Jumping Connections: A Graph-Theoretic Model for Recommender Systems

Batul J. Mirza

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

February 8, 2001
Blacksburg, Virginia

Keywords: Recommender Systems, Collaborative Filtering, Random Graphs

Jumping Connections: A Graph-Theoretic Model for Recommender Systems

Batul J. Mirza

ABSTRACT

Recommender systems have become paramount to customize information access and reduce information overload. They serve multiple uses, ranging from suggesting products and artifacts (to consumers), to bringing people together by the connections induced by (similar) reactions to products and services. This thesis presents a graph-theoretic model that casts recommendation as a process of 'jumping connections' in a graph. In addition to emphasizing the social network aspect, this viewpoint provides a novel evaluation criterion for recommender systems. Algorithms for recommender systems are distinguished not in terms of predicted ratings of services/artifacts, but in terms of the combinations of people and artifacts that they bring together. We present an algorithmic framework drawn from random graph theory and outline an analysis for one particular form of jump called a 'hammock.' Experimental results on two datasets collected over the Internet demonstrate the validity of this approach.

Acknowledgments

I would like to thank my advisor, Dr. Naren Ramakrishnan, for not only giving me the opportunity to conduct research in the field of recommender systems, but also for his extreme patience with me during the entire process. I sincerely thank him for his assistance, guidance, and valuable motivation.

I would also like to thank other people who have played a significant part in bringing about this thesis. Dr. Ben Keller provided me with numerous suggestions and ideas that helped me accomplish my work. Dr. Lenwood Heath identified important shortcomings of random graph models that are expressed in Section 3.4. I thank Dr. Calvin Ribbens and Dr. Srinidhi Varadarajan who helped provide me with access to the Cluster Computing Lab, a division of the Laboratory for Advanced Scientific Computing and Applications (LASCA). I also thank the computer science department at Virginia Tech for providing the services I used to conduct my experiments. Particular mention must be made of Alex Verstak who went out of his way to help me complete the final phase of my experiments. Acknowledgements are also due to the Compaq Equipment Corporation, which provided the EachMovie dataset and the University of Minnesota, which provided the MovieLens dataset used in my experiments.

Special thanks to my parents, sister, and brother for their love, encouragement, and prayers. Finally, I would like to thank my husband, Farooq, without whose invaluable help, constant support, endless patience, and understanding, my thesis would have been unachievable.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recommender systems [RV97] constitute one of the fastest growing segments of the Internet economy today. They help reduce information overload and provide customized information access for targeted domains. Building and deploying recommender systems has thus matured into a fertile business activity, with benefits in retaining customers and enhancing revenues. Elements of the recommender landscape include search engines, handcrafted content indices, personalized shopping agents on e-commerce sites, and news-on-demand services. The scope of such 'personalization' thus extends to many different forms of information content and delivery, not just web pages. The underlying algorithms and techniques, in turn, range from simple keyword matching of consumer profiles, 'collaborative filtering,' to more sophisticated forms of data mining, such as clustering web server logs.

1.1 Problem Outline

Historically, recommender systems have been studied in domains that focus on harnessing distributed information resources, information aggregation, social schemes for decision making, and user interfaces. Indicative of this multidisciplinary foci, current research on recommender systems can be sliced in several different ways. The four main dichotomies are: (i) how is a recommender system modeled? (i.e., how are recommendations actually made?), (ii) how is a recommender system targeted (i.e., at what level is the information tailored?), (iii) how is a recommender system built (is it a 'public transportation' system or a 'hot-rod'? [Ram00, RS97]), (iv) how is a recommender system maintained (online versus offline designs).

Such dichotomies typically reflect the philosophies of the underlying domains, the proposers, and their parent communities (we will cover one such dichotomy in detail in Chapter 2). A recurring theme among many of these viewpoints is that recommendation is implicitly cast as a task of learning mappings (from people to recommended artifacts, for example) or of filling in entries to missing cells in a matrix (of consumer preferences, for example). Such simplifications miss many important aspects of a recommender system such as (i) its role as an indirect way of bringing people together, (ii) its 'signature pattern' of making 'connections,' (iii) the explainability of its recommendations, and (iv) the mathematical models of the social networks in which the recommendations are delivered. In parallel, the undue emphasis on the predictive capability of a recommender system (of future ratings) has limited the exploration of novel evaluation criteria for recommender system algorithms.

