been reported that the pure CF approaches suffer from the cold-start problems heavily in this environment (Massa and Avesani, 2007), which is actually common in e-Commerce sites such as Amazon.com (McAuley and Leskovec, 2013) and Yelp.com (Blomo et al., 2013). The need for new information like trust to be incorporated to enhance recommendation performance is therefore highlighted.

The other selected dataset is a music sharing dataset[2] of Last.fm, published by Cantador et al. (2011). There is no explicit rating information in this dataset but the "listening count" logs of users are often seen as implicit "ratings" to the music tracks (Tan et al., 2011). In addition, an explicit online friendship network of users is provided in the dataset. In Last.fm, users are also able to assign free tags to describe the music tracks or artists. It has been reported that a tag-derived social network of users can be abstracted for recommendations (Zhen et al., 2009; Liang et al., 2010). We follow the settings of Zhen et al. (2009) to initialize a co-tagging network of users as another social network besides the friendship network. Some statistics of the Last.fm dataset is summarized in Table 4-4. This table indicates that the Last.fm dataset has much denser rating data, which represents a common scenario of information goods. We transform the listening counts to ratings in the following way.

[1] http://www.trustlet.org/wiki/Downloaded_Epinions_dataset

[2] http://grouplens.org/datasets/hetrec-2011/

For every user, the top one fifth of the most listened artists are seen to be rated with 5 stars (most loved), the second fifth are given the rating 4, and so on. Consequently, the ratings generated for the Last.fm dataset range from 1 to 5 as well as in the Epinions dataset.

Table 4-3 Statistical information of the Epinions dataset

	Total	per-user	Sparsity
#Users	49290	-	-
#Products	139738	-	-
#Ratings	664824	13.5	99.99%
#Trust	487181	9.9	99.98%

Table 4-4 Statistical information of the Last.fm dataset

	Total	Per-user	Sparsity
#Users	2100	-	-
#Artists (items)	18745	-	-
#Listening count (rating)	92834	44.2	99.76%
#Friendship	25424	12.1	99.42%
#Tags	186479	88.8	99.53%

Table 4-5 shows the two types of user relations collected in the Epinions dataset and the three types of user relations collected in the Last.fm dataset.

Table 4-5 Involved user-user relations in two datasets

User relations	Epinions	Last.fm
Rating Similarity	√	√
Social Trust	√	
Social Friendship		√
Tagging correlation		√
No. of networks (Z)	2	3

In the experiment, our proposed multigraph ranking-based recommendation approach is named as **MGrank** for short, and we compare it with the following single or hybrid approaches. At first, each original single social network is used to implement a user-based CF recommendation approach for testing the recommendation performance relying on a single type of information resources. We also implement an enriched version of each single social network using the propagation method proposed in Section 4.2. Comparing the original and propagated social networks-based approaches can evaluate whether the proposed random walk-based propagation model is effective for improving the recommendation performance. For the two datasets, the following single approaches are implemented and labelled.

- **sCF** and **sCFPro**: the single approaches using the original and propagated Cosine similarity networks for collaborative filtering (for both datasets).
- **sTrust** and **sTrustPro**: the single approaches using the original and propagated trust network (for the Epinions dataset only).
- **sFriend** and **sFfriendPro**: the single approaches using the original and propagated friendship network (for the Last.fm dataset only).
- **sTag** and **sTagPro**: the single approach using the original and propagated tag-derived user social network (for the Last.fm dataset only).

According to our taxonomy for the three fusion strategies reviewed in Section 2.2.2, we select a representative hybrid approach from each category as a benchmark to compare with our multigraph-based approach.

- **Post hoc combination approach**: The post hoc combination model of Shambour et al. (2012) is selected, in which the harmonic mean is used to aggregate the prediction result of each single approach. This approach is denoted as **PostCB** for short.
- **Unified model-based approach**: The TrustWalker model proposed by Jamali and Ester (2009) is selected. Note that the TrustWalker model can only incorporate a single social network; the tagging information in the Last.fm dataset is not included. The name is further simplified as **TrustWK** for short.
- **Neighbourhood integration-based approach**: The union graph referring to Definition 4-2 can be seen as a neighbourhood integration model. Here, the arithmetic average of different relations is used to set the weighting of the union edges. This approach is labelled as **Union** for short.

The experiments aim to test the recommendation performance of each approach in terms of recommendation success rate and rating prediction accuracy, thus the recommendation coverage and RMSE are selected as the two evaluation metrics, referring to the definitions in Section 2.4.

4.5.2 Performance Comparison

Five-fold validations are conducted on both datasets, i.e., each dataset is split in five partitions and at each round one of them is selected as the test set and the rest four partitions are used as training set. At first, we collect the the recommendation coverage rates of all approaches in Figure 4-4 (Epinions dataset) and Figure 4-5 (Last.fm dataset). Both figures demonstrate that our approach achieves the highest coverage score compared to all approaches.

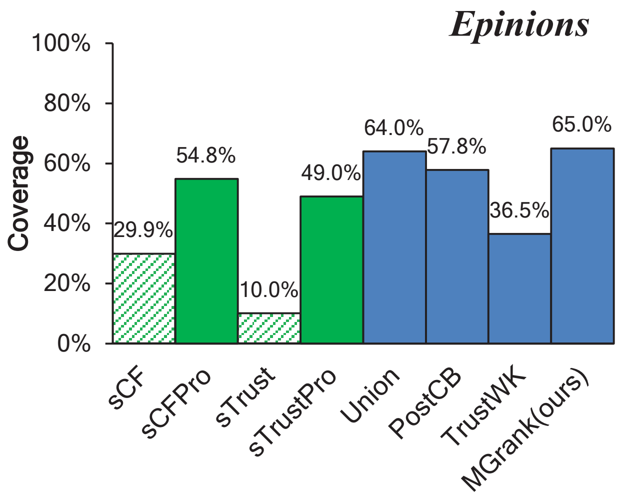


Figure 4-4 Comparison of recommendation coverage (Epinions dataset)

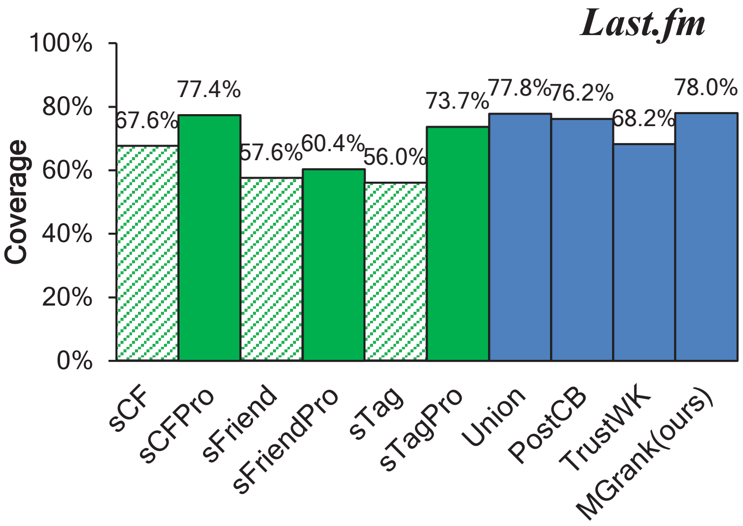


Figure 4-5 Comparison of recommendation coverage (Last.fm dataset)

Focusing on the comparison of the recommendation coverages of single approaches, we can find the significant improvements after the use of social network propagation, especially when the original social network is very sparse. This finding indicates the success of our proposed random walk-based social network propagation model in alleviating the data sparseness problem. In the particularly sparse environment of the Epinions dataset, the original single approaches are found suffering from the sparseness problem heavily. For example, the sCF completes only 30% of predictions while sTrust performs even worse (10%). After the network propagation, the coverage rates of the enhanced single approaches are increased to 55% (sCFPro) and 49% (sTrustPro), respectively. Similar improvements are also observed in the experiments on the Last.fm dataset though the data are relatively denser, e.g., the coverage of sCF is increased from 68% to 76% after propagation.

Figure 4-4 and Figure 4-5 also shows that the hybrid approaches incorporating more than one social network acquire higher coverage rates than single ones. It demonstrates that the incorporation of different types of social relations indeed helps to alleviate the sparseness problem suffered by the single resource-based approaches. In the Epinions dataset, the highest recommendation coverage scores are achieved by the MGrank (65%) and Union (64%) approaches, followed by the PostCB (58%). In contrast, the TrustWK completes only 37% of predictions in such a sparse environment of the Epinions dataset. It is worth mentioning that MGrank, Union and PostCB all have social network propagation processes to enrich the original social networks (PostCB uses its own trust propagation model), while the model-based TrustWK uses the original non-propagated networks directly. As a result, TrustWK generally has lower coverage than other three hybrid approaches, but it still significantly outperforms single approaches such as sCF and sTrust. In the Last.fm dataset with more and denser social networks, all of the four hybrid approaches acquire high coverage rates ranging from 76% to 78%, among which the MGrank approach still maintains the best performance.

Like other social network-based recommender systems, our approach aims to mostly improve the recommendation coverage without sacrificing the accuracy. Hence it is more important to compare the prediction errors of all approaches. The RMSE is selected as the evaluation metric and experiments are conducted on different datasets. Figure 4-6 and Figure 4-7 present the RMSE results for Epinions dataset and Last.fm datasets respectively, along with the different settings of the selected neighbourhood size for KNN-based approaches, such as the single network-based approaches, PostCB, Union and MGrank. Note that the TrustWK approach is not a KNN approach so its RMSE scores do not vary with K.

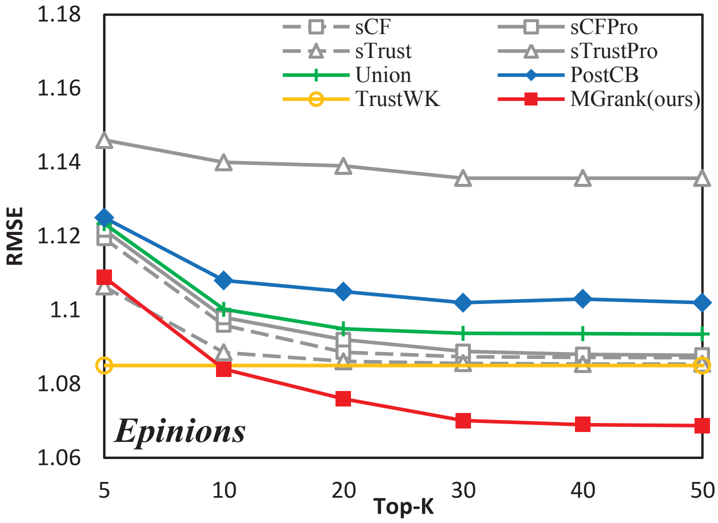


Figure 4-6 Comparison of RMSE on the Epinions dataset

For the Epinions dataset, Figure 4-6 shows that the sCFPro and sTrustPro approaches are of even higher error rates than sCF and sTrust which do not apply social network propagations. Furthermore, the hybrid approaches Union, PostCB and TrustWK that incorporate different input resources are also hard to overcome the single resource-based approaches sCF and sTrust in terms of prediction errors. This finding agrees with the conclusions of previous studies such as (Massa and Avesani, 2007) that the traditional social network-based recommender systems are hard to maintain the recommendation accuracy though the recommendation coverage is commonly improved as multiple resources are integrated. However, our approach MGrank is able to outperform the single approaches sCF and sTrust and achieve the lowest errors among all compared models under the most settings of K. Recalling that MGrank also performs the highest recommendation coverage, the experiments on Epinions dataset demonstrate that the proposed multigraph ranking model is effective for improving the recommendation performance in terms of both accuracy and success rate in the sparse environments.

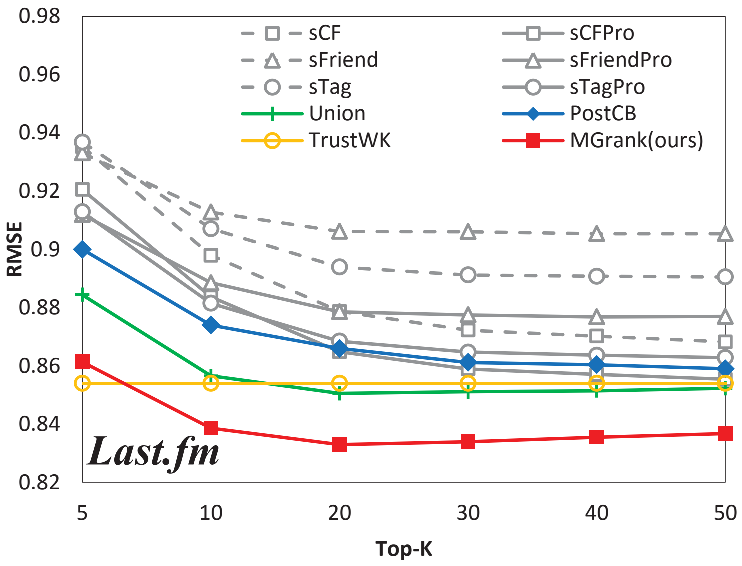


Figure 4-7 Comparison of RMSE on the Last.fm dataset

In the Last.fm dataset with relatively richer data, we find the social network propagation is effective for single approaches in reducing prediction errors, referring to the comparison of sCF and sCFPro in Figure 4-7 for example. In addition, the most hybrid approaches carry out better results than the single approaches, which indicates that the incorporation of different resources is successful in reduce prediction errors. In this dataset, MGrank still achieves significant improvement compared to all other approaches.

It is also noteworthy to analyse the performances of the three selected benchmark hybrid approaches that represent different information fusion strategies of conventional social network-based recommender systems. The post hoc combination approach (PostCB) is found able to increase recommendation coverage but cannot guarantee the precision by just simply averaging the outputs of different single resource-based approaches. The union graph-based approach (Union) has the similar problem of the post hoc combination, in that it is hard to further increase prediction accuracy if the original data are too sparse. Both of these two approaches can be essentially seen as simple average methods. While the model-based approach (TrustWalker) generally outperforms the former two approaches in terms of accuracy but the improvement in recommendation coverage is limited, especially in the sparse environments.

4.5.3 Parameter Adjusting

In Equation (4.35), a trade-off parameter $\alpha = 1/(1+\mu) \in (0,1)$ needs to be tuned for the proposed multigraph ranking model. We elaborate the variation of RMSE of MGrank with increasing values of α on the both datasets. Figure 4-8 shows a gradual decreasing trend of RMSE for the Epinions dataset until it maintains the lowest level since $\alpha = 0.99$, equally $\mu = 0.01$. Referring to the cost function Equation (4.34), this result indicates that the structural cost (the first part of the equation) should be considered mostly, while the experience cost (the second part) should be weighted less. This finding demonstrates that when a recommender system has no sufficient input data to generate precise relations between users, the initial query vector of the proposed multigraph ranking problem is not accurate enough to be a reference for the final ranking result.

For the Last.fm dataset, on the other hand, RMSE reaches the best performance (lowest level) at $\alpha = 0.8$, followed by a shape increase when α goes higher, as shown in Figure 4-9. The optimization is thus obtained at $\mu = 0.25$, which indicates the best balance of the cost functions for Last.fm dataset. This finding demonstrates that when a recommender system has rich input data to generate precise relations between users, the initial query vector is a good reference for the final ranking result such that the balancing parameter should not be too high or to low.

The above discussions give a guideline for choosing an appropriate trade-off parameter for different environment. As a result of the high level of data sparsity, such as in Epinions, a less accurate query vector $\mathbf{y}$ is initialized so that less consideration is given to the experience cost $\hat{R}(f;y)$, equivalently, α should be tuned to high in this circumstance. On the other side, if the data is dense such as in

Last.fm, the initial query vector $\mathbf{y}$ becomes informative and accurate; the ranking result $\mathbf{f}$ is hence required to maintain consistency with $\mathbf{y}$. Therefore, parameter α should be tuned lower if the input data are sufficient.