This thesis proposes a graph-theoretic model of a recommender system as a process of 'jumping connections' in a bipartite graph of people versus artifacts. This model is inspired by all of the above limitations and, in addition, provides a novel approach to experimental algorithmics. Algorithms for recommendation can be characterized by (i) the kinds of connections they 'jump,' and (ii) the number of entities they serve to bring together by such 'jumps.' We will make these notions formal in Chapter 3 but an intuitive example of a jump can be had from real life.

Assume that Linus would like to get a recommendation for a certain movie M. One way to obtain a rating for this item would be to find somebody (say, Lucy) who has indeed seen this movie *and* who could serve as the *rating provider*. For example, perhaps Linus and Lucy are known to have similar tastes and preferences. A recommender system would then assume that Linus can be *reached from* Lucy in a single 'jump'; this would then serve as the bridge by which Lucy's rating of M could be transformed into a recommendation for Linus.

It is our thesis that all recommender systems obey this model and differ in only the nature of their jumps. We will show how seemingly dissimilar recommender algorithms in the literature are really special cases of this general setting. By casting recommendation as a process of making connections (perhaps in sophisticated ways), algorithms can be viewed as mechanisms for bringing people (and movies) together (into a connected component). Thus, two algorithms can be distinguished based on the jumps they posit and the kinds (and number) of nodes that are brought together. From the viewpoint of a recommender system designer, the jumps should be as 'expressive' as possible and should bring together as many cohesive subgroups of nodes as possible. How these two conflicting goals can be consolidated is another contribution of this thesis; we develop an algorithmic framework in which to conduct such studies.

1.2 Reader's Guide

Chapter 2 surveys current research and motivates the need for a new approach to analyzing algorithms for recommender systems. Chapter 3 introduces the 'jumping connections' framework and develops a mathematical model based on random graph theory. Chapter 4 provides experimental results for one particular way of 'jumping' on two application datasets. Issues related to interpretation of results from our model are also presented here. Finally, Chapter 5 identifies various opportunities for future research.

Chapter 2

Background

The increase in the amount of information available through online resources (such as web pages and email) has greatly exacerbated the problem of information overload. This has fueled an extensive ongoing research effort in recommender systems, which select subsets of information to be presented from a typically larger set of possibilities. Recommender systems can be defined as systems that help reduce information overload by filtering out information which may be otherwise inapplicable to an individual or a group of individuals. They are realized by algorithms that model preferences (of people) and use such models to predict ratings (of products, services) or provide recommendations.

A typical recommender system could answer questions such as:

- What movie should I go see this weekend?
- Which web pages are most related to my query?
- Which is the most popular restaurant in Washington D.C.?
- Which people have similar reading interests in medieval art history?

Typically, the most common way to obtain recommendations of artifacts is by 'word of mouth,' i.e., to solicit them from friends with similar tastes, or to depend on articles about the artifacts written by well-known critics or experts. Such strategies are mirrored in the current landscape of recommender systems. `Amazon.com` uses ratings and profiles to precisely tailor the content provided at its web pages (for example, "Since you liked *Sense and Sensibility*, you might also be interested in *Pride and Prejudice* too"). 'Expert sites' such as `epinions.com` provide a conduit for critics and judges to voice their reactions. In the general case, people provide ratings and evaluations as inputs to an automated system, which then aggregates and directs them to appropriate recipients [RV97]. Recommender systems thus help bring people together by modeling commonalities of interests.

Table 2.1: Categorization of recommender systems applications.

Provide Recommendations for	*Provide Recommendations of*	*Example Applications and Systems*
People	(Other) People	Match-Making Services (e.g. Yenta [Fon96])
People	Artifacts	E-commerce Applications, Recreation (e.g., Jester [Gol])
Artifacts	People	Identifying Experts (e.g. Referral Web [Kau])
Artifacts	(Other) Artifacts	Feature-Based Retrieval (e.g. Art Paintings Selection [Koh])

Most recommender systems are web-based since data abounds in the Internet environment. In fact, operating recommender systems offline is more challenging due to the associated problems of data gathering and collection in a 'bricks and mortar' setting. Furthermore, the web has become the preferred source of information (over resources such as libraries, encyclopedia, and televisions). Data such as the keywords of a person's query, click-stream data (actions of the user during browsing, i.e. the links (s)he follows and the amount of time spent on each page), purchase data, assessments provided for products in the form of ratings, and bookmarks could be useful in gleaning information about the person's interests and background. For example, the Siteseer system [RP97] uses the event of saving a web page into a bookmark folder as a measure of user preference. Systems such as WebWatcher [AFJM95] use link navigation information to gather information about user interests. Recommender systems exploit such data from the web and automatically provide recommendations for entities such as movies, books, web pages, or restaurants [Tri]. Besides the above mentioned forms of data, e-commerce sites track dynamic information indicating how *recently* the customer bought a product from the site, how *frequently* the customer purchases, and how much the e-commerce site profits from the customer's purchases (*monetary data*). Sites that exploit these three categories of information are said to employ *RFM* data [DR99].

A broad categorization of recommender systems applications is provided in Table 2.1. The input descriptions involve features (of artifacts), preferences, and experiences (of people). The output recommendations involve identifying specific artifacts or particular users. This categorization is intended to be neither exhaustive nor mutually exclusive. Applications involving recommendations of artifacts to people will be used to motivate the ideas presented in this thesis.

2.1 E-Commerce Applications

Recommender systems benefit both customers as well as sellers in e-commerce applications. They minimize the amount of information that customers need to process before they find the artifact they are looking for. For the seller, recommender systems aid in:

1. *Improving cross-sell*: Cross-selling implies offering additional related products to a customer when he/she shows interest in purchasing a product [SKR99, DR99]. "Did you know we have a hat and glove set that matches this jacket for only $69.95?" is an example of cross-selling. At sites such as `amazon.com`, `barnesandnoble.com`, and `cdnow.com`, if a customer selects any product X, the site also recommends a set of other related products that were purchased by customers who bought product X in the past.

2. *Improving up-sell*: Up-selling means recommending a similar but more expensive product package to the customer. An example of up-selling is offering a suggestion such as "Would you like to purchase the limited edition version of the accompanying software for just $5.95 extra?" As another example, if the customer is looking for 32 MB RAM, offer him 64 MB at a discount.

3. *Turning browsers into buyers*: Recommender systems help e-commerce systems to hold the attention of new and occasional visitors at specific site(s) [SKR99]. Typical information, such as best sellers and recommendations by experts, can be provided for such customers to engage their attention. Personalized email notification on arrival of products desired by customers is another commonly used method of inviting customers back to the e-commerce site.

4. *Providing targeted assortments of products*: Recommender systems can break the generic content mold and offer personal service based on the depth of knowledge and understanding of the tastes and preferences of each customer [DR99]. For example, a fan of Sheryl Crow would be presented with a new release by this artist or artists that other customers with similar tastes have rated highly, while a Latin music fan would receive an entirely different experience with recommendations for the new Ricky Martin or Enrique Iglesias CDs.

5. *Providing personalized ads*: Related to point 4 above is the technique of tailoring web advertisements to the individual customer. Recommender systems enable sites, by tracking generalized patterns of customer behavior, to make good guesses over time about the kinds of ads a customer would welcome, indirectly enhancing possibilities of additional profits.

   The above two aspects are sometimes jointly referred to as *personalization*. Personalization is the process of targeting web content by customizing information access to meet a user's needs and preferences. Its scope is broader than e-commerce, as discussed in section 2.5.

6. *Providing gift recommendations*: `Amazon.com`, Yahoo! Shopping, `Theman.com`, `Mondera.com` are examples of e-commerce sites that have incorporated 'gift recommenders' to aid customers in finding a perfect gift for a third party [DR99]. These sites conduct extensive modeling of the recipient to match profiles and preferences.

7. *Enhancing loyalty of customers*: The concept of loyalty here refers to making customers revisit the e-commerce site in an effort to retain their business [SKR99]. Personalization leads to loyalty; if a customer uses an e-commerce site that remembers enough contextual information about her, she is more likely to revisit the site in the future. This also has implications for the privacy of the users of recommender systems.

2.2 Recommender Systems Dichotomies

Various dichotomies have been proposed that classify recommender systems according to the philosophies of the underlying domains, the proposers, and their parent communities. A commonly accepted classification distinguishes between content-based and collaborative recommender systems [AT99, ATar, RV97].