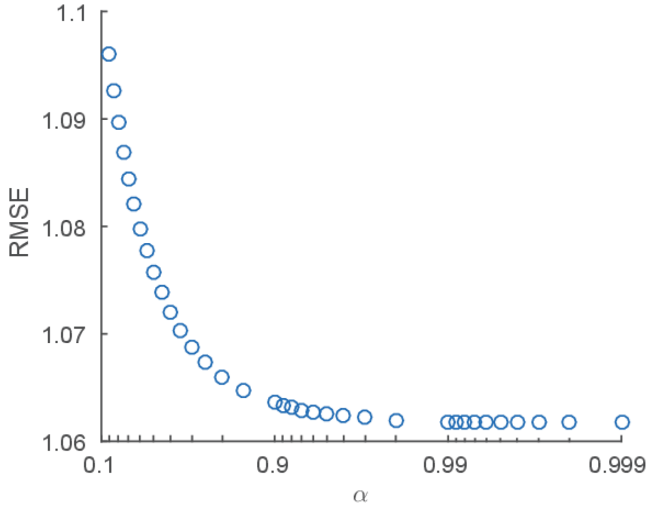


Figure 4-8 Performance variation with different parameter settings (Epinions dataset)

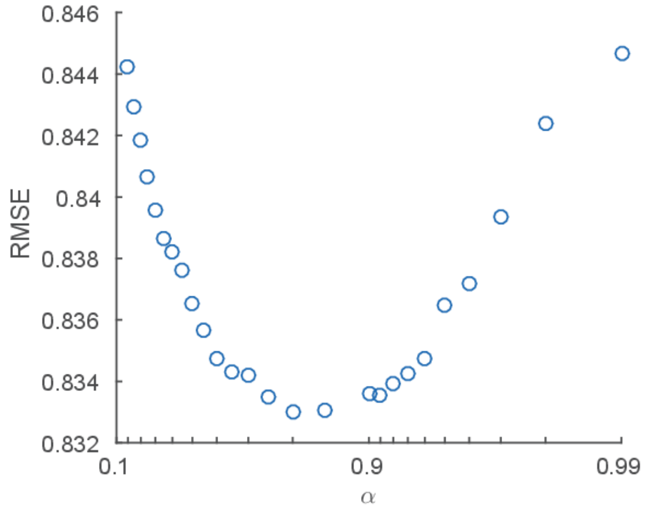


Figure 4-9 Performance variation with different parameter settings (Last.fm dataset)

4.6 Summary

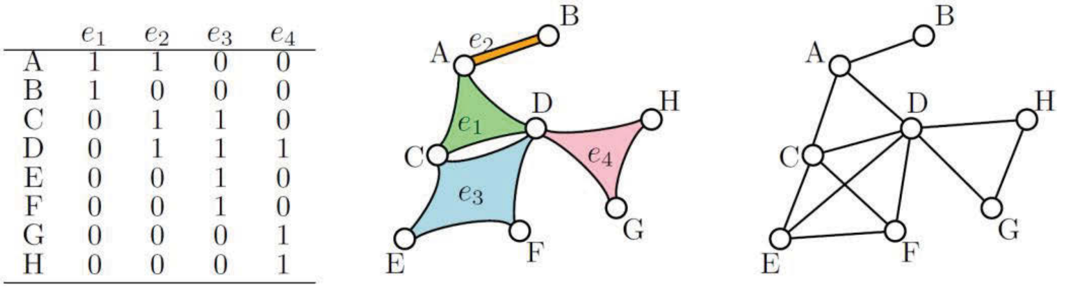
Collaborative filtering has been successfully employed in recommender systems to find potentially preferred items for users. As a result of relying only on rating data, however, it suffers from the cold start problem when ratings are too sparse. To overcome this drawback, there are increasing social network-based recommender systems that import the social connections or correlations between users as an alternative or additional resource for recommendations. The study of this chapter differs from previous works as it can handle the situation when users are connected by multiple social networks simultaneously, which actually becomes more and more common nowadays with the development of online social networking techniques. This chapter proposes a multigraph model to represent users' multi-relational social networks and develops a hybrid recommendation approach based on multigraph ranking. First, a multigraph model is proposed retaining the original structural information of users' multi-relational social networks. An inter-network diversity measurement is also proposed for evaluating the structural complementarity of different social networks. We then address a multigraph ranking problem in which both the intra-network relations and inter-network diversities are considered. Solving this problem is able to precisely identify users' overall closeness for recommendation making from the diverse explicit and implicit correlations between them. The implementation of the proposed multigraph ranking model is believed to be able to enhance recommendation performance in terms of both success rate and accuracy, in accordance with the results of a series of comprehensive experiments on two real-world datasets of Epinions and Last.fm.

It is also noteworthy that this chapter proposes a random walk-based social network propagation model as an effective data pre-processing step to enrich the original sparse social relations. A unique "walk and select" manner is introduced to perform random walks on a social network to infer indirect relations of users. This model provides a new attempt to solve the social network propagation problem and carries out a clearer result (the propagation formula) compared to other approaches that apply depth-first or breadth-first searching to traverse the possible connections between each pair of users. Empirical experiments also indicate that the proposed propagation model is helpful to enrich network density and increase recommendation coverage.

CHAPTER 5. A UNIFIED RECOMMENDER SYSTEM VIA MULTIPARTITE HYPERGRAPH RANKING

5.1 Overview

Recommendation techniques have been greatly explored in very recent decades and vary in the utilized information resources that different ideas have carried out, such as ranting-based CF approaches, item content-based approaches and user social network-based approaches as mentioned in previous chapters. However, there is still a lack of generic and unified recommendation models that can handle all possible information entities and their complex relations that may appear in real-world recommender systems. Despite item content and user social relations as discussed above, there are also extra information entities such as the textual comments and third-party environmental context information in recent Web2.0 applications. Thus, how to incorporate all these valuable information resources properly is challenging us to build a unified recommender system, which is the aim of the study in this chapter. In particular, there are some high-order relations among the different parties of information entities, which cannot be well represented by the ordinary pairwise relations. For example, the textual reviewing behaviour when a user selects some comments (words) to associate an item can be seen as a special connection between the three types of entities: a user, an item and some words. Take another instance, if a user's choice of item is influenced by environment context such as time, then the user-item preference like "this user likes to listen to classic music in the morning" becomes a high-order relationship between three entities $\langle user, context, item \rangle$ instead of a pairwise relationship $\langle user, item \rangle$ as usual. Naturally, these high-order relationships between different information entities construct a hypergraph model (Zhou et al., 2006) where a *hyperedge* can connect an arbitrary number of vertices rather than only two vertices as in simple graphs. Therefore, the hypergraph model is employed in this chapter to handle the various information entities and their complex relationships.



	e_1	e_2	e_3	e_4
A	1	1	0	0
B	1	0	0	0
C	0	1	1	0
D	0	1	1	1
E	0	0	1	0
F	0	0	1	0
G	0	0	0	1
H	0	0	0	1

(a) Relation table **(b) Hypergraph model** **(c) Ordinary graph model**

Figure 5-1 Modelling high-order relations using hypergraph vs. ordinary graph

The advantage of hypergraphs vs. ordinary graphs in dealing with complex relationships is illustrated in Figure 5-1 where the subfigure (a) is a collected data table showing four high-order relations e_1 to e_4 among eight entities A to G. Figure 5-1 (b) is the corresponding hypergraph model consisting of eight vertices and four hyperedges that exactly represents the high-order relations in the data table. In contrast, Figure 5-1 (c) presents an ordinary graph model. Clearly, it is hard to restore the original relations in the data table from the ordinary graph, that is, the ordinary graph loses the structural information of the high-order relations. For instance, the ordinary graph cannot identify that nodes C and D are connected by two different relations, e_1 and e_3.

Because different types of information entities are considered, this chapter will propose a multipartite hypergraph ranking model for recommendations, following the similar idea of the bipartite (simple) graph ranking model proposed in Chapter 3. The rest is followed by a preliminary introduction of hypergraphs and hypergraph ranking problem in Section 5.2. Next, Section 5.3 exploits the possible information entities and relations in recommender systems and presents a generic User-Item-Attribute-Context data model. With the generic data model, a multipartite hypergraph is constructed to handle different information entities and their high-order relations. To generate recommendations, Section 5.4 develops a balanced hypergraph ranking model which is suitable for the multipartite hypergraphs where the hyperedges vary greatly in the number of connected vertices. In Section 5.5, we conduct empirical experiments in a real-world dataset with various input resources from Yelp.com. Result analyses and conclusions are given at the last section.

5.2 Hypergraph and Hypergraph Ranking

This section introduces the basic notations of hypergraphs and conventional hypergraph ranking models.

5.2.1 Notations of Hypergraphs

An ordinary graph is a representation of a set of vertices where some pairs of vertices are connected by (pairwise) edges. While, a hypergraph is a generalization of an ordinary graph where edges, called hyperedges, can connect any number of vertices

(Zhou et al., 2006). In other words, a hyperedge represents a high-order relation of a set of vertices instead of only two vertices. Formally, the definition of a hypergraph can be given as follows.

Definition 5-1 (Hypergraph). A hypergraph is a pair $G=(V,E_h)$ where $V=\{v_1,v_2,\dots\}$ is the vertex set representing a finite number of objects, and $E_h=\{e_1,e_2,\dots\}$ is the hyperedge set representing the high-order relations of the objects. Each hyperedge $e\in E_h$ is a non-empty subset of V, to which a weighting function $w:E_h\to\mathbb{R}^+$ is assigned indicating the quantitative strength of a hyperedge.

Let $h(v,e)=1$ denote that a vertex v is in (connected by) a hyperedge e and $h(v,e)=0$ otherwise, then an incidence matrix $\mathbf{H}\in\mathbb{R}^{|V|\times|E_h|}$ can be established for a hypergraph as follows.

$$H_{ij}=h(v_i,e_j)=\begin{cases}1, & \text{if } v_i\in e_j\\ 0, & \text{otherwise}\end{cases} \tag{5.1}$$

The degree of a vertex $v\in V$ is defined as:

$$d(v)=\sum_{e\in E_h} w(e)h(v,e). \tag{5.2}$$

In addition, the degree of a hyperedge $\delta(e)$ is defined by the number of vertices connected by this hyperedge, given by:

$$\delta(e)=\sum_{v\in V} h(v,e). \tag{5.3}$$

Thus, an ordinary graph can be seen as a special case of hypergraphs where the edge degrees always being two. Throughout the rest of the chapter, we define the diagonal matrix forms for $w(e)$, $\delta(e)$ and $d(v)$ as $\mathbf{W} \in \mathbb{R}^{|E_h| \times |E_h|}$, $\mathbf{D}_e \in \mathbb{Z}^{|E_h| \times |E_h|}$ and $\mathbf{D}_v \in \mathbb{R}^{|V| \times |V|}$, respectively.

5.2.2 Hypergraph Ranking

As mentioned in Section 4.4.1, the ordinary graph ranking problem is considered as seeking for a function $f : V \rightarrow \mathbb{R}$ to label each vertex based on the graph and a given query vector $\mathbf{y} = [y_1, y_2, \ldots, y_{|V|}]^T$, of which a small number of vertices have been initially given the input scores. Similarly, the ranking scores f for vertices can also be seen as a column vector $\mathbf{f} = [f_1, f_2, \ldots, f_{|V|}]^T$. The hypergraph ranking problem has the same goal of simple graph ranking to seek the ranking order $\mathbf{f}$ for the vertices with an input query vector $\mathbf{y}$.

As introduced in Chapter 3, the simple graph ranking problem has been studied in considerable depth in recent years, and some solutions have been readily adopted. For hypergraphs, Zhou et al. (2006) proposes the regularization framework for hypergraph clustering and ranking to solve the following cost function:

$$\min_{f:V \rightarrow \mathbb{R}} Q(\mathbf{f}) = \mathbf{f}^T(\mathbf{I} - \mathbf{A})\mathbf{f} + \mu(\mathbf{f} - \mathbf{y})^T(\mathbf{f} - \mathbf{y}), \tag{5.4}$$

where $\mu > 0$ is the regularization parameter, and matrix $\mathbf{A}$ is defined as follows.

$$\mathbf{A} = \mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}} \tag{5.5}$$

Vanishing the gradient of (5.4) and following the similar calculations of ordinary graph ranking as in (4.21), the optimal ranking result $\mathbf{f}^*$ can be obtained as follows.

$$\mathbf{f}^* = (\mathbf{I} - \alpha\mathbf{A})^{-1}\mathbf{y} \tag{5.6}$$

The regularized framework of hypergraph ranking has been recently introduced into social music recommender systems, such as in (Tan et al., 2011) and (Theodoridis et al., 2013).

5.2.3 Hypergraph Random Walks

The random walk theory has also been employed to rank hypergraph data. Given a hypergraph and a starting point the runner randomly moves to a neighbour of it and again moves to a neighbour's neighbour, etc. Finally, the stationary visiting probability distribution of vertices is treated as their ranking scores. In a hypergraph, the transition probability of moving from the current vertex u to another vertex v is conventionally defined as follows:

$$p(v|u) = \sum_{e:u,v\in e} \frac{w(e)}{d(u)\delta(e)}. \tag{5.7}$$

Thus a transition matrix $\mathbf{T}^{|V|\times|V|}$ with $T_{ij} = p(v_j|v_i)$ can be obtained to update the probability distribution of vertices. The matrix $\mathbf{T}$ is calculated in the following equation:

$$\mathbf{T} = \mathbf{D}_v^{-1}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T. \tag{5.8}$$

Let $\mathbf{p}^{(t)}$ be a column vector where $p_i^{(t)}$ denotes the probability of that v_i is being visited at a certain time t, and we assume multiple random walkers are started from a given (small) query vertex set $V_q \subset V$. Correspondingly, a column vector $\mathbf{q}$ is constructed where q_i is the probability of the i-th vertex v_i being selected as a starting point.

According to the random walk with restarts theory (Tong et al., 2006; Konstas et al., 2009), the walking process is considered to have a probability of $1-\alpha$ to restart from the initial query set at every step. The updating formula of $\mathbf{p}^{(t)}$ is then obtained as follows:

$$\mathbf{p}^{(t+1)} = \alpha \mathbf{T} \mathbf{p}^{(t)} + (1-\alpha)\mathbf{q}. \tag{5.9}$$

The stationary probabilities when the above equation reaches convergence represent the long term visiting rates of vertices, which can be seen as the vertex ranking scores. Let $\mathbf{p}^{(\infty)} = \mathbf{p}^{(t)} = \mathbf{p}^{(t+1)}$ and we obtain:

$$\mathbf{p}^{(\infty)} = (1-\alpha)(\mathbf{I} - \alpha \mathbf{T})^{-1}\mathbf{q}. \tag{5.10}$$

As $1-\alpha \in (0,1)$ does not change the ranking order, we let the optimized ranking vector $\mathbf{f}^*$ be the following form:

$$\mathbf{f}^* = (\mathbf{I} - \alpha \mathbf{T})^{-1}\mathbf{q}. \tag{5.11}$$

The above formula has a similar expression of (5.6). It is also easy to see that $\mathbf{A}$ is essentially a normalized version of $\mathbf{T}$. Thus, the regularization framework of hypergraph ranking can be explained using the hypergraph random walks with restarting theory.

5.3 The Data Model and Multipartite Hypergraph

In this section, we consider the application of a *Restaurant Recommender System* (RRS) that assists customers to find appropriate restaurants. In this scenario, there exist various objects and relations that are valuable for profiling customer preference, such as customer visiting history, restaurant attributes and contextual information. To include all possible information entities and relations, we propose a generic User-Item-Attribute-Context data model as illustrated in Figure 5-2. The entities and relations are then elaborated in the following.

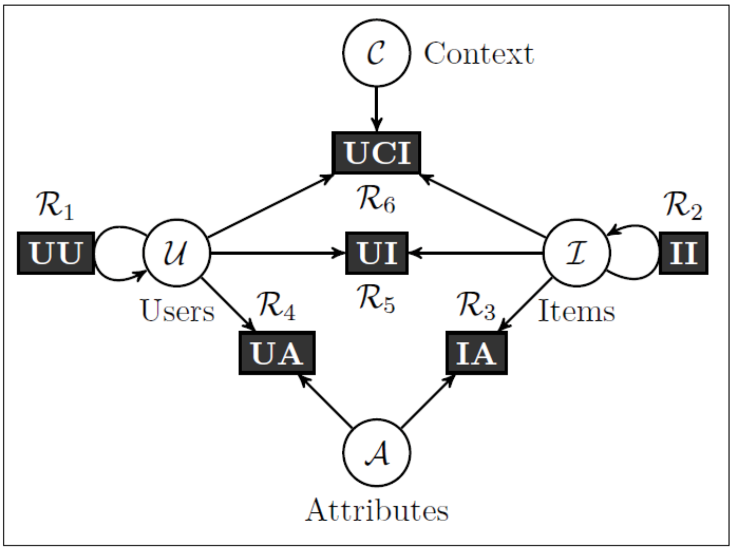


Figure 5-2 The proposed User-Item-Attribute-Context data model

5.3.1 Information Entities

We consider the following four types of information entities (objects) in RRS: *Users*, *Restaurant,* restaurant *Attribute* and *Context* information, as detailed in the follows.

User ($\mathcal{U}$) Users are the registered customers and requesters in RRS. Users are also encouraged to make ratings or comments to the restaurants they have visited, as a one facet of resources for preference profiling.

Item ($\mathcal{I}$) Restaurants are the recommended objects, which are commonly called as "items" in recommender systems. The ultimate goal of RSS is to recommend a list of restaurants as the alternative options for particular request user.

Attribute ($\mathcal{A}$) The attributes represent the descriptions of items such as restaurant conditions, food cuisines, etc. As mentioned in Chapter 3, the taxonomy attributes of items may contain tree-structured features from different aspects. In this chapter, to import attributes as special vertices to construct a hypergraph model, we only consider then end-level attributes from different aspects as a special type of information objects, denoted as $\mathcal{A} = \{a_1, a_2, \ldots\}$. Figure 5-3 shows an example of the set of attributes in terms of five aspects of restaurants, where there are initially five aspects of attributes denoted as $\mathcal{A}_1$ to $\mathcal{A}_5$, and the whole attribute set is set to be $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \ldots \cup \mathcal{A}_5$.