2.2.1 Content-Based Design

Content-based recommender systems model the content (features) of an artifact and recommend artifacts by querying a database of artifact features against the preferences of the user [KB96]. They have been applied in several domains, from recommending movies [AKK98] and books [MR00] to recommending web sites [PMB96]. Content-based recommendation has its roots in database modeling [FLM98] and information filtering [CBS92].

The simplest form of such information filtering is through *keyword matching*. One of the earliest information filtering systems was called 'selective dissemination of information' (SDI) [HK70]. SDI was designed as an automatic way of alerting scientists of new documents published in their areas of interest. Users can create a profile of keywords that describes their fields of interest; when new publications matching a profile appear, the system informs the user instantly.

Belkin and Croft [BC92] describe information filtering systems to be designed for unstructured or semistructured data, where filtering is based on user profiles, representing the user's long-term interests. They also show that information filtering and information retrieval (IR) are almost identical at an abstract level. IR, in general, is the process of searching for and extracting specific information from amongst a collection of information items [SM83]. IR uses queries as specifications of information needs, and is concerned with single uses of the system whereas information filtering employs user profiles to satisfy long-term goals of the users. While retrieval denotes 'finding' the most relevant data, filtering carries a connotation of 'removing' data that is irrelevant for the user. Software such as Letizia [Lie95] works in concert with browsers, inferring user preferences from their browsing behavior, using a strategy that is midway between information filtering and information retrieval.

Since content-based filtering involves parsing (for descriptive features), it works best with text-based documents, and has found limited success in domains such as sound, images, and video. In keyword matching, the context of the search phrase has to be taken into account, as well. Polysemy (the situation when words have multiple meanings) and synonymy (when different words have the same meaning) of words further increase the difficulty of identifying relevant selections. Dimensionality reduction techniques such as 'latent semantic indexing' (LSI) have been empirically shown [Fol90] to overcome such problems.

2.2.2 Collaborative Filtering Systems

While information filtering depends only on an individual's preferences, collaborative filtering considers preferences of other (similar) people. (In information filtering, an individual only 'collaborates with himself' to gradually improve his profile [Tur].) First introduced by Goldberg et al. [GNOT92], collaborative filtering today forms the core underlying technology for many recommender systems [AWWY99]. It involves:

- Accumulating preferences (or profiles) of people and recording their behavior (e.g. for a movie recommender system, explicit ratings given by people for different movies can be recorded)
- Selecting a number of 'neighbors' for each person. Neighbors of a person are other people whose ratings/experiences are similar to this person's.
- Based on the behavior of these neighbors, predicting future behavior of the person (e.g. the system would recommend the movie 'Erin Brochovich' to a person, because many of his neighbors gave high ratings to this movie).

Collaborative filtering can be done using either explicit or implicit ratings. Explicit ratings are directly provided by users to declare preferences, while implicit ratings are gleaned (often secretively) by observing a person's behavior, such as browsing patterns. Collaborative filtering techniques have also been used for unconventional applications, such as in the PYTHIA recommender system for selecting solvers for partial differential equations (PDEs) [Ram99]. Algorithms are recommended for newly presented problems based on their performances on 'similar' problem instances [RG99a].

Table 2.2: Input matrices for content-based design (left) and for collaborative design (right).

	Feature 1	$\cdots$	Feature n
Artifact 1	f_{11}	$\cdots$	f_{1n}
$\cdots$	$\cdots$	$\cdots$	$\cdots$
Artifact m	f_{m1}	$\cdots$	f_{mn}

	Artifact 1	$\cdots$	Artifact m
Person 1	r_{11}	$\cdots$	r_{1m}
$\cdots$	$\cdots$	$\cdots$	$\cdots$
Person k	r_{k1}	$\cdots$	r_{km}

There are various endemic problems with collaborative filtering:

- *Cold Start:* The cold start problem has also been called the *day one* problem [ME95], due to the fact that on the first day of its service, the system has no available data to begin to make recommendations. Collaborative filtering also gives poor performance for new items, i.e., items that have not yet been rated by any person. As the number of people gets larger, the number of multidimensional comparisons over all combinations seriously affects the scalability as well.