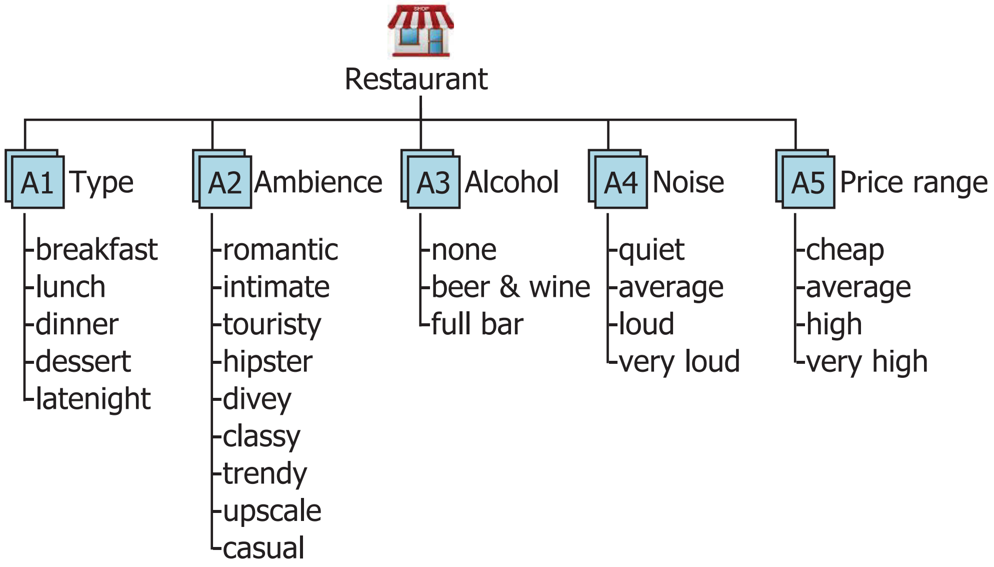


Figure 5-3 An example of restaurant attributes in terms of five aspects

Context ($\mathcal{C}$) This information indicates the environmental situation including temporal context (e.g. the season), spacial context (e.g. the location of both users and restaurant), or any context that may affect a user's choice for items (Dey et al., 2001). Despite the environmental context, we also include the textual *comments* as additional context entities. Thus, we extend the "context" objects to all third-party resources beyond users and items that may affect or reflect users' preferences. The whole context set is denoted as $\mathcal{C}$ in our model.

In summary, we can possibly collect four groups of objects, i.e., users, items, attributes and context, in a recommender system; thereby we call our data model as the U-I-A-C data model. In the following, we elaborate six types of possible pairwise or high-order relations between the same or different groups of objects.

5.3.2 Possible Relations between Different Objects

According to the connected objects, the complex relations between the mentioned four types of information entities can be classified into six categories, as detailed in the following.

UU: User-User relations. As mentioned, various kinds of social relations or implicit correlations between users have attracted increasingly attentions in recommender systems to alleviate the sparseness problem of rating data (Jamali and Ester, 2009; Massa and Avesani, 2007; Yang et al., 2012). In the day of Web.2.0, the complex relations between users lie in the following two aspects. First, the same two users may be connected by different types of relations which have been modelled as multigraph models in the last chapter. Second, there are also many high-order relations that connected more than two users, which have to be modelled by hypergraphs. It is clear that a hypergraph can be seen as also a generalization of multigraph models, in which multiple relations are allowed for a same group of vertices. An example of such high-order relations is the interest groups in some recommender systems (Rae et al., 2010), which are more appropriately considered as multiple-to-multiple relations among the whole group users rather than single-to-single relations between every two of them.

II: Item-Item relations. There are also various kinds of relations between items. In our case, restaurants may be connected as they are owned by same catering groups or they are located in near places, etc. Some of these relations are naturally not pairwise relations. Figure 5-4 shows an example of the inclusion relations of KFC and MacDonald restaurant groups. Furthermore, Figure 5-5 indicates the restaurant neighbourhoods in Sydney city.

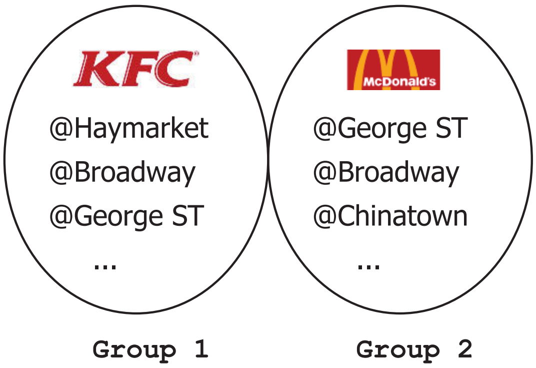


Figure 5-4 An example of catering group inclusion relations of restaurants

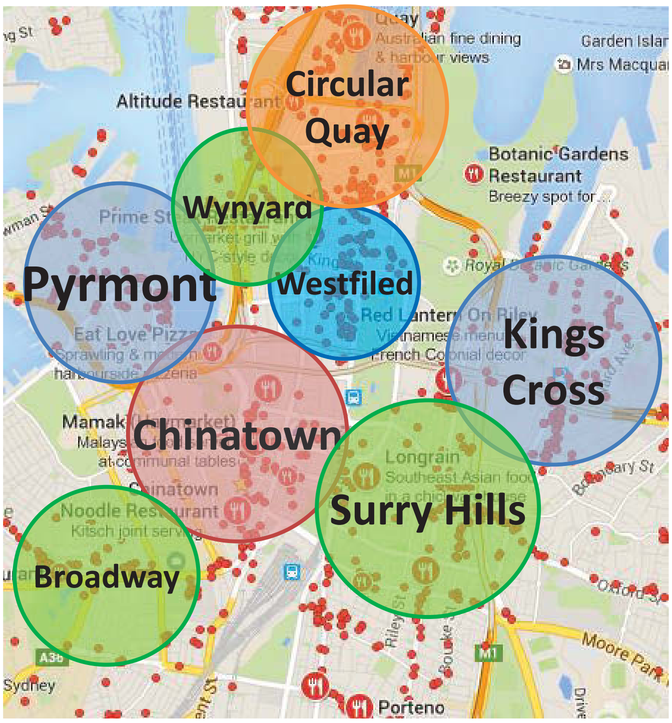


Figure 5-5 An example of neighbourhood relations of restaurants

IA: Item-Attribute relations. Traditionally, the item-attribute association relations are modelled as pairwise relations between items and each associated attributes. In our study, however, all associated attributes of a particular item and this item itself are seen as connected by a single high-order relation.

UA: User-Attribute relations. Users may have particular preference for item attributes, such as movie genres. Similar to the item-attribute association, we assign each user a high-order relation connecting this particular user and the attributes preferred by this user.

UI: User-Item relations. This type of relations contains binary or scaled preferences of users to items. Binary preference includes whether a user like/accept/visit an item, etc. Scaled preferences refer to numerical ratings such as 1 (the worst) to 5 (the best) in most recommender systems.

UCI: User-Context-Item relations. If a users' preference to items is also impact by third-party context information, the 2-dimension preference relation, can be represented as $user \times item \rightarrow utility$, is then extended to a 3-dimension preference relation $user \times context \times item \rightarrow utility$ (Adomavicius and Tuzhilin, 2011). Thus, a high-order relation $\langle user, context, item \rangle$ is raised among the three parties of objects. As we also include the textual comments as extended context entities, a user's comments issued to an item can be handled as a complex relation that connects the user, the item and a number of words. Here, we treat a comment as "a bag of words" as in many textual mining studies (Blei et al., 2003) and the applications in recommender systems (McAuley and Leskovec, 2013). The user-comment-item relation is then represented as a high-order relation between a set of objects, such as $\langle user, word\ a, word\ b, \ldots, item \rangle$. Figure 5-6 shows our presentation the user-comment- restaurant high-order relations.

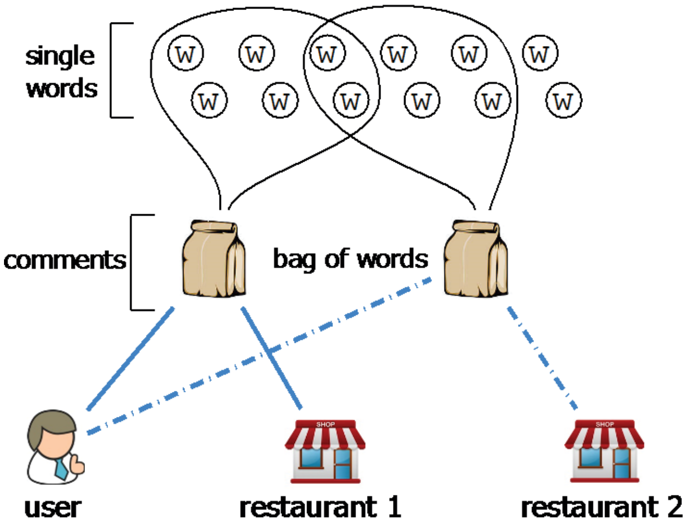


Figure 5-6 The user-comment-restaurant high-order relations

In the above, we exploit six types of relations between the four parties of objects that may appear in a restaurant recommender system. Because some high-order relations are included, which cannot be handled by ordinary graph models, we propose a multipartite hypergraph model to represent this data model.

5.3.3 Multipartite Hypergraph Construction

To handle different information entities and relations, we construct a multipartite hypergraph $G = \{V, E_h\}$ with a unified vertex set V and a hyperedge set E_h, associated with a weighting function $w : E_h \to \mathbb{R}^+$. The unified vertex set is the union of the four parties of objects, i.e., $V = \mathcal{U} \cup \mathcal{I} \cup \mathcal{A} \cup \mathcal{C}$. The unified hyperedge set contains all possible relations, i.e., $E_h = \mathbf{UU} \cup \mathbf{II} \cup \mathbf{IA} \cup \mathbf{UA} \cup \mathbf{UI} \cup \mathbf{UCI}$. As an example, the weights of hyperedges are initialized as follows. (Note that this setting is only for the scenario of a restaurant recommender system using a dataset of Yelp.com, referring to Section 5.5.1.)

– $E_h^{(1)} \leftarrow \mathbf{UU}$: Initially, the online friendship in the dataset is imported as a pairwise relation. We build a hyperedge $\langle u_1, u_2 \rangle \in E_h^{(1)}$ for a pair of user vertices if they are friends, and set the weight to be 1.

- $E_h^{(2)} \leftarrow \mathbf{II}$: We build a hyperedge $\langle i_1, i_2, i_3, \ldots \rangle \in E_h^{(2)}$ to connect those restaurant vertices in a same neighbourhood. The weight of each hyperedge is set to be 1.
- $E_h^{(3)} \leftarrow \mathbf{IA}$: A hyperedge $\langle i, a_1, a_2, \ldots \rangle \in E_h^{(3)}$ is built for each restaurant vertex, which contains the restaurant itself and all associated attribute vertices. The weight of each of this type hyperedge is initialized to be 1.
- $E_h^{(4)} \leftarrow \mathbf{UA}$: Similarly, we build a hyperedge including a user vertex and his/her preferred attributes, i.e., $\langle u, a_1, a_2, \ldots \rangle \in E_h^{(4)}$, of which the weight is set to be 1.
- $E_h^{(5)} \leftarrow \mathbf{UI}$: We build a pairwise edge $\langle u, i \rangle \in E_h^{(5)}$ for a particular user and an item if this user has rated the item, and the normalized rating value is set to be the hyperedge weight.
- $E_h^{(6)} \leftarrow \mathbf{UCI}$: The high-order relation between a user vertex $u \in \mathcal{U}$, a restaurant vertex $i \in \mathcal{I}$ and a number of context vertex $c_1, c_2, \ldots \in \mathcal{C}$ is handled as a hyperedge $\langle u, c_1, c_2, \ldots, i \rangle \in E_h^{(6)}$. The weight of this type edge is initialized to be 1.

Consequently, we obtain the structure of the constructed multipartite hypergraph in terms of the incidence matrix $\mathbf{H}$, as shown in Table 5-1. We also define the following matrices: vertex degree matrix $\mathbf{D}_v$ with size $|V| \times |V|$, edge degree matrix $\mathbf{D}_e$ and weighting matrix $\mathbf{W}$ with size $|E_h| \times |E_h|$, as introduced in Section 5.2.

Table 5-1 The incidence matrix $\mathbf{H}$ and its sub-matrices

Edges Vertices	$E^{(1)}$	$E^{(2)}$	$E^{(3)}$	$E^{(4)}$	$E^{(5)}$	$E^{(6)}$
$\mathcal{U}$	$UE^{(1)}$			$UE^{(4)}$	$UE^{(5)}$	$UE^{(6)}$
$\mathcal{I}$		$IE^{(2)}$	$IE^{(3)}$		$IE^{(5)}$	$IE^{(6)}$
$\mathcal{A}$			$AE^{(3)}$	$AE^{(4)}$		
$\mathcal{C}$						$CE^{(6)}$
relations	**UU**	**II**	**IA**	**UA**	**UI**	**UCI**

5.4 Balanced Hypergraph Ranking and Recommendations

In this section, we point out the bias of traditional hypergraph ranking methods for the multipartite hypergraphs where different types of hyperedges usually vary greatly with each other in the number of contained vertices. Thus we proposed a balanced hypergraph ranking model to replace the traditional methods. We would like to present our ranking model at first, and then indicate the drawbacks of traditional models by comparing them using a simple example.

5.4.1 Balanced Hypergraph Ranking Model

For a multipartite hypergraph where the hyperedges vary greatly in the number of connected vertices, we propose an enhanced vertex degree metric as follows.

$$d_{+}(v) = \sum_{e \in E_h} h(v,e)w(e)\sqrt{\delta(e)} \tag{5.12}$$

Differing from the traditional vertex degree as in Equation (5.2), the enhanced vertex degree metric also considers the degree information of related hyperedges. Similarity, we define a new vertex degree matrix $\mathbf{D}_{+}^{|V|\times|V|}$ with $D_{+(ii)} = d_{+}(v_i)$ on the diagonal.

The cost function of Equation (5.4) is then modified in the following form with the enhanced vertex degree metric.

$$\begin{aligned} Q(\mathbf{f}) =& \frac{1}{2}\sum_{i,j=1}^{|V|}\sum_{e\in E}\frac{w(e)h(v_i,e)h(v_j,e)}{\sqrt{\delta(e)}}\left(\frac{f_i}{\sqrt{d_{+}(v_i)}}-\frac{f_j}{\sqrt{d_{+}(v_j)}}\right)^2 \\ &+\mu\sum_{i=1}^{|V|}(f_i-y_i)^2 \end{aligned} \tag{5.13}$$

In the above equation, the first part of the right side denotes the smoothness function $\mathcal{S}(f)$. We define a new matrix $\mathbf{A}_{+}$ as:

$$\mathbf{A}_{+} = \mathbf{D}_{+}^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_{e}^{-1/2}\mathbf{H}^{T}\mathbf{D}_{+}^{-1/2}. \tag{5.14}$$

Then, the smoothness function can be simplified in the following steps:

$$\begin{aligned} \mathcal{S}(f) =& \frac{1}{2}\sum_{i,j=1}^{|V|}\sum_{e\in E_h}\frac{w(e)h(v_i,e)h(v_j,e)}{\sqrt{\delta(e)}}\left(\frac{f_i}{\sqrt{d_{+}(v_i)}}-\frac{f_j}{\sqrt{d_{+}(v_j)}}\right)^2 \\ =& \sum_{i,j=1}^{|V|}\sum_{e\in E_h}\frac{w(e)h(v_i,e)h(v_j,e)}{\sqrt{\delta(e)}}\left(\frac{f_i^2}{d_{+}(v_i)}-\frac{f_if_j}{\sqrt{d_{+}(v_i)}\sqrt{d_{+}(v_j)}}\right) \\ =& \sum_{i=1}^{|V|}f_i^2\sum_{e\in E_h}\frac{w(e)h(v_i,e)\sqrt{\delta(e)}}{d_{+}(v_i)}\sum_{j=1}^{|V|}\frac{h(v_j,e)}{\delta(e)} \\ &-\sum_{i,j=1}^{|V|}\sum_{e\in E_h}\frac{f_iw(e)h(v_i,e)h(v_j,e)f_j}{\sqrt{d_{+}(v_i)}\sqrt{\delta(e)}\sqrt{d_{+}(v_j)}} \\ =& \mathbf{f}^T\mathbf{f}-\mathbf{f}^T\mathbf{D}_{+}^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_{e}^{-1/2}\mathbf{H}^T\mathbf{D}_{+}^{-1/2}\mathbf{f} \\ =& \mathbf{f}^T\left(\mathbf{I}-\mathbf{A}_{+}\right)\mathbf{f} \end{aligned} \tag{5.15}$$

Performing the similar calculation of Equation (5.6), the optimized balanced ranking order is obtained as:

$$\mathbf{f}^* = \left(\mathbf{I} - \alpha\mathbf{A}_+\right)^{-1}\mathbf{y}, \tag{5.16}$$

where y represents the ranking scores of a small part of vertices as the input query vector.