- *Banana Problem* [Bur99]: Recommender systems that exploit associations among customer purchases are particularly sensitive to the frequency of ratings for specific items. For example, since bananas are frequently purchased by customers in many grocery stores, a recommender system using market basket data to infer preferences will always recommend bananas. A converse problem is that of recommending a product that is bought much less frequently and only one at a time (e.g., automobiles) [Bur99]. Similarly, there might be a user with unusual preferences for whom there will not be any similar users and which could lead to poor recommendations.

- *Effusivity and Subjectivity in Ratings:* Ratings of similar users are typically combined to make recommendations in collaborative filtering, but users often use entirely different ranges of ratings to express identical preferences. One user may be more effusive in his ratings than another, although they rate objects in the same order. Techniques that preserve relative and ordinal sequences are often used to identify significant variations in rating patterns [FISS98, AWWY99].

2.2.3 Hybrid Systems

A consequence of both content-based and collaborative filtering is that one ends up with recommendations that are very similar to what was recommended previously, since the preferences are based on profiles of a person or obtained from people with similar ratings. The input matrices for content-based and collaborative designs are shown in Table 2.2. In content-based designs, each artifact is described as a vector of feature values (numeric or symbolic). For collaborative designs, ratings of artifacts are provided by each person. Both designs do not provide for *serendipitous recommendations*, in the way of 'new finds.' In social life, we do not seek to interact with only those who are similar to ourselves. Rather, we will often enjoy the recommendations given by people who are different from ourselves, in exploring a new area of interest.

Collaborative filtering is useful when there are enough other users in the system with overlapping preferences. Content-based filtering is appropriate when a new user or a new object is added to the system. Notice that the features employed by content-based filtering are typically orthogonal, whereas the profiles (of people) captured by collaborative filtering might have redundancy. This is one of the primary reasons dimensionality reduction techniques have found some success in collaborative filtering.

Hybrid recommender systems bring the advantages of different recommendation technologies together in order to improve accuracy of the predicted ratings. Approaches that harness both these approaches include the GroupLens 'filterbot agents' [SKB$^+$98], the Research Assistant Agent Project [DIT98], and the CLEVER search engine [Cen] that uses link analysis to model collaborative recommendations. Attempts at a hybrid movie recommender system [BHC98] showed an improved performance (in terms of accuracy of retrieval of relevant selections) over pure content and collaborative approaches.

In practice, hybrid systems which provide both group as well as individual recommendations, are most prevalent. Such systems employ user profiles built by content analysis to make good recommendations even if there are no similar users and they can also filter out objects the user may have disliked in the past. At the same time, by making collaborative recommendations, they can combine other users' experiences using content-based filtering.

2.3 Review of Some Specific Projects

This section describes five recommender system algorithms. GroupLens and LikeMinds are well-established algorithms and power some web sites operating publicly. Firefly was purchased by the Microsoft Corporation in 1998 and has been out of the news since then. The fourth algorithm presented here is relatively new, published in August 1999. The above four are all collaborative filtering algorithms. The last algorithm discussed here is a hybrid recommender system — Fab.

GroupLens

The GroupLens system [RIS$^+$94] was first used in a world wide trial for Usenet (online collection of newsgroups) in 1996 where a total of 250 volunteers rated over $20,000$ news articles. Miller et al. [MRK97] provide a detailed summary of the results of the trial. The large number of users and news postings in Usenet provide a rich source of data and a challenge for real-time implementation. Collaborative filtering in an Usenet application is different from issues in other domains such as movies or music because the item (news articles) volume is much larger and their lifetime is very short [KMM+97].

The basic idea was to allow readers to rate articles using special servers called 'better bit bureaus' (BBBs). BBBs measured correlation coefficients, using a Pearson algorithm to describe the extent of agreement between two users on their ratings. Ratings are then predicted for a particular user i of a new (unrated) artifact (article) j by computing a weighted average of all ratings for article j from users who are in agreement with i. While the Usenet research trial is now over, the GroupLens idea is now applied in MovieLens, which recommends movies at the publicly operating site `http://movielens.umn.edu`.

Firefly

Evolving from Ringo [SM95] and HOMR (Helpful Online Music Recomendation Service), Firefly lets a web site make intelligent recommendations about books, music, or movies, that users might like depending on their stated tastes. Firefly's algorithm is now used on sites like `BarnesandNoble.com`. The idea is very similar to the GroupLens algorithm, except a weighted average is calculated of only the ratings of those users whose coefficients are greater than some threshold [Sha94]. Ringo had become available to the public in 1994 and after its huge success, was subsequently disbanded, having been acquired by Microsoft.