5.4.2 Comparison with Traditional Models

We point out that the traditional hypergraph ranking model mentioned in Section 5.2 will produce a bias if the degrees of hyperedges vary greatly, which can be explained using random walks. A first example is given in Figure 5-7 (a), where we assume the runner is currently at the vertex A and will move to one of the other three nodes B, C and D via two hyperedges e_1 and e_2. Notice that e_1 is a 3-degree hyperedge and e_2 is a 2-degree (pairwise) hyperedge. The weightings of the both edges are supposed to be 1. With Equation (5.7) and assuming the runner will not stay at node A, the runner will have $1/2$ probability to select hyperedge e_2 and move to node B. The runner also has $1/2$ probability to select edge e_1 and then has $1/4$ to move to node C or D, as e_1 connects C and D simultaneously. Thus, the probability of moving to each node is obtained as $p(B) = 1/2$ and $p(C) = p(D) = 1/4$. We can find that the probability of moving to node B is twice as the probability of moving to node C. However, the hyperedge e_1 and hyperedge e_2 have the same strength initially, such that, B and C or D should be of equivalent strength of relations as for the node A. In other words, this setting biases to select node B, which is connected with a lower-degree hyperedge.

A more significant but common example is given in Figure 5-7 (b), where the degrees of e_1 and e_2 are 21 and 2, respectively. We assume $w(e_1) = 1$ and

$w(e_2)=0.5$, saying that the nodes in hyperedge e_1 have stronger connections than the nodes in e_2, i.e., nodes A and B. Using the traditional ranking model, the probability of moving to a single node in hyperedge e_1 is $1/30$, which becomes much lower than the probability of moving to node B with $p(B)=1/3$, though the fact is that e_1 has a higher (double) strength than e_2. This finding indicates that, for the hypergraphs with different hyperedges such as the multipartite hypergraphs in our study, the truly strong hyperedges may be discarded only because they connected a large number of vertices.

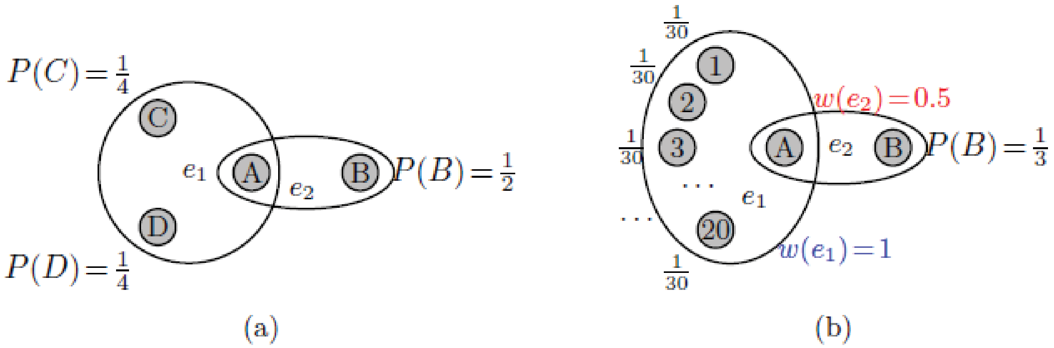


Figure 5-7 The bias of traditional hypergraph ranking model

To alleviate this bias, we can modify the transition equation (5.7) as follows.

$$\begin{aligned} p(v|u) &= \sum_{e:u,v\in e} \frac{w(e)\sqrt{\delta(e)}}{\sum_{e:v\in e} w(e)\sqrt{\delta(e)}} \frac{1}{\delta(e)} \\ &= \sum_{e:u,v\in e} \frac{w(e)}{\sqrt{\delta(e)} d_+(v)} \end{aligned} \tag{5.17}$$

This setting improves the probability of moving to the nodes in higher-degree hyperedges. For example, the balanced result of the example in Figure 5-7 (b) will be obtained as follows.

$$p(B) = \frac{0.5 \times \sqrt{2}}{0.5 \times \sqrt{2} + 1 \times \sqrt{20}} \times 1 \approx 0.137 < \frac{1}{3}$$
$$p(1) = \ldots = p(20) = \frac{1 \times \sqrt{20}}{0.5 \times \sqrt{2} + 1 \times \sqrt{20}} \times \frac{1}{20} \approx 0.043 > \frac{1}{30}$$

With the modified transition equation, the probability of selecting lower-degree hyperedges is reduced. A new transition matrix $\mathbf{T}'_+$ is hence given as follows, which can be seen as the random walk version of the matrix $\mathbf{A}_+$ in Equation (5.14) of our proposed balanced hypergraph ranking model. Actually, $\mathbf{A}_+$ can be seen as just a normalized version of $\mathbf{T}_+$, just like that, in traditional models, the matrix (5.5) is a normalized version of the matrix (5.8).

$$\mathbf{T}_+ = \mathbf{D}_+^{-1}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1/2}\mathbf{H}^T \tag{5.18}$$

5.4.3 Query Vector and Recommendation

Assuming the current request user u is the i-th vertex $v_i \in V$ in the unified hypergraph, the i-th row of the matrix $\mathbf{A}_+$ (or $\mathbf{T}'_+$ for the random walk-based version) can be thought of an initial evaluation of the closeness of this vertex to other vertices and selected as the input query vector of the proposed balanced hypergraph ranking model. Therefore, we initialize a column vector $\mathbf{y}^u \in \mathbb{R}^{|V|}$ as the input query vector as follows.

$$\mathbf{y}_j^u = (\mathbf{A}_+)_{ij} \tag{5.19}$$

For the particular requester u, Equation (5.16) can be solved with query vector $\mathbf{y}^u$ as input and we obtain a ranking vector $\mathbf{f}^*$ of all vertices. Particularly, the ranking scores of all item vertices (restaurants in our example) are extracted and the top-ranked items are selected as the recommendation list for the request user u. So far, the personal recommendation for the requester is completed.

5.5 Experiments

Empirical experiments are conducted on a dataset of Yelp.com[1], which is a local business review site whereby users can give numerical ratings and also textual comments to the merchants they have known or visited in the past. There are also various information entities and complex relations available in this dataset and this dataset can be seen as a representative case of real-world e-Commerce recommender systems.

5.5.1 Data Sampling

The following four parties of information entities are collected as the unified vertices to build a multipartite hypergraph model.

- **Restaurants**: The dataset contains 4119 restaurants in total, for the scenario of restaurant recommendations.
- **Users**: There are 1911 users who have given more than 20 ratings or comments to the selected restaurants. These users are chosen as the user vertices in our hypergraph model.
- **Attribute**s: We extract 24 restaurant attributes from five perspectives as the attribute set, referring to Figure 5-3.
- **Context entities**: In the dataset, there are in total 28282 short comments (called as “tips” in Yelp.com) issued by the users to visited restaurants. We split these comments into single words. The most common words[2] and meaningless symbols are pruned. As a result, there are totally 7205 single words extracted as context entities. Thus, a comment record can be thought of

[1] http://www.yelp.com.au/dataset_challenge

[2] Based on the “Long Stopword List” downloaded from http://www.ranks.nl/stopwords

a high-order relation between the user, the item and some single words, i.e., the special context entities.

Table 5-2 Statistical information of the constructed hypergraph from Yelp dataset

vertex/hyperedge	objects/relations	Sample size*	Avg. Degree**
$\mathcal{U}$	users	1911	79.9
$\mathcal{I}$	restaurants	4119	17.7
$\mathcal{A}$	attributes	24	851.6
$\mathcal{C}$	context	7205	19.2
$E^{(1)}$	UU: friendship	43760	2
$E^{(2)}$	II: neighbourhood	16	226.7
$E^{(3)}$	IA: category	4032	6.1
$E^{(4)}$	UA: preference	1899	3.4
$E^{(5)}$	UI: rating	60688	2
$E^{(6)}$	UCI: comments	24541	7.6

* the total number of this type vertices or hyperedges;

** the average degree of this type vertices or hyperedges.

The ratings in the dataset range from 1 star (the worst) to 5 stars (the best). It is known that the purpose of a recommender system is to guess the truly preferred items for the active user. In the dataset, the items acquiring high ratings (i.e., 4 or 5 stars) are thought of being preferred by a particular user. So we extract a half of the high rated items of each user to build a test set as the hidden knowledge, and let the compared recommendation approaches to guess the preferred items in the test set. In particular, to ensure there are sufficient data for both training and testing, only the users who have more than 10 “highly rated” items are selected as the tested user. Finally, 1827 out of 1911 users and their 26632 preferred items (could be duplicated) are successfully constructed as the test set, in which per tested user has 14.6 preferred items in average.

Removing the test data, we collect various types of relations from the dataset and construct a multipartite hypergraph, as shown in Table 5-2.

It can be found that the different types of hyperedges of the constructed multipartite hypergraph indeed vary greatly in the edge degree (the number of connected vertices); as a result, the bias problem presented in 5.4.2 may be significant in this environment. The modified balanced ranking model is thereby expected to alleviate this problem in some degrees.

5.5.2 Compared Approaches

In our experiments, each compared recommendation approach is required to predict a short list (top-N) of items for each tested user to guess his/her potentially preferred items in the test data. We compare our model with other five baseline recommendation approaches. The first one is the standard user-based **CF** approach (Resnick et al., 1994) that resorts on only rating data. CF predicts the possible rating of a user to each unknown item (restaurant in our case) and then ranks these items according to the predicted scores. We test a second baseline method that makes non-personalized recommendations, using the restaurant reputations in terms of their average ratings as the ranking scores. We call this method as **AVG** for short. In particular, the contextual graph model proposed by Bogers (2010) can be seen as a represented framework handling various information entities and relations using simple graph random walks. We call this approach as the Simple graph Random Walk (**SGRW**). The fourth approach is the general hypergraph random walk model (marked as **HGRW**). The recommendation (ranking) equation of this model is Equation (5.11). Notice that for a tested user v_i, the input query vector $\mathbf{y}$ is set to be the i-th row of the transition matrix $\mathbf{T}$ in Equation (5.8) . We also compare our model with the latest Music Recommendation via Hypergraph (**MRH**) model of (Tan et al., 2011) . MRH introduces the regularization framework of hypergraph ranking (Zhou et al., 2006) for music recommendations. The ranking equation of MRH is

Equation (5.6). We name our proposed Balanced Hypergraph Ranking algorithm as **BHR**, and denote the random walk version using Equation (5.18) as **BHR(RW)**.

5.5.3 Evaluation Metrics

In our experiments, each compared model is required to recommend N restaurants to guess hidden elements in the test set. We use Precision, Recall, F1 and Mean Average Precision (MAP) as the evaluation metrics to compare the performance of different approaches. Precision is defined as the number of correctly recommended restaurants divided by the number of totally recommended items, i.e., N. Recall is defined as the number of correctly recommended restaurants divided by the number of actually preferred restaurants, i.e., the preferred restaurants appeared in the test data. The definition of F1 and MAP have been introduced in Section 2.4.3.

Here we also propose another metric to evaluate the satisfaction of users. We assume that a tested user will satisfy the recommendations if there is at least one restaurant preferred by this user, i.e., found in the test set. The overall satisfaction degree of a recommender system is then defined by the percentage of satisfied users taking account of all tested user. Thus a new evaluation metric **User Satisfaction** (SA) is defined as follows.

$$\mathrm{SA} = \frac{Number\ of\ satisfied\ users}{Number\ of\ tested\ users} \times 100\% \tag{5.20}$$

In the experiments, every model ranks the candidate items and recommends top-N of them for a particular tested user and we adjust the level of N to make clearer comparisons. Intuitively, the selection of N is expected to be small (i.e., recommending five or ten items) for users' ease to browse and select the options.

5.5.4 Performance Comparison

All mentioned evaluation metrics are computed for every compared recommendation approach. Figure 5-8 firstly shows the Precision-Recall curves of compared approaches. Figure 5-9 presents the values of MAP under different setting of N. It is evident that our proposed BHR approach outperforms all others in most cases, especially at lower ranks. Particularly, the superiority of BHR compared to MRH indicates the success of the proposed balanced hypergraph ranking model, which is suitable for multipartite hypergraphs and improve recommendation accuracy. Besides, the improvement of the random walk-based version BHR(RW) compared to the traditional hypergraph random walk model HGRW is more significant, which indicates the success of our replacement of Equation (5.14) to Equation (5.5).

Figure 5-10 presents the MAP measurements of for the graph or hypergraph-based approaches with different settings of the parameter α, which demonstrates that the best performance is archived by BHR at $\alpha = 0.95$. It is also worth to notice that the MAP scores of both BHR and MRH drop dramatically when $\alpha > 0.95$ and become the lowest levels than other models.

For a recommender system, it is important to be able to successfully discover the right items when recommending only a few items. Thus, we also test the performance of each compared approach when N is set to be 5, i.e., each approach recommends five items for a particular user. The result is presented in Table 5-3, which shows that our approach BHR acquires the best performance in terms of all evaluation metrics. In particular, 57.25% of users will “satisfy” the recommendations of BHR, meaning they can find at least one potentially interested restaurant from the five ones recommended by BHR.

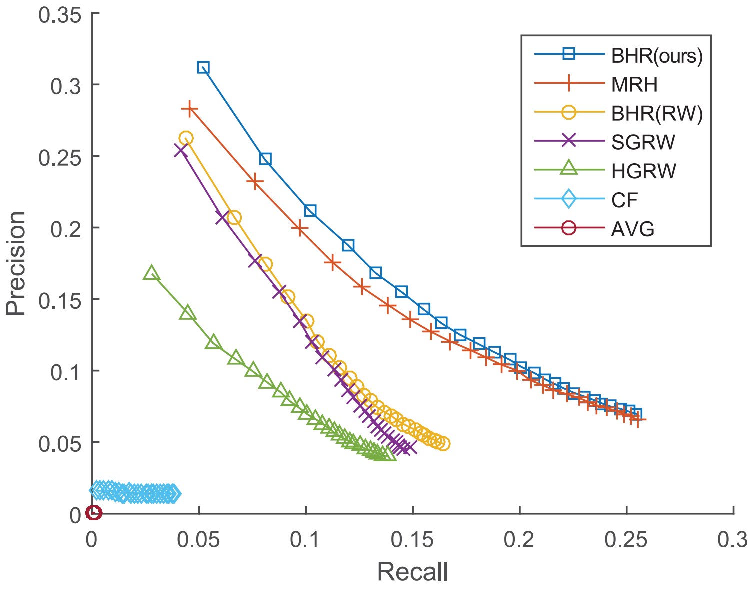


Figure 5-8 Recall-Precision curves for all compared approaches

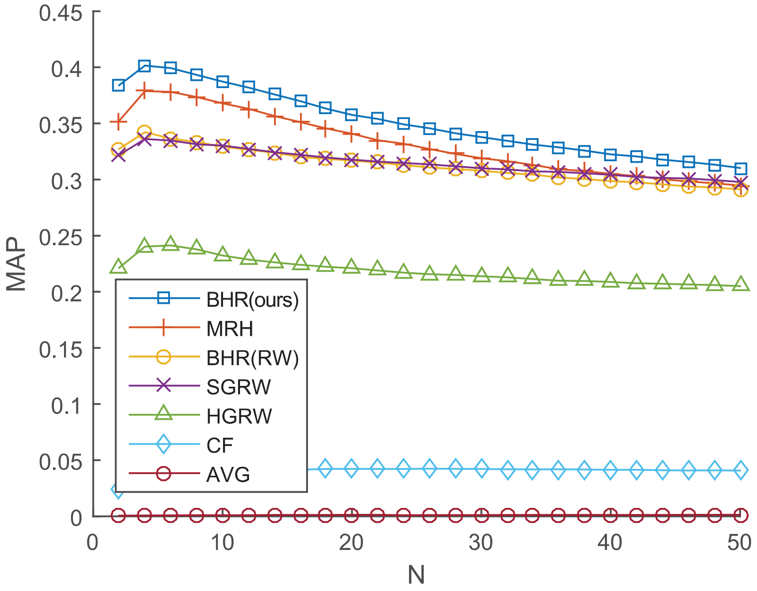


Figure 5-9 The performance in terms of MAP under different levels of top-N

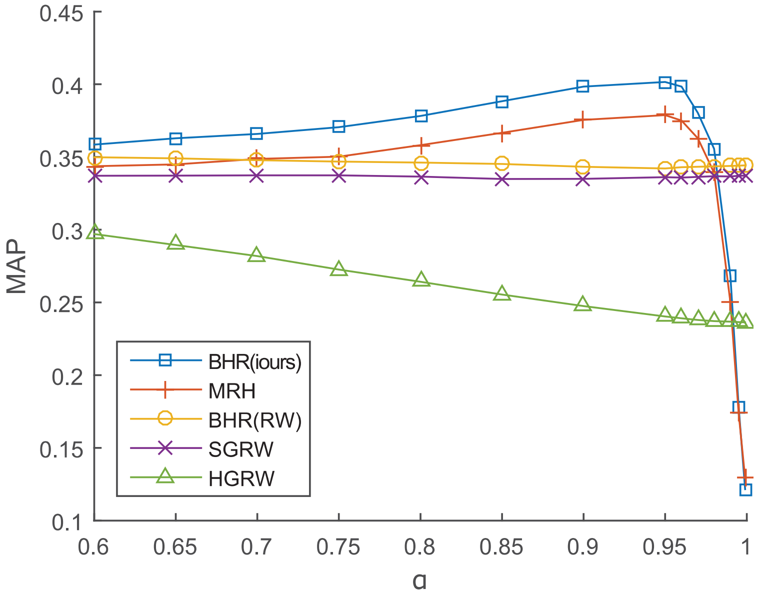


Figure 5-10 The performance in terms of MAP under different level of α

Table 5-3 Performance comparison with N=5

Approaches	Recall	Precision	Satisfaction	F1	MAP
AVG	0.01%	0.03%	0.16%	0.0001	0.0008
CF	0.55%	1.56%	7.28%	0.0077	0.0358
HGRW	5.55%	11.29%	36.23%	0.069	0.2413
SGRW	7.90%	16.88%	46.31%	0.0994	0.3347
BHR(RW)	8.02%	16.56%	46.14%	0.1001	0.3365
MRH	9.69%	18.88%	56.43%	0.1191	0.3781
BHR(our)	**10.01%**	**20.05%***	**57.25%***	**0.124**	**0.3933***

Bold typeface indicates the best performance.

* indicates statistical significance at p<0.01 compared to the second best

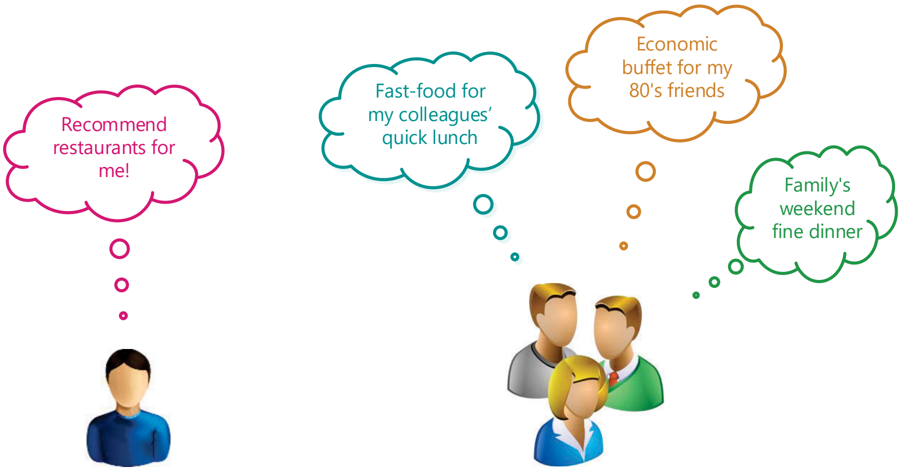
5.6 Summary

In this chapter, we have proposed a generic recommendation approach to handle different information entities and relations that may appear in recommender systems. We have exploited four parties of information entities and six types of pairwise or high-order relations that may contribute to recommendation making and propose a general User-Item-Attribute-Context data model, which can naturally construct a multipartite hypergraph model. As a result, the recommendation problem is transferred to the proposed multipartite hypergraph ranking problem and we solve it using a balanced hypergraph ranking algorithm. Overall, the proposed multipartite hypergraph ranking-based recommendation approach provides a guideline for full-information based recommendations and it is easy to apply in different scenarios. Restaurant recommendation is used as the example scenario in this chapter and the real-world dataset of Yelp.com is abstracted for empirical evaluations. We have conducted a set of comprehensive experiments on this dataset, which demonstrate that our approach can improve recommendation performance significantly, especially in the case of recommending a small number of items. It can be concluded that the proposed multipartite hypergraph-based recommendation approach is able to handle complex input information to produce precise suggestions for users. More importantly, with the generic data model, the proposed approach is easy to apply for different recommendation scenarios.

CHAPTER 6. A MULTI-OBJECTIVE RECOMMENDER SYSTEM VIA HYPERGRAPH RANKING

6.1 Overview

In the last chapter, we propose a hypergraph ranking model BHR to help to discover potential items from various kinds of information entities and relationships for a particular user. Like traditional recommender systems, BHR assumes a single user will not change his/her demand, meaning that the user always has a simple request of finding the best matched items at every time. In practice, however, users may have special requirements at different times, which could be related to multiple objectives. Still taking the restaurant recommendation as an example, a user may have different requirements about restaurant flavours or conditions each time such that it is not appropriate to always suggest same restaurants for this user by only considering his/her past preferences. We consider the changing and flexible demand of users at every time as a multi-objective recommendation request, and we say the traditional unchanged user demands are single-objective requests. Clearly, existing recommender systems are only able to respond to single-objective requests but cannot handle well the mentioned multi-objective requests. Some examples of single-objective and multi-objective recommendation requests are illustrated in Figure 6-1, which indicates that the traditional single-objective recommendation request is changeless for a particular user representing the user's overall preference to all items, while the multi-objective recommendation requests contains different aspects of requirements on item conditions every time.


Recommend restaurants for me!
Fast-food for my colleagues' quick lunch
Economic buffet for my 80's friends
Family's weekend fine dinner

(a) the unchanged single-objective request

(b) three multi-objective requests at different time

Figure 6-1 Examples of single and multi-objective recommendation requests

To handle the changing and flexible multi-objective recommendation requests, this chapter aims to model the complex user requests to computable inputs and proposes a multi-objective recommendation framework. In the rest of this chapter, Section 6.2 indicates that the multi-objective recommendation requests can be decomposed to a set of queried information entities corresponding to the proposed U-I-A-C data model in Chapter 5, which can be further represented by an input "query vector". The multipartite hypergraph ranking model of the previous chapter is employed and Section 6.3 proposes a multi-objective recommendation framework including four major steps of multi-objective requests collection, hypergraph construction, hypergraph ranking, and recommendation generating. In section 6.4, a demonstration restaurant recommender system named FoodGo is developed, which indicates how the multi-objective recommendation framework is implemented and displayed in real applications. We also conduct empirical experiments to evaluate the performance of our recommender system in Section 6.5. The results demonstrate that our approach is able to precisely respond to users' multi-objective requests.

6.2 Multi-objective Request Analysis

As illustrated in Figure 6-1, recommender systems users may have changing demands on the participants, item conditions and environments, which are called by us the *multi-objective recommendation requests*. This complex request of recommendations differs from the *single-objective requests* as in traditional recommender systems. In brief, single-objective recommendations only provide a general option of items for an individual user, while multi-objective recommendations suggest a list of items with respect to the particular situation or requirements each time such as the specific participant users, item attributes and/or environmental information. In combination with the U-I-A-C and hypergraph data model proposed in the last chapter, we can decompose the multi-objective requests as in Figure 6-1 to some constraints in different types of information entities, i.e., a set of query vertices in the hypergraph model, as illustrated in Figure 6-2.

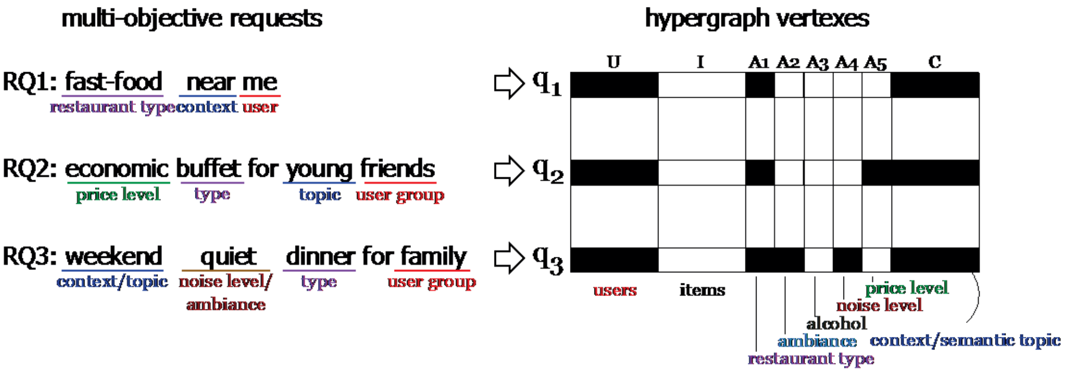


Figure 6-2 Decomposing multi-objective requests to query vertices

In these examples, the request RQ1 "to suggest fast-food near me" actually involves three unique requirements related to three objects/entities. First, it requires the candidate restaurants can provide fast-food, which is related to the *attribute* entities in our U-I-A-C model referring to Figure 5-3. The second, it requires the candidate restaurants are in short distance, which is related to the *context* objects in the U-I-A-C model. The third, it requires the restaurant flavour fits the participant's preference, related to the *user* objects. Therefore, RQ1 can be decomposed to some constraints on a set of related objects, which is called the query set for our multi-objective recommender system.

Similarly, the requests RQ2 and RQ3 can be decomposed as follows.

- RQ2: "economic (*lower price level is better*) buffet (*restaurant type*) for young (*semantic topic*) friends (*specific user group*)".
- RQ3: "weekend (*context or topic*) quiet (*low noise level*) dinner (*restaurant type*) for family (*multiple users*)".

In summary, a multi-objective request can be decomposed to a query set containing a number of objects, i.e., a set of vertices of a multipartite hypergraph constructed by the U-I-A-C data model. Correspondingly, we define a (column) query vector $\mathbf{q} = \{q_1, q_2, \ldots, q_{|V|}\}^T$ to represent the query set as follows.

$$q_i = \begin{cases} 1, & \text{if the } i\text{-th object } v_i \text{ is quered} \\ 0, & \text{if the } i\text{-th object } v_i \text{ is not quered} \end{cases} \tag{6.1}$$

Accordingly, three query vectors $\mathbf{q}_1$ to $\mathbf{q}_3$ are generated from the three multi-objective requests RQ1 to RQ3, respectively, as indicated in Figure 6-2.

It is also noticeable that our setting till works for traditional single-objective recommendation requests which are related to only one object, i.e., the request user itself and the query vector $\mathbf{q}$ will have only one non-zero element corresponding to the request user.

6.3 Multi-objective Recommendation Framework

Although multi-objective recommendations have extra constraints of user requests than traditional single-objective recommendations, they have the same goal which can be thought of seeking for an optimal ranking order of items and selecting the best ones as the output result. Thus, we can still apply the proposed BHR recommendation approach in last chapter by replacing a input query vector to be suitable for users' multi-objective requests.

Concretely, each multi-objective request will be transferred to a query vector $\mathbf{q} \in \mathbb{R}^{|V|}$ where the non-zero elements indicate the user request has special requirements on the correlated vertices in the multipartite hypergraph. To enrich the input information, we use $\mathbf{y} = \mathbf{A}_{+}^{T}\mathbf{q}$ instead of $\mathbf{q}$ as the input of our BHR model. Consequently, the optimization ranking result is rewritten as follows, referring to the solution of Equation (5.16).

$$\mathbf{f}^{*} = (\mathbf{I} - \alpha\mathbf{A}_{+})^{-1}\mathbf{A}_{+}^{T}\mathbf{q} \tag{6.2}$$

The multi-objective recommendations can then be generated by solving (6.2). Particularly, if the query vector $\mathbf{q}$ has only one non-zero element corresponding to the request user itself, $\mathbf{A}_{+}^{T}\mathbf{q}$ becomes exactly the same of Equation (5.19), i.e., the original input query vector of BHR. Accordingly, the multi-objective recommendation framework can be seen as a generalization of the single-objective recommendation approach proposed in the last chapter. In addition, replacing matrix $\mathbf{A}_{+}$ with $\mathbf{A}_{+}$ will return the result of random walk-based version of the proposed BHR algorithm.

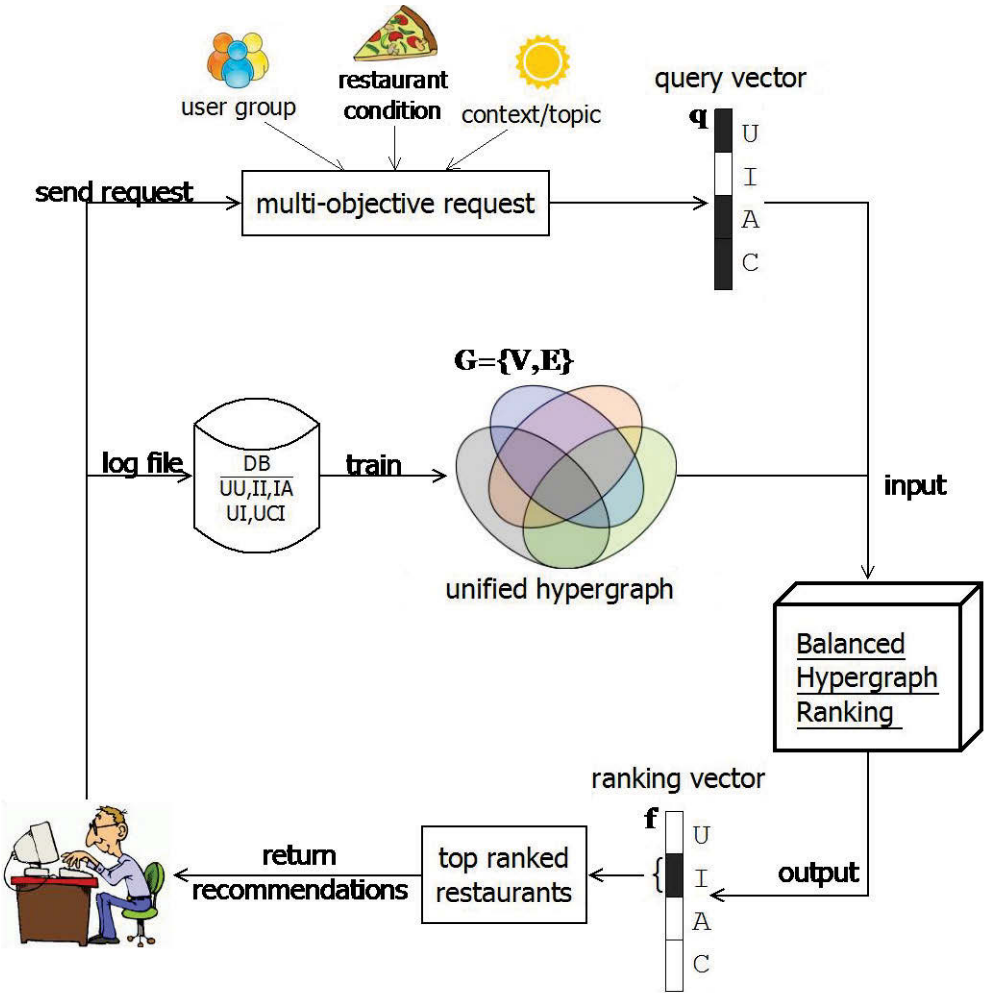


Figure 6-3 Multi-objective recommendation framework

The whole multi-objective recommendation framework of a restaurant recommender system as an example is illustrated in Figure 6-3. To handle multi-objective recommendations, there are four main steps in the framework as detailed as follows.

Step 1. *Query Generation.* This step collects a multi-objective recommendation request and decomposes it to a set of related objects and generates a query vector $\mathbf{q}$, referring to Section 6.2.

Step 2. *Hypergraph Construction.* In this step, a multipartite hypergraph is constructed from various ordinary or high-order relations of users, restaurants, attributes and context information, referring to Section 5.3.

Step 3. *Balanced Hypergraph Ranking.* In this step, the obtained query vector and the multipartite hypergraph are as input to the BHR algorithm. Some important matrices such as $\mathbf{W}$, $\mathbf{D}_{+}$, and $\mathbf{A}_{+}$ are trained in advance. A ranking vector $\mathbf{f}$ will be generated as the output which ranks all hypergraph vertices based on their closeness to the whole query objects.

Step 4. *Recommendation Generation.* This step extracts the ranking order of only restaurants and the top-N ranked ones will be selected as the ultimate recommendations to users responding to their multi-objective recommendation request. So far, a multi-objective recommendation is completed and it is flexible if users to change and resend their requests.

6.4 FoodGo: A Multi-objective Restaurant Recommender System

We develop a protocol of a restaurant recommender system named *FoodGo* to illustrate how multi-objective recommendations are collected and implemented in real-life e-Commerce environment.