LikeMinds

Macromedia LikeMinds goal was to offer fast, accurate personalization in real-time [Lik]. The LikeMinds personalization server is a web-based system that accumulates a database of consumer product preferences, then uses simple L_2 and Mahalanobis distances to measure 'closeness' between users (in terms of ratings) [Gre]. A demonstration site called Movie Critic asks users to initially rate 12 movies to receive recommendations (accessible at `http://www.moviecritic.com`).

'Intelligent Recommendation Algorithm'

Aggarwal et al. [AWWY99] illustrate an interesting graph-based approach to collaborative filtering, used in the Intelligent Recommendation Algorithm (IRA) project at IBM Research. The paper also describes experiments that were conducted to compare the accuracy, speed and scalability of the LikeMinds, Firefly and the IRA algorithms using synthetic data.

The algorithm extends the basic ideas presented above in several important directions. First, it addresses sparsity of ratings by not requiring a direct link (commonality of ratings) between the consumer (of recommendations) and the supplier (who has rated the artifact of interest).

Instead, the bipartite graph of users and artifacts is traversed to determine an appropriate user, whose ratings are then propagated back (via a sequence of nodes) to the consumer. To address effusivity of ratings, such propagation is allowed to introduce linear transformations of ratings to achieve matching of individual values.

The twin notions of *horting* and *predictability* [AWWY99] effectively provide these extensions. A user X is said to *hort* user Y if there are 'enough' items that X and Y have rated in common. This commonality constraint is tunable either as an absolute measure or as a fraction of the items rated by X (in the latter case, horting would not be a symmetric relation). Predictability is then defined in a reverse manner: Y *predicts* X if X horts Y and there exists a linear transformation (within some tunable bounds) that can map Y's ratings to (predictions for) X.

Thus, IRA is different from approaches such as GroupLens, Firefly and LikeMinds in which predictions are computed by taking a weighted average of the ratings of *only immediate neighbors*. In IRA, the opinions of the users who have not rated the item in question are also considered while producing predictions, thus making the notion of predictability more general than 'closeness.'

Fab

Fab [BS97], a part of the Stanford University digital library project, is an agent-based hybrid filtering system for recommending web pages. User profiles are maintained using content-based filtering and comparison of these profiles to determine similarity between users is done collaboratively. Fab consists of three main components: *collection* agents, *selection* agents and a central router. The collection agents find and send web pages based on the 'current topic' profiles, which they maintain, to the central router. The router forwards these pages to the selection agents, which use a particular user's profile to select web pages to present to him. The user is then required to rate the page and his ratings are used to update the selection agent's user profiles and the collection agent's topic profiles.

2.4 Other Technologies

This section describes research in related areas; namely data mining, clustering, link analysis, and small-world networks.

2.4.1 Data Mining

Data mining, sometimes referred to as knowledge discovery in databases (KDD), has been defined as 'the nontrivial extraction of implicit, previously unknown, and potentially useful information from data' [CCH91]. Organizations can use such information for competitive business advantage, e.g. for e-commerce, user profiling, and web personalization. Goals common to all data-mining applications are the detection, interpretation, and prediction of qualitative or quantitative patterns in data [RG99a]. Data mining techniques [Han96], including association rule mining and data clustering, have had significant impact on the design of recommender systems. For example, association rule mining has been used to make recommendations by exploring association between people, associations between objects, and between the two [Lin00]. It can also be used to develop top-N recommendations of products [SKKR00a]. Similarly, clustering methods can be used to group 'similar' people into clusters in order to provide typical recommendations [UF98, KM99], described in detail below.

2.4.2 Clustering

Several clustering techniques including K-means clustering have been used in conjunction with collaborative filtering [UF98]. In most cases, clustering has been viewed as an approach to combat the problems of new users and new objects in the system [Lee00]. The K-means clustering technique was first described by MacQueen [Mac67], and uses a centroid-based distance metric to assign nodes (users as well as artifacts) to clusters. Breese et al. [BHK98] describe the use of bayesian classification in collaborative filtering, which has been used in the Do-I-Care collaborative web agent [ASP97]. Bayesian classification assumes that for every classifiable object, there exists a set of attributes that allow the computation of probabilities that the object belongs to each of a known set of clusters. METIS [Kar] and kMETIS [KK95], developed at the Univ. Minnesota, use multilevel graph partitioning and have been explored in recommender systems research.