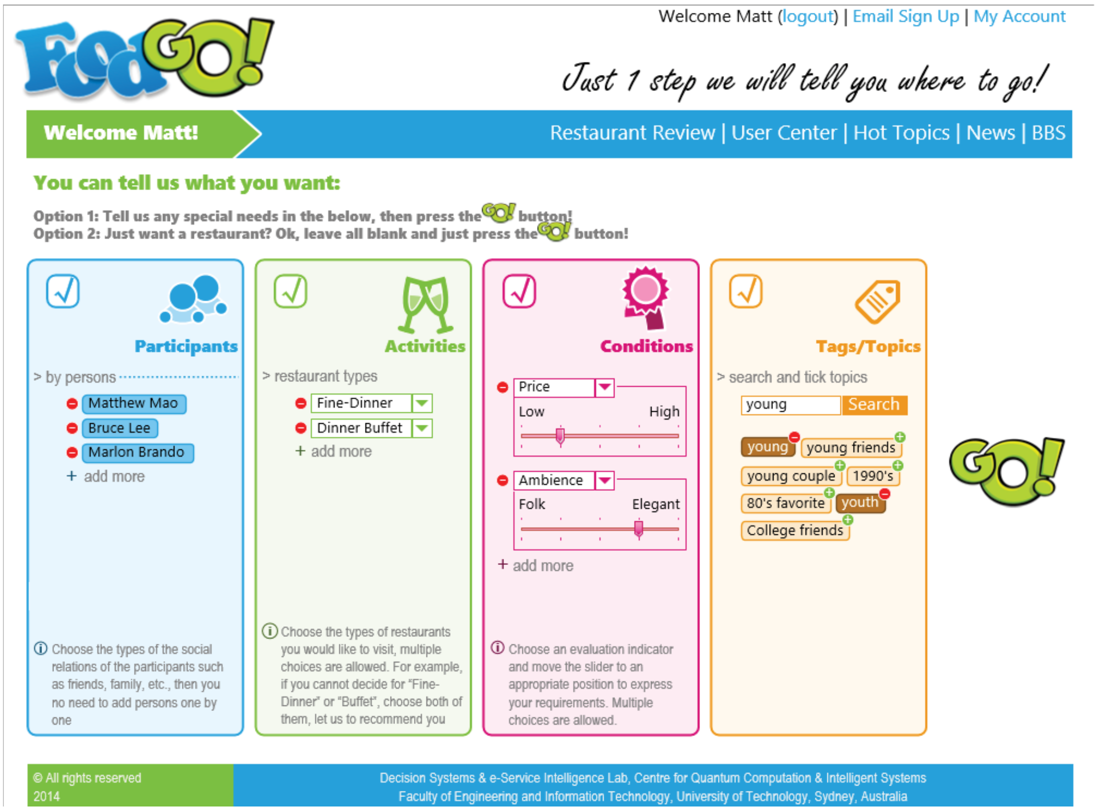
6.4.1 Multi-objective Request Collection

FoodGo is a multi-objective recommender system in food industry, to help single or group users discovering potential restaurants in different criteria. Figure 6-4 shows the start-up page where users can tick their special requirements in the following four aspects.

- **Participants**. Users can invite others as the participants for an activity, thus the recommendations will be generated with regarding to the preferences of all participants rather than an individual user. This is similar to the group recommendations in previous studies (Gorla et al., 2013).

- **Activities**. Users can declare the types of activities, such as dinner, buffet, fast-food, etc. Note that multiple alternative activities are allowed.
- **Restaurant Conditions**. Special requirements of restaurant conditions can be selected by users in terms of the food cuisine, ambience, alcohol, noise level and price level.
- **Topics**. Users can declare extra demands by searching the existed topics abstracted from textual tags or comments.

Note that the users can skip this step as a result the sing-objective recommendations will be generated to match the personalization of only the active user.


Welcome Matt (logout) | Email Sign Up | My Account
FoodGO!
Just 1 step we will tell you where to go!
Welcome Matt!
Restaurant Review | User Center | Hot Topics | News | BBS
Participants
Activities
Conditions
Tags/Topics
GO!

You can tell us what you want:

Option 1: Tell us any special needs in the below, then press the GO! button!
Option 2: Just want a restaurant? Ok, leave all blank and just press the GO! button!

> by persons
Matthew Mao
Bruce Lee
Marlon Brando
+ add more

> restaurant types
Fine-Dinner
Dinner Buffet
+ add more

Price: Low – High
Ambience: Folk – Elegant
+ add more

> search and tick topics
young | Search
young, young friends, young couple, 1990's, 80's favorite, youth, College friends

ⓘ Choose the types of the social relations of the participants such as friends, family, etc., then you no need to add persons one by one

ⓘ Choose the types of restaurants you would like to visit, multiple choices are allowed. For example, if you cannot decide for "Fine-Dinner" or "Buffet", choose both of them, let us to recommend you

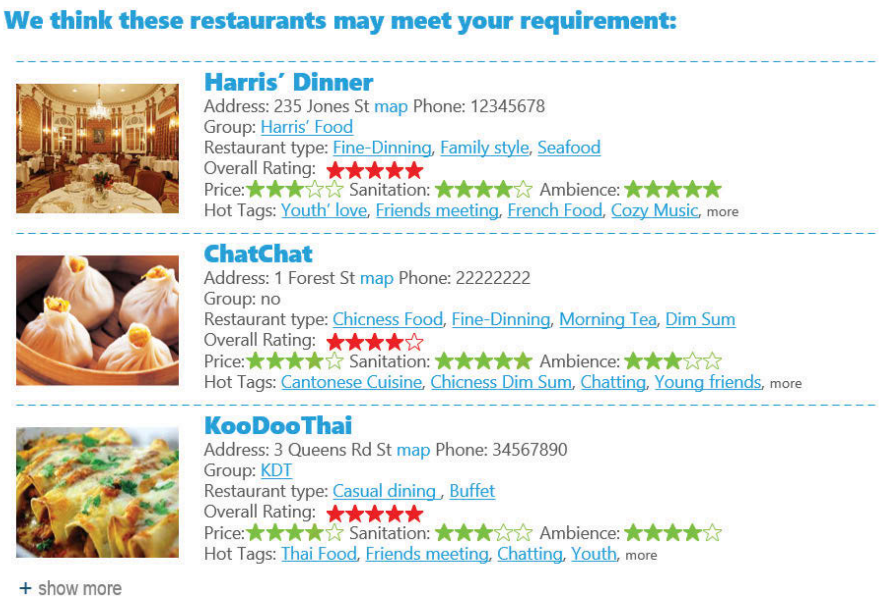
ⓘ Choose an evaluation indicator and move the slider to an appropriate position to express your requirements. Multiple choices are allowed.

© All rights reserved 2014
Decision Systems & e-Service Intelligence Lab, Centre for Quantum Computation & Intelligent Systems
Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia

Figure 6-4 A screenshot of the multi-objective requesting page of FoodGo

6.4.2 Background Model Training

The U-I-A-C data model and constructed multipartite hypergraph are trained and stored in database. In the background, several matrices such as $\mathbf{H}$, $\mathbf{W}$, $\mathbf{D}_+$ and $\mathbf{D}_e$ are ready for the BHR algorithm. Also, the model parameter α is initialized by domain experts or administrators or tuned by off-line experiments. With all trained matrices and parameters, and the collected query vector transferred from users' multi-objective requests, the system will generate a ranking list of all candidate restaurants as the output by solving the BHR algorithm.



We think these restaurants may meet your requirement:

Harris' Dinner
Address: 235 Jones St map Phone: 12345678
Group: Harris' Food
Restaurant type: Fine-Dinning, Family style, Seafood
Overall Rating: ★★★★★
Price:★★★☆☆ Sanitation: ★★★★☆ Ambience: ★★★★★
Hot Tags: Youth' love, Friends meeting, French Food, Cozy Music, more

ChatChat
Address: 1 Forest St map Phone: 22222222
Group: no
Restaurant type: Chicness Food, Fine-Dinning, Morning Tea, Dim Sum
Overall Rating: ★★★★☆
Price:★★★★☆ Sanitation: ★★★★★ Ambience: ★★★☆☆
Hot Tags: Cantonese Cuisine, Chicness Dim Sum, Chatting, Young friends, more

KooDooThai
Address: 3 Queens Rd St map Phone: 34567890
Group: KDT
Restaurant type: Casual dining , Buffet
Overall Rating: ★★★★★
Price:★★★★☆ Sanitation: ★★★☆☆ Ambience: ★★★★☆
Hot Tags: Thai Food, Friends meeting, Chatting, Youth, more

Figure 6-5 Recommendation page of FoodGo

6.4.3 Recommendation Display

By solving Equation (6.2) with the offline trained matrices and online obtained query vector, a ranking order is generated for the alternative restaurants. Consequently, the foreground page will display several (top-N, e.g., 5 or 10) top-ranked restaurants as suggestions. Figure 6-5 is a screenshot of a recommendation result. So far, a list of restaurants is recommended as the response to the multi-objective request. Users can go back to the previous page and re-define their requirements, and then the recommendations shall be refreshed correspondingly. This indicates our system is flexible for dynamic requests of users.

6.5 Empirical Evaluation

We use the same dataset sampled in the last chapter to conduct empirical experiments, as described in Section 5.5.1. Based on the dataset, we compare the ability of each recommendation approach mentioned in Section 5.5.2 in response to multi-objective recommendations.

6.5.1 Generating Multi-objective Requests for Testing

Because no multi-objective requests are explicitly available in the dataset, we randomly select three participants, three restaurant attributes and three topics from the user vertices ($\mathcal{U}$), attribute vertices ($\mathcal{A}$) and context vertices ($\mathcal{C}$), respectively. Notice that the "topics" are selected only from the 200 mostly appeared words in the comments of users. We repeat this selection 1000 times as the tested multi-objective recommendation requests. One of the generated multi-objective requests is shown in the following table as an example.

Table 6-1 A random multi-objective recommendation request and query set

Participant user ID	Restaurant condition	Activity type	hypergraph vertices
$\left\{\begin{array}{l} U_{21} \\ U_{640} \\ U_{660} \end{array}\right.$	$\left\{\begin{array}{l} Alcohol:\ none \\ Noise:\ quiet \\ Prise:\ average \end{array}\right.$	$\left\{\begin{array}{l} \text{“}lunch\text{”} \\ \text{“}pizza\text{”} \\ \text{“}drinks\text{”} \end{array}\right.$	$\{V_{421},\ V_{640},\ V_{660},$ $V_{66045},\ V_{6048},\ V_{6053},$ $V_{9435},\ V_{10345},\ V_{12003}\}$

For each testing multi-objective recommendation request, a query vector $\mathbf{q}$ can be generated as the input for the compared graph/hypergraph ranking-based approaches BHR, BHR(RW), MRH, HGRW and SGRW. For the non-graph-based approaches CF and AVG, we use an "aggregating and filtering" manner to handle multi-objective recommendations. First, a restaurant ranking list is obtained for each single tested user using CF or AVG approach. For a testing multi-objective request correlated to multiple participants, the obtained restaurant ranking scores by the first step of all participants are averaged to output an aggregated result. Next, the restaurants which do not have any relations with the required restaurant attributes and topics in the multi-objective requester are removed from the ranking list and the top-N ones from the rest restaurants are returned as the final recommendations.

6.5.2 Evaluation Method

It is hard to directly evaluate a multi-objective recommendation result because the original data set only records the preferred items of each single user. However, we can generate a ranking order of restaurants only from the test data for a particular multi-objective request and evaluate whether a recommendation approach can carry out correct ranking order of the selected restaurants. Because each testing multi-objective request corresponds to three users as in our experiments, the union set of the preferred items of these users (in the test set) are collected as the candidate items. Denoting the query set corresponding to the current multi-objective request that contains different objects as V_q, then we can estimate the relevance degree of each candidate item to the query set V_q as the basis for ranking. Namely, the relevance of an item i in the candidate set to the query set V_q is defined as follows.

$$rel(i) = \frac{\sum\limits_{v \in V_q} \max\limits_{e \in E_h : i, v \in e} w(e)}{|V_q|} \tag{6.3}$$

The above metric can be seen as the ideal/test ranking score for a candidate item. Given the ranking result of a compared recommendation approach, the NDCG can be measured to evaluate if this approach is able to rank the candidate items correctly.

Table 6-2 NDCG comparison for multi-objective recommendations

Approaches	NDCG@5	NDCG@10	NDCG@30	NDCG@50	NDCG@100
AVG	0.1980	0.2652	0.4826	0.5521	0.5789
CF	0.2392	0.2996	0.5019	0.5719	0.5997
HGRW	0.1336	0.1884	0.4087	0.4982	0.5373
SGRW	0.1348	0.1921	0.4317	0.5121	0.5453
BHR(RW)	0.1392	0.1961	0.4142	0.5021	0.5403
MRH	0.4132	0.5048	0.6560	0.6907	0.7030
BHR(our)	**0.4916***	**0.5688***	**0.6963***	**0.7274***	**0.7396***

Bold typeface indicates the best performance.

* indicates statistical significance at $p<0.001$ compared to the second best

6.5.3 Performance Comparison

Table 6-2 collects the NDCG measurement of each compared approach, which indicates that our approach BHR significantly outperforms other in all cases. This superiority indicates the ranking result of BHR is more consistent with the ideal/actual ranking order in test data, that is, BHR is more effective to identify the potentially interested restaurants with respect to multi-objective recommendation requests.

6.6 Summary

In this chapter, we point out and resolve a novel multi-objective recommendation problem. A multi-objective recommendation framework is proposed using the User-Item-Attribute-Context data model and the multipartite hypergraph ranking model as in Chapter 5. Differing from the traditional single-objective recommendation request, a multi-objective request can be decomposed to a set of additional constraints on multiple information entities such as group users, item conditions and environmental information, which can be modelled as a set of query vertices/objects as the input of multipartite hypergraph ranking. The multi-objective recommendation is hence seen as a generalization of the hypergraph ranking model in Chapter 5 with “advanced” input query vectors. The theoretical framework of the multi-objective recommendations is proposed and the practical application is illustrated by a demonstration recommender system named FoodGo in the scenario of restaurant recommendations. Empirical experiments are also conducted on a real-world dataset of Yelp.com with randomly generated testing multi-objective requests. The result demonstrates that our approach is able to rank the test items precisely, especial when the recommendation size is small.

CHAPTER 7. CONCLUSIONS AND FUTURE STUDY

This chapter concludes the whole thesis and its contributions to the field of recommender systems. It also provides some further research directions of related topics.

7.1 Conclusions

This thesis is motivated by an awareness of practical issues in recommender systems. The rapid growth of both technologies and user volumes of large-scale Web applications has offered both opportunities and challenges to facilitate and improve recommender systems. This research focuses on the following four new trends of recommender systems arising in four aspects: (1) the diverse and complex content information of items, (2) the diverse and complex relations of users, (3) the diverse and complex new information entities and relations, and (4) the flexible and multi-objective user requests. All these trends will provide new advanced features of modern recommender systems but there exists inadequate studies in related topics in the literature. Hence, this research has conducted comprehensive analysis of each of the mentioned challenges and developed a set of novel recommendation techniques or frameworks, and correspondingly, has solved different graph ranking problems abstracted from the recommendation questions.

There are four main contributions of this study, as detailed as follows.

- It proposes a novel taxonomy-folksonomy integrated content comparison mechanism for items (to achieve the research objective 1) and a hybrid recommendation model based on random walking on user-item relational networks (to achieve the research objective 2).

Folksonomy tags can be seen as the user's judgments or supplementary for the standard, official, taxonomy attributes. The folksonomy information is utilized in this research for item content analysis in two ways. First, identifying the correlations between attributes and tags enable us to compare the semantic similarity of two attributes, even though these attributes look "unrelated" from the taxonomy tree. In previous studies, the semantic similarity of attributes, which plays an essential role for the comparison of item similarity, is established by the taxonomy distances (Shambour and Lu, 2011) or manual set up (Wu et al., 2014b). However, the distance-based method is not able to discover the potential correlations between attributes in different aspects such as a director and a movie genre, while the manual set up needs extra domain experts. Thus, the tags can be seen as the natural judgements of users, who are actually more "professional" than the domain experts as they have really experienced the items. The second use of folksonomy information is to identify the most discussed/hot topic of a particular item, e.g., one movie, or a specific domain of items, e.g., the movies. The hot tags tell us the main characteristics of single or a domain of items, which provides a new clue to find similar items. The two usages of tags are well integrated in an overall content similarity induction algorithm which applies a top-down subtree matching manner for the comparison of taxonomy trees. Finally, the overall content similarity correlations of items are imported in a user-item hybrid network and a random walk model is proposed for the bipartite graph. The user-item bipartite graph random walk model is essentially an effective hybrid recommendation model where both CF and

CB ideas are combined. More importantly, the model is convenient to adjust the weights of the two ideas to adapt to different sparsity levels of data. As a guideline from empirical experiments, the CB idea should be highlighted when rating data are sparse, and vice versa.

- It proposes a novel social network propagation model (to achieve the research objective 3) and a multi-relational social network-based recommender system based on multigraph ranking (to achieve the research objective 4).

Social network propagation, based on indirect trust inference as in literature, is one of the key issue in many social network-based recommender systems (Golbeck, 2005; Massa and Avesani, 2009; Shambour and Lu, 2012). Existing approaches usually focus on the perspective of an individual user and to search other indirect users using tree/graph searching techniques such as BFS (breadth-first search) or DFS (depth-first search). There are two flaws for these techniques: firstly, many parameters need to be manually set up such as search depth and width and attenuation; secondly, the explanation from the perspective of the whole network is lacking. The proposed random walk-based social network propagation model is effective in overcoming the two flaws. More importantly, the propagation model is applicable for different user-to-user networks with directed or undirected, scaled or binary explicit or implicit relations of users. The propagation model is used in our research as a pre-processing step to enrich the original social data, after which, there may exist multiple social networks available as the input data of the proposed multigraph ranking-based recommendation model. Compared to other models, our model considers both intra-network relations and inter-network diversities to investigate users' overall closeness in a complex multi-relational environment. In particular, the network-to-network comparison prevents the repeat use of the same resource-derived different social networks. Overall, this study provides a guideline of how to incorporate different social networks to facilitate and improve recommendations. Some key issues of single social network generation and propagation, network-to-network comparison and identification are also comprehensively analysed.

- It proposes a multipartite hypergraph model (to achieve the research objective 5) and a full-information based recommender system using balanced hypergraph ranking algorithm (to achieve the research objective 6).

The real situation of a practical recommender system may be more complex than a single resource-based recommendation model. There are increasing various types of information entities and their relations that may appear in a recommender system and impact user adoption of items. This study presented the User-Item-Attribute-Context data model as a guideline to identify any possible information resources and model them as computable input data. Correspondingly, the multipartite hypergraph model is constructed to retain the original structure information of high-order relations such as the impact of environment context for user adoption of items, and the textual reviews of users to visited items. The hypergraph ranking problem is not new but the traditional model does not perform well for multipartite hypergraphs, especially when the degrees of hyperedges vary greatly. The proposed balanced hypergraph ranking (BHR) algorithm is suitable for this circumstance and has produced excellent results in empirical experiments. The data model and multipartite hypergraph ranking model provide a generic framework to develop a full-information based recommender system to incorporate all valuable information resources in practice.

- It points out the multi-objective recommendation demands of users and proposes a multi-objective recommender system (to achieve the research objective 7).

The multi-objective recommender system aims to advance the functionalities of existing recommender systems to respond to not only single-objective demands of individual users but also the changing multi-objective demands of both single and group users. In practice, users may have extra requirements of item conditions or context every time they use a recommender system. Users are also accompanied with different people and the requesters are actually a group of users rather than a single user. The proposed analysis is effective in decomposing such multi-objective requests to a set of queried information entities corresponding to the multipartite hypergraph vertices based on the U-I-A-C data model. The set of queried entities is further handled as a calculable input query vector, which is just an advanced version of the BHR algorithm, thus the multipartite hypergraph-based recommender system is also applicable for multi-objective recommendations. The FoodGo system gives an example of how the multi-objective recommendations are queried, solved and returned to users in real applications. Both the data model and recommendation model are easy to apply for different domains.

7.2 Future Studies

This research still holds some limitations and can be further advanced in the following aspects.

For the incorporation of both taxonomy and folksonomy for item content analysis, we have not considered the uncertainty of the item-attribute association, which has gained the attention of many researches (Lu et al., 2013; Wu et al., 2014b). We expect to propose an advanced model to integrate fuzzy taxonomy attributes and folksonomy.

For the unique random walk model on a user-item bipartite graph, we currently only focus on the user-item rating relations and the item-item content similarity relations. In the future, we expect to import user-to-user social relations into the bipartite graph and perform new random walks. Thus, we will combine the item content study in Chapter 3 and the user social network analysis in Chapter 4 to develop a more advanced hybrid recommendation model.

For the multi-relational social network-based recommender system, we have emphasized that different user-user relations or correlations need to be constrained for items in a same domain. Taking the Last.fm dataset as an example, all the information we utilized is in-system information, meaning that the friendship, preference similarity and co-tagging networks of users are all related to the topic of music. In the future, we want to build a more general multigraph model that incorporates broader user-user relationships even in different domains. Trust relations on the theme of movies, for example, can then be imported into music recommender systems to further enrich the input data. We expect to address this cross-domain transfer learning problem in a future study.

As mentioned, the multipartite hypergraph model in Chapter 5 currently is not able to handle the tree-structured taxonomy attributes as indicated in Chapter 3 as this goes beyond of the scope of Chapter 5. However, we leave this as a further research topic on hierarchical graph ranking problem, which will also be an interesting topic in both areas of recommender systems and graph ranking.

As one of the advantages, the multi-objective recommender system is able to make recommendations for a group of users, which is closely related to the recent emerging research topic of group recommender systems. In future studies, we expect to import and improve more group negotiation and group decision techniques of existing group recommender systems into our model.

References

Abel, F., Bittencourt, I., Henze, N., Krause, D., Vassileva, J., 2008. A Rule-Based Recommender System for Online Discussion Forums, in: Nejdl, W., Kay, J., Pu, P., Herder, E. (Eds.), Adaptive Hypermedia and Adaptive Web-Based Systems, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 12–21.

Adomavicius, G., Manouselis, N., Kwon, Y., 2011. Multi-criteria recommender systems, in: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.), Recommender Systems Handbook. Springer US, pp. 769–803.

Adomavicius, G., Tuzhilin, A., 2011. Context-aware recommender systems, in: Recommender Systems Handbook. Springer, pp. 217–253.

Adomavicius, G., Tuzhilin, A., 2005. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17, 734–749. doi:10.1109/TKDE.2005.99

Agarwal, A., Chakrabarti, S., 2007. Learning random walks to rank nodes in graphs, in: Proceedings of the 24th International Conference on Machine Learning. ACM, pp. 9–16.

Agarwal, S., 2010. Learning to rank on graphs. Machine learning 81, 333–357.

Agarwal, S., 2006. Ranking on graph data, in: The 23rd International Conference on Machine Learning. ACM, pp. 25–32.

Albadvi, A., Shahbazi, M., 2009. A hybrid recommendation technique based on product category attributes. Expert Systems with Applications 36, 11480–

11488. doi:10.1016/j.eswa.2009.03.046

Al-hassan, M., Lu, H., Lu, J., 2011. Personalized e-government services: tourism recommender system framework, in: Filipe, J., Cordeiro, J. (Eds.), Web Information Systems and Technologies, Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, pp. 173–187.

Al-Shamri, M.Y.H., Bharadwaj, K.K., 2008. Fuzzy-genetic approach to recommender systems based on a novel hybrid user model. Expert systems with applications 35, 1386–1399.

Amatriain, X., Jaimes, A., Oliver, N., Pujol, J.M., 2011. Data mining methods for recommender systems, in: Recommender Systems Handbook. Springer, pp. 39–71.

Armstrong, R., Freitag, D., Joachims, T., Mitchell, T., 1995. Webwatcher: A learning apprentice for the world wide web. Presented at the AAAI Spring symposium on Information gathering from Heterogeneous, distributed environments, pp. 6–12.

Arwan, A., Priyambadha, B., Sarno, R., Sidiq, M., Kristianto, H., 2013. Ontology and semantic matching for diabetic food recommendations, in: The 5th International Conference on Information Technology and Electrical Engineering. Presented at the 2013 International Conference on Information Technology and Electrical Engineering, pp. 170–175. doi:10.1109/ICITEED.2013.6676233

Athreya, K.B., Doss, H., Sethuraman, J., 1996. On the convergence of the Markov chain simulation method. Ann. Statist. 24, 69–100. doi:10.1214/aos/1033066200

Babas, K., Chalkiadakis, G., Tripolitakis, E., 2013. You are what you consume: a bayesian method for personalized recommendations, in: Proceedings of the 7th ACM Conference on Recommender Systems. ACM, pp. 221–228.

Basu, C., Hirsh, H., Cohen, W., 1998. Recommendation as classification: Using social and content-based information in recommendation. Presented at the AAAI/IAAI, pp. 714–720.

Belém, F.M., Martins, E.F., Almeida, J.M., Gonçalves, M.A., 2014. Personalized and object-centered tag recommendation methods for Web 2.0 applications. Information Processing & Management 50, 524–553.

doi:10.1016/j.ipm.2014.03.002

Belkin, M., Matveeva, I., Niyogi, P., 2004. Regularization and semi-supervised learning on large graphs, in: Learning Theory. Springer, pp. 624–638.

Bellogín, A., Cantador, I., Díez, F., Castells, P., Chavarriaga, E., 2013. An empirical comparison of social, collaborative filtering, and hybrid recommenders. ACM Transactions on Intelligent Systems and Technology 4, 14.

Bernardi, R., Chambers, S., Gottfried, B., Segond, F., Zaihrayeu, I. (Eds.), 2011. Advanced Language Technologies for Digital Libraries, Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg.

Biadsy, N., Rokach, L., Shmilovici, A., 2013. Transfer learning for content-based recommender systems using tree matching, in: Cuzzocrea, A., Kittl, C., Simos, D.E., Weippl, E., Xu, L. (Eds.), Availability, Reliability, and Security in Information Systems and HCI, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 387–399.

Biletskiy, Y., Baghi, H., Keleberda, I., Fleming, M., 2009. An adjustable personalization of search and delivery of learning objects to learners. Expert Systems with Applications 36, 9113–9120. doi:10.1016/j.eswa.2008.12.038

Billsus, D., Pazzani, M., 2000. User modeling for adaptive news access. User Modeling and User-Adapted Interaction 10, 147–180. doi:10.1023/A:1026501525781

Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022.

Blomo, J., Ester, M., Field, M., 2013. Recommender system challenge 2013, in: Proceedings of the 7th ACM Conference on Recommender Systems. ACM, pp. 489–490.

Bobadilla, J., Ortega, F., Hernando, A., Alcalá, J., 2011. Improving collaborative filtering recommender system results and performance using genetic algorithms. Knowledge-based systems 24, 1310–1316.

Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, A., 2013. Recommender systems survey. Knowledge-Based Systems 46, 109–132. doi:10.1016/j.knosys.2013.03.012

Bogers, T., 2010. Movie recommendation using random walks over the contextual graph, in: The 2nd International Workshop on Context-Aware Recommender Systems.

Burke, R., 2007. Hybrid web recommender systems, in: Brusilovsky, P., Kobsa, A., Nejdl, W. (Eds.), The Adaptive Web, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 377–408.

Burke, R., 2002. Hybrid recommender systems: Survey and experiments. User modeling and user-adapted interaction 12, 331–370.

Burke, R., 2000. Knowledge-based recommender systems. Encyclopedia of Library and Information Systems 69, 175–186.

Burke, R., 1999. The Wasabi Personal Shopper: a case-based recommender system, in: The 11th National Conference on Innovative Applications of Artificial Intelligence. John Wiley & Sons, pp. 844–849.

Candillier, L., Meyer, F., Fessant, F., 2008. Designing specific weighted similarity measures to improve collaborative filtering systems, in: Advances in Data Mining. Medical Applications, E-Commerce, Marketing, and Theoretical Aspects. Springer, pp. 242–255.

Cantador, I., Brusilovsky, P., Kuflik, T., 2011. Second workshop on information heterogeneity and fusion in recommender systems (HetRec2011)., in: The 5th ACM Conference on Recommender Systems. Chicago, IL, USA, pp. 387–388.

Cao, Y., Li, Y., 2007. An intelligent fuzzy-based recommendation system for consumer electronic products. Expert Systems with Applications 33, 230–240. doi:10.1016/j.eswa.2006.04.012

Capuano, N., Gaeta, M., Ritrovato, P., Salerno, S., 2014. Elicitation of latent learning needs through learning goals recommendation. Computers in Human Behavior 30, 663–673. doi:10.1016/j.chb.2013.07.036

Cheng, H., Tan, P.-N., Sticklen, J., Punch, W.F., 2007. Recommendation via query centered random walk on k-partite graph, in: The 7th IEEE International Conference on Data Mining. IEEE, pp. 457–462.

Chen, Z., Meng, X., Zhu, B., Fowler, R.H., 2000. WebSail: from on-line learning to Web search. Presented at the Proceedings of the First International

Conference on Web Information Systems Engineering, 2000., pp. 206–213 vol.1. doi:10.1109/WISE.2000.882394

Chesnevar, C.I., Maguitman, A.G., 2004. ArgueNet: an argument-based recommender system for solving Web search queries. Presented at the 2nd International IEEE Conference on Intelligent Systems, pp. 282–287 Vol.1. doi:10.1109/is.2004.1344683

Christakou, C., Vrettos, S., Stafylopatis, A., 2007. A hybrid movie recommender system based on neural networks. International Journal on Artificial Intelligence Tools 16, 771–792.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., Sartin, M., 1999. Combining content-based and collaborative filters in an online newspaper, in: Proceedings of ACM SIGIR Workshop on Recommender Systems. Citeseer.

Cobos, C., Rodriguez, O., Rivera, J., Betancourt, J., Mendoza, M., León, E., Herrera-Viedma, E., 2013. A hybrid system of pedagogical pattern recommendations based on singular value decomposition and variable data attributes. Information Processing & Management 49, 607–625. doi:10.1016/j.ipm.2012.12.002

Cornelis, C., Lu, J., Guo, X., Zhang, G., 2007. One-and-only item recommendation with fuzzy logic techniques. Information Sciences 177, 4906–4921.

Creswell, J.W., 2013. Research design: Qualitative, quantitative, and mixed methods approaches. Sage publications.

Deshpande, M., Karypis, G., 2004. Item-based top-N recommendation algorithms. ACM Trans. Inf. Syst. 22, 143–177. doi:10.1145/963770.963776

Dey, A.K., Abowd, G.D., Salber, D., 2001. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. Human–Computer Interaction 16, 97–166. doi:10.1207/S15327051HCI16234_02

Diao, Q., Qiu, M., Wu, C.-Y., Smola, A.J., Jiang, J., Wang, C., 2014. Jointly Modeling Aspects, Ratings and Sentiments for Movie Recommendation (JMARS), in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’14. ACM, New York, NY, USA, pp. 193–202. doi:10.1145/2623330.2623758

Eirinaki, M., Louta, M.D., Varlamis, I., 2014. A trust-aware system for personalized user recommendations in social networks. IEEE Transactions on Systems, Man, and Cybernetics: Systems 44, 409–421. doi:10.1109/TSMC.2013.2263128

Fouss, F., Pirotte, A., Renders, J.-M., Saerens, M., 2007. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Transactions on Knowledge and Data Engineering 19, 355–369.

Gallo, G., Hammer, P.L., Simeone, B., 1980. Quadratic knapsack problems, in: Combinatorial Optimization. Springer, pp. 132–149.

Gallupe, R.B., 2007. The tyranny of methodologies in information systems research 1. ACM SIGMIS Database 38, 20–28.

Garfinkel, R., Gopal, R., Tripathi, A., Yin, F., 2006. Design of a shopbot and recommender system for bundle purchases. Decision Support Systems 42, 1974–1986.

Ge, Y., Xiong, H., Tuzhilin, A., Liu, Q., 2014. Cost-Aware Collaborative Filtering for Travel Tour Recommendations. ACM Trans. Inf. Syst. 32, 4:1–4:31. doi:10.1145/2559169

Ghazanfar, M.A., Prügel-Bennett, A., 2013. Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. Expert Systems with Applications 41, 3261–3275. doi:10.1016/j.eswa.2013.11.010

Gjoka, M., Butts, C.T., Kurant, M., Markopoulou, A., 2011. Multigraph sampling of online social networks. IEEE Journal on Selected Areas in Communications 29, 1893–1905. doi:10.1109/JSAC.2011.111012

Golbeck, J., 2006. Combining provenance with trust in social networks for semantic web content filtering, in: Provenance and Annotation of Data. Springer, pp. 101–108.

Golbeck, J.A., 2005. Computing and applying trust in web-based social networks.

Golbeck, J., Hendler, J., 2006. Filmtrust: Movie recommendations using trust in web-based social networks, in: Proceedings of the IEEE Consumer Communications and Networking Conference. University of Maryland, pp. 282–286.

Goldberg, D., Nichols, D., Oki, B.M., Terry, D., 1992. Using collaborative filtering to weave an information tapestry. Commun. ACM 35, 61–70. doi:10.1145/138859.138867

Goldberg, K., Roeder, T., Gupta, D., Perkins, C., 2001. Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval 4, 133–151. doi:10.1023/a:1011419012209

Gori, M., Pucci, A., 2007. ItemRank: A random-walk based scoring algorithm for recommender engines, in: The 20th International Joint Conference on Artifical Intelligence, IJCAI'07. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 2766–2771.

Gorla, J., Lathia, N., Robertson, S., Wang, J., 2013. Probabilistic group recommendation via information matching, in: The 22nd International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, pp. 495–504.

Herlocker, J., Konstan, J.A., Riedl, J., 2002. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Information Retrieval 5, 287–310.

Hsu, S.H., Wen, M.-H., Lin, H.-C., Lee, C.-C., Lee, C.-H., 2007. AIMED-A personalized TV recommendation system, in: Interactive TV: A Shared Experience. Springer, pp. 166–174.

Hwang, C.-S., Chen, Y.-P., 2007. Using trust in collaborative filtering recommendation, in: New Trends in Applied Artificial Intelligence. Springer, pp. 1052–1060.

Jacob, Y., Denoyer, L., Gallinari, P., 2011. Classification and annotation in social corpora using multiple relations, in: Proceedings of the 20th ACM International Conference on Information and Knowledge Management. ACM, pp. 1215–1220.

Jalali, M., Mustapha, N., Sulaiman, M.N., Mamat, A., 2010. WebPUM: A Web-based recommendation system to predict user future movements. Expert Systems with Applications 37, 6201–6212. doi:10.1016/j.eswa.2010.02.105

Jamali, M., Ester, M., 2009. Trustwalker: A random walk model for combining trust-based and item-based recommendation, in: The 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 397–406.