Clustering that reveals genres and rating patterns appears to be the immediate benefit of this mode of analysis in recommender systems [Lep]. However, some studies show that the quality of prediction is not significantly improved by clustering [Lep]. Results from experiments conducted by O'Conner and Herlocker [OH99], show that although clustering greatly increases scalability (distances to only neighbors within a cluster need be computed for making recommendations), it does not improve accuracy of the recommendations. However, their results show that generating clusters using the kMETIS graph partitioning algorithm produces predictions which are much more accurate than those generated by random clustering or by clustering based on movie genres (in the case of a movie recommender system).

2.4.3 Link Analysis

Link analysis begins with data that can be modeled as a network and attempts to infer useful knowledge from the nodes and links of the network. Data is represented as a graph, with nodes representing entities in the domain, and edges representing relationships between the entities. Link analysis has been used to extract information in many areas such as in web search engines [Kle98], in exploration of associations among criminals [Ric98], and in the field of medicine [SS98]. One of the earliest uses of link analysis in recommender systems is described in [SW93].

ReferralWeb [KSS97], a recommender system applying ‘referral chaining,’ a form of link analysis, uses the co-occurrence of names in any of the documents available on the web, to detect the existence of direct relationships between people and thus indirectly form social networks. The underlying assumption is that people with similar interests swarm in the same circles to discover *collaborators* [Pay98].

Jon Klienberg [Kle98] uses his HITS (Hyperlink-Induced Topic Search) link analysis algorithm, to introduce the notion of two types of web sites: *hubs* and *authorities*. A good hub links to many authorities and a good authority will be linked to by many hubs. Starting with a specific search query, HITS performs a text-based search to seed an initial set of results. An iterative relaxation algorithm then assings hub and authority weights using a matrix power iteration. Empirical results show that remarkably authoritative results are obtained for search queries. The CLEVER search engine [CDRK$^+$99] is built primarily on top of the basic algorithm described above. The Google search engine [BP98b] also makes use of the web’s link structure in addition to the anchor text, though in a limited way. The offline characteristic of Google, as opposed to the topic-induced search of CLEVER, is one of the main reasons for the commercial success of the former. Use of link analysis thus promises improvement in the performance of recommender systems.

2.4.4 Small-World Networks

The exploration of link analysis in social structures has led to several new avenues of research, most notably ‘small-world networks.’ Small-world networks refer to highly clustered but relatively sparse networks with small average length. The reader may be familiar with the folklore notion of six degrees of separation separating any two people in our universe i.e., the phenomenon where a person can discover a link to any other random person through a chain of atmost six acquaintances. A small-world network is sufficiently clustered so that most second neighbors of a node X are also neighbors of X (a typical ratio would be 80%). On the other hand, the average distance between any two nodes in the graph is comparable to the low characteristic path length of a random graph. Until recently, a mathematical characterization of such small-world networks has proven elusive. Watts and Strogatz, in their 1998 pioneering work [WS98] described a graph generation model for small-world networks.

They use a regular ‘wreath’ network with n nodes, and k edges per node (to its nearest neighbors) as a starting point for the design. A small fraction of the edges are then randomly *rewired* to arbitrary points on the network. A full rewiring (probability $p = 1$) leads to a completely random graph, while $p = 0$ corresponds to the (original) wreath as shown in Figure 2.1. The starting point in the figure is a regular wreath topology of 12 nodes with