Jameson, A., Smyth, B., 2007. Recommendation to groups, in: The Adaptive Web. Springer, pp. 596–627.

Jannach, D., Karakaya, Z., Gedikli, F., 2012. Accuracy Improvements for Multi-criteria Recommender Systems, in: Proceedings of the 13th ACM Conference on Electronic Commerce, EC ’12. ACM, New York, NY, USA, pp. 674–689. doi:10.1145/2229012.2229065

Jiang, M., Cui, P., Wang, F., Yang, Q., Zhu, W., Yang, S., 2012. Social recommendation across multiple relational domains, in: Proceedings of the 21st ACM International Conference on Information and Knowledge Management. ACM, pp. 1422–1431.

Katakis, I., Tsapatsoulis, N., Mendez, F., Triga, V., Djouvas, C., 2014. Social voting advice applications—definitions, challenges, datasets and evaluation. IEEE Transactions on Cybernetics 44, 1039–1052.

Kazienko, P., Musial, K., Kajdanowicz, T., 2011. Multidimensional social network in the social recommender system. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 41, 746–759.

Kim, K., Ahn, H., 2008. A recommender system using GA K-means clustering in an online shopping market. Expert systems with applications 34, 1200–1209.

Konstas, I., Stathopoulos, V., Jose, J.M., 2009. On social networks and collaborative recommendation, in: The 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 195–202.

Lampropoulos, A.S., Lampropoulou, P.S., Tsihrintzis, G.A., 2012. A cascade-hybrid music recommender system for mobile services based on musical genre classification and personality diagnosis. Multimedia Tools and Applications 59, 241–258.

Lee, S., Song, S., Kahng, M., Lee, D., Lee, S., 2011. Random walk based entity ranking on graph for multidimensional recommendation, in: The 5th ACM Conference on Recommender Systems, RecSys ’11. ACM, Chicago, IL, USA, pp. 93–100. doi:10.1145/2043932.2043952

Leung, W.C., 2009. Enriching user and item profiles for collaborative filtering: from concept hierarchies to user-generated reviews. The Hong Kong Polytechnic

University.

Liang, H., Xu, Y., Li, Y., Nayak, R., 2010. Personalized recommender system based on item taxonomy and folksonomy, in: The 19th ACM International Conference on Information and Knowledge Management, CIKM ’10. ACM, New York, NY, USA, pp. 1641–1644. doi:10.1145/1871437.1871693

Li, Q., Wang, J., Chen, Y.P., Lin, Z., 2010. User comments for news recommendation in forum-based social media. Information Sciences 180, 4929–4939.

Lopes, G.R., Moro, M.M., Wives, L.K., De Oliveira, J.P.M., 2010. Collaboration recommendation on academic social networks, in: Advances in Conceptual Modeling–Applications and Challenges. Springer, pp. 190–199.

Lucas, J.P., Luz, N., Moreno, M.N., Anacleto, R., Figueiredo, A.A., Martins, C., 2013. A hybrid recommendation approach for a tourism system. Expert Systems with Applications 40, 3532–3550.

Lu, J., 2004. A personalized e-learning material recommender system, in: Proceedings of the 2nd International Conference on Information Technology and Applications.

Lu, J., Guo, X., Huynh, A.T., Benkovich, L., 2004. SRS: A Subject Recommender System to Enhance E-learning Personalisation.

Lu, J., Shambour, Q., Xu, Y., Lin, Q., Zhang, G., 2013. A web-based personalized business partner recommendation system using fuzzy semantic techniques. Computational Intelligence 29, 37–69. doi:10.1111/j.1467-8640.2012.00427.x

Lu, J., Shambour, Q., Xu, Y., Lin, Q., Zhang, G., 2010. BizSeeker: a hybrid semantic recommendation system for personalized government-to-business e-services. Internet Research 20, 342–365.

Lu, J., Wu, D., Mao, M., Wang, W., Zhang, G., 2015. Recommender system application developments: A survey. Decision Support Systems 74, 12–32. doi:10.1016/j.dss.2015.03.008

Ma, H., Yang, H., Lyu, M.R., King, I., 2008. Sorec: Social recommendation using probabilistic matrix factorization, in: Proceedings of the 17th ACM Conference on Information and Knowledge Management. ACM, pp. 931–

940.

Mao, K., Fan, J., Shou, L., Chen, G., Kankanhalli, M., 2014. Song recommendation for social singing community, in: Proceedings of the ACM International Conference on Multimedia. ACM, pp. 127–136.

Marceau, V., Noël, P.-A., Hébert-Dufresne, L., Allard, A., Dubé, L.J., 2011. Modeling the dynamical interaction between epidemics on overlay networks. Physical Review E 84, 026105.

Massa, P., Avesani, P., 2009. Trust metrics in recommender systems, in: Computing with Social Trust. Springer, pp. 259–285.

Massa, P., Avesani, P., 2007. Trust-aware recommender systems, in: The 2007 ACM Conference on Recommender Systems. ACM, pp. 17–24.

Masthoff, J., 2011. Group recommender systems: Combining individual models, in: Recommender Systems Handbook. Springer, pp. 677–702.

McAuley, J., Leskovec, J., 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text, in: The 7th ACM Conference on Recommender Systems, RecSys '13. ACM, New York, NY, USA, pp. 165–172. doi:10.1145/2507157.2507163

McCarthy, K., Reilly, J., McGinty, L., Smyth, B., 2004. Thinking positively-explanatory feedback for conversational recommender systems. Presented at the Proceedings of the European Conference on Case-Based Reasoning (ECCBR-04) Explanation Workshop, pp. 115–124.

Melville, P., Mooney, R.J., Nagarajan, R., 2002. Content-boosted collaborative filtering for improved recommendations, in: AAAI/IAAI. Presented at the 14th Innovative Applications of Artificial Intelligence Conference, pp. 187–192.

Middleton, S.E., De Roure, D.C., Shadbolt, N.R., 2002. Foxtrot recommender system: user profiling, ontologies and the World Wide Web. Presented at the The Eleventh International World Wide Web Conference (WWW2002).

Middleton, S., Roure, D., Shadbolt, N., 2009. Ontology-based recommender systems, in: Staab, S., Studer, R. (Eds.), Handbook on Ontologies, International Handbooks on Information Systems. Springer Berlin Heidelberg, pp. 779–796.

Milgram, S., 1967. The small world problem. Psychology Today 2, 60–67.

Mobasher, B., Cooley, R., Srivastava, J., 2000. Automatic personalization based on Web usage mining. Commun. ACM 43, 142–151. doi:10.1145/345124.345169

Mooney, R.J., Roy, L., 2000. Content-based book recommending using learning for text categorization, in: Proceedings of the Fifth ACM Conference on Digital Libraries. ACM, pp. 195–204.

Moreno, A., Valls, A., Isern, D., Marin, L., Borràs, J., 2013. Sigtur/e-destination: ontology-based personalized recommendation of tourism and leisure activities. Engineering Applications of Artificial Intelligence 26, 633–651.

Nanopoulos, A., 2011. Item recommendation in collaborative tagging systems. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 41, 760–771. doi:10.1109/TSMCA.2011.2132708

Nguyen, T., Lu, H., Lu, J., 2013. Web-page recommendation based on web usage and domain knowledge. IEEE Transactions on Knowledge and Data Engineering PP, 1041–4347. doi:10.1109/tkde.2013.78

O'Donovan, J., Smyth, B., 2005. Trust in recommender systems, in: Proceedings of the 10th International Conference on Intelligent User Interfaces. ACM, 1040870, pp. 167–174. doi:10.1145/1040830.1040870

O'Sullivan, D., Smyth, B., Wilson, D., 2004. Preserving recommender accuracy and diversity in sparse datasets. International Journal on Artificial Intelligence Tools 13, 219–235.

Page, L., Brin, S., Motwani, R., Winograd, T., 1999. The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

Pashtan, A., Blattler, R., Andi, A.H., Scheuermann, P., 2003. CATIS: a context-aware tourist information system. Presented at the The 4th International Workshop of Mobile Computing.

Pazzani, M., 1999. A framework for collaborative, content-Based and demographic filtering. Artificial Intelligence Review 13, 393–408. doi:10.1023/a:1006544522159

Pazzani, M., Billsus, D., 2007. Content-based recommendation systems, in:

Brusilovsky, P., Kobsa, A., Nejdl, W. (Eds.), The Adaptive Web, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 325–341.

Porcel, C., López-Herrera, A.G., Herrera-Viedma, E., 2009. A recommender system for research resources based on fuzzy linguistic modeling. Expert Systems with Applications 36, 5173–5183.

Quijano-Sánchez, L., Bridge, D., Díaz-Agudo, B., Recio-García, J.A., 2012. A case-based solution to the cold-start problem in group recommenders, in: Case-Based Reasoning Research and Development. Springer, pp. 342–356.

Rae, A., Sigurbjörnsson, B., van Zwol, R., 2010. Improving tag recommendation using social networks, in: The 2010 Adaptivity, Personalization and Fusion of Heterogeneous Information, RIAO '10. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, Paris, France, France, pp. 92–99.

Rafailidis, D., Daras, P., 2013. The TFC model: Tensor factorization and tag clustering for item recommendation in social tagging systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems 43, 673–688. doi:10.1109/TSMCA.2012.2208186

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J., 1994. GroupLens: An open architecture for collaborative filtering of netnews, in: The 1994 ACM Conference on Computer Supported Cooperative Work. pp. 175–186. doi:10.1145/192844.192905

Resnick, P., Varian, H.R., 1997. Recommender systems. Commun. ACM 40, 56–58. doi:10.1145/245108.245121

Ruiz-Montiel, M., Aldana-Montes, J.F., 2009. Semantically Enhanced Recommender Systems, in: Meersman, R., Herrero, P., Dillon, T. (Eds.), On the Move to Meaningful Internet Systems: OTM 2009 Workshops, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 604–609.

Salehi, M., Sharma, R., Marzolla, M., Magnani, M., Siyari, P., Montesi, D., 2014. Spreading processes in Multilayer Networks. arXiv:1405.4329 [physics].

Salton, G., McGill, M.J., 1986. Introduction to modern information retrieval.

Santos, O.C., Boticario, J.G., Pérez-Marín, D., 2014. Extending web-based educational systems with personalised support through user centred designed recommendations along the e-learning life cycle. Science of Computer Programming 88, 92–109. doi:10.1016/j.scico.2013.12.004

Sarwar, B., Karypis, G., Konstan, J., Riedl, J., 2001. Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web. ACM, pp. 285–295.

Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S., 2007. Collaborative filtering recommender systems, in: Brusilovsky, P., Kobsa, A., Nejdl, W. (Eds.), The Adaptive Web, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 291–324.

Shambour, Q., 2012. Hybrid Recommender Systems for Personalized Government-to-Business e-Services. University of Technology, Sydney.

Shambour, Q., Lu, J., 2012. A trust-semantic fusion-based recommendation approach for e-business applications. Decision Support Systems 54, 768–780.

Shambour, Q., Lu, J., 2011. A hybrid multi-criteria semantic-enhanced collaborative filtering approach for personalized recommendations, in: The 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01, WI-IAT ’11. IEEE Computer Society, Washington, DC, USA, pp. 71–78. doi:10.1109/WI-IAT.2011.109

Shardanand, U., Maes, P., 1995. Social information filtering: Algorithms for automating “word of mouth,” in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM Press/Addison-Wesley Publishing Co., pp. 210–217.

Shinde, S.K., Kulkarni, U., 2012. Hybrid personalized recommender system using centering-bunching based clustering algorithm. Expert Systems with Applications 39, 1381–1387.

Shiratsuchi, K., Yoshii, S., Furukawa, M., 2006. Finding unknown interests utilizing the wisdom of crowds in a social bookmark service, in: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology. IEEE Computer Society, pp. 421–424.

Shi, Y., Larson, M., Hanjalic, A., 2014. Collaborative filtering beyond the user-item matrix: a survey of the state of the art and future challenges. ACM Computing Surveys 47, 3:1–3:45. doi:10.1145/2556270

Smyth, B., 2007. Case-based recommendation, in: Brusilovsky, P., Kobsa, A., Nejdl, W. (Eds.), The Adaptive Web, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 342–376.

Smyth, B., Cotter, P., 2000. A personalized television listings service. Communications of the ACM 43, 107–111. doi:10.1145/345124.345161

Stabb, S., Werther, H., Ricci, F., Zipf, A., Gretzel, U., Fesenmaier, D.R., Paris, C., Knoblock, C., 2002. Intelligent systems for tourism. IEEE Intelligent Systems 17, 53–66. doi:10.1109/MIS.2002.1134362

Tan, S., Bu, J., Chen, C., Xu, B., Wang, C., He, X., 2011. Using rich social media information for music recommendation via hypergraph model. ACM Transactions on Multimedia Computing, Communications and Applications 7S, 22:1–22:22. doi:10.1145/2037676.2037679

Theodoridis, A., Kotropoulos, C., Panagakis, Y., 2013. Music recommendation using hypergraphs and group sparsity, in: The 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 56–60. doi:10.1109/ICASSP.2013.6637608

Tong, H., Faloutsos, C., Pan, J.-Y., 2006. Fast random walk with restart and its applications, in: The Sixth International Conference on Data Mining, ICDM ’06. IEEE Computer Society, Washington, DC, USA, pp. 613–622. doi:10.1109/ICDM.2006.70

Vaishnavi, V.K., Kuechler, W., 2015. Design science research methods and patterns: innovating information and communication technology. CRC Press.

Verbert, K., Manouselis, N., Ochoa, X., Wolpers, M., Drachsler, H., Bosnic, I., Duval, E., 2012. Context-aware recommender systems for learning: a survey and future challenges. IEEE Transactions on Learning Technologies 5, 318–335. doi:10.1109/TLT.2012.11

Woerndl, W., Brocco, M., Eigner, R., 2009. Context-aware recommender systems in mobile scenarios. International Journal of Information Technology and Web Engineering 4, 67–85. doi:10.4018/jitwe.2009010105

Wu, D., Zhang, G., Lu, J., 2014a. A fuzzy tree matching-based personalised e-learning recommender system, in: 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). Presented at the 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1898–1904.

doi:10.1109/FUZZ-IEEE.2014.6891594

Wu, D., Zhang, G., Lu, J., 2014b. A fuzzy preference tree-based recommender system for personalized business-to-business e-services. IEEE Transactions on Fuzzy Systems 22, 1–1. doi:10.1109/TFUZZ.2014.2315655

Xue, G.-R., Lin, C., Yang, Q., Xi, W., Zeng, H.-J., Yu, Y., Chen, Z., 2005. Scalable collaborative filtering using cluster-based smoothing, in: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 114–121.

Yager, R.R., 2003. Fuzzy logic methods in recommender systems. Fuzzy Sets and Systems 136, 133–149.

Yang, X., Steck, H., Guo, Y., Liu, Y., 2012. On top-k recommendation using social networks, in: The 6th ACM Conference on Recommender Systems, RecSys '12. ACM, New York, NY, USA, pp. 67–74. doi:10.1145/2365952.2365969

Yildirim, H., Krishnamoorthy, M.S., 2008. A random walk method for alleviating the sparsity problem in collaborative filtering, in: The 2nd ACM Conference on Recommender Systems. ACM, pp. 131–138.

Yin, R.K., 2013. Case study research: Design and methods. Sage publications.

Yuan, W., Shu, L., Chao, H.-C., Guan, D., Lee, Y.-K., Lee, S., 2010. ITARS: Trust-aware recommender system using implicit trust networks. IET Communications 4, 1709–1721.

Yu, K., Tresp, V., Yu, S., 2004. A nonparametric hierarchical bayesian framework for information filtering, in: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 353–360.

Zenebe, A., Norcio, A.F., 2009. Representation, similarity measures and aggregation methods using fuzzy sets for content-based recommender systems. Fuzzy Sets and Systems 160, 76–94.

Zhang, Z., Lin, H., Liu, K., Wu, D., Zhang, G., Lu, J., 2013. A hybrid fuzzy-based personalized recommender system for telecom products/services. Information Sciences 235, 117–129.

Zhao, D., Li, L., Peng, H., Luo, Q., Yang, Y., 2014. Multiple routes transmitted epidemics on multiplex networks. Physics Letters A 378, 770–776. doi:10.1016/j.physleta.2014.01.014

Zhen, Y., Li, W.-J., Yeung, D.-Y., 2009. TagiCoFi: Tag informed collaborative filtering, in: Proceedings of the Third ACM Conference on Recommender Systems. ACM, pp. 69–76.

Zhou, D., Huang, J., Schölkopf, B., 2006. Learning with hypergraphs: Clustering, classification, and embedding, in: The 2006 Advances in Neural Information Processing Systems. pp. 1601–1608.

Zhou, D., Schölkopf, B., 2004. A regularization framework for learning from graph data, in: Workshop on Statistical Relational Learning at the 21st International Conference on Machine Learning. Canada.

Zhu, T., Harrington, P., Li, J., Tang, L., 2014. Bundle recommendation in Ecommerce, in: Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR '14. ACM, New York, NY, USA, pp. 657–666. doi:10.1145/2600428.2609603