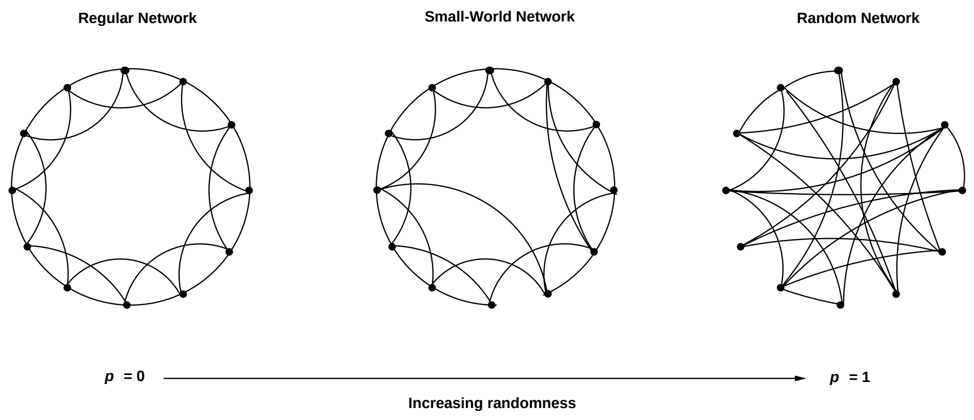

Figure 2.1: Generation of a small-world network by random rewiring from a regular wreath network. Figure adapted from [WS98].

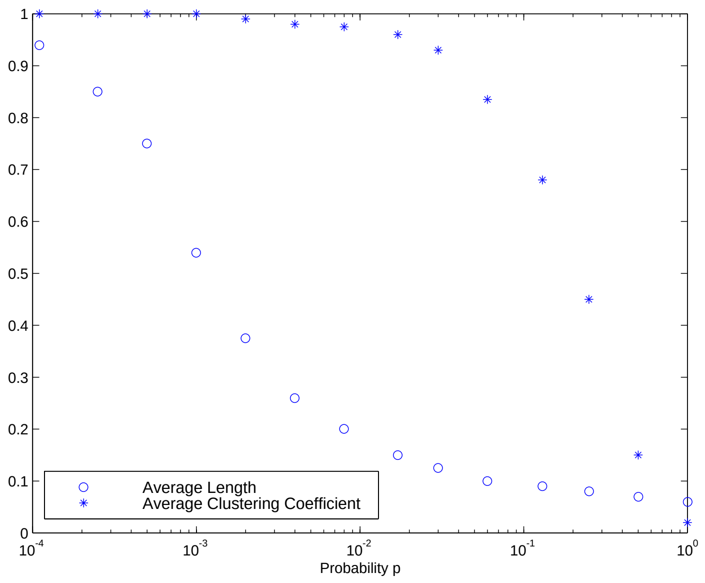

Figure 2.2: Average path length and clustering coefficient versus the rewiring probability p (from [WS98]). All measurements are scaled w.r.t. the values at $p = 0$.

every node connected to its four nearest neighbors. This structure has a high characteristic path length ($O(n)$) and high clustering coefficient. The average length is the mean of the shortest path lengths over all pairs of nodes. The clustering coefficient is determined by first computing the local neighborhood of every node. The number of edges in this neighborhood as a fraction of the total possible number of edges denotes the 'cliquishness' of this neighborhood. This factor is averaged over all nodes to determine the clustering coefficient. The other extreme is a random network with a low characteristic path length and almost no clustering. The small-world network, an interpolation between the two, has the low characteristic path length (of a random network), and retains the high clustering coefficient (of the wreath). Measuring properties such as average length and clustering coefficient in the region $0 \leq p \leq 1$ produces surprising results (see Fig. 2.2).

As shown in Fig. 2.2, only a very small fraction of edges need to be rewired to bring the length down to random graph limits, and yet the clustering coefficient is high. On closer inspection, it is easy to see why this should be true. Even for small values of p (e.g., 0.1), the result of introducing edges between distantly separated nodes reduces not only the distance between these nodes but also the distances between the neighbors of those nodes, and so on (these reduced paths between distant nodes are officially known as *shortcuts*). The introduction of these edges further leads to a rapid decrease in the average length of the network, but the clustering coefficient remains almost unchanged. Thus, small-world networks fall in between regular and random networks, having the small average lengths of random networks but high clustering coefficients akin to regular networks.

While the Watts-Strogatz model describes how small-world networks can be formed, it does not explain how people are adept at actually finding short paths through such networks in a decentralized fashion. Kleinberg in [Kle99] addresses precisely this issue and proves that this is not possible in the family of one-dimensional Watts-Strogatz networks (the wreaths above, formally *rings*). Embedding the notion of random rewiring in a two-dimensional lattice leads to one unique model for which such decentralization is effective.

The small-world network concept has implications for a variety of domains. Watts and Strogatz [WS98] simulate the 'wildfire' like spread of an infectious disease in a small-world network. Adamic [Ada99] shows that the world wide web is a small-world network too and suggests that search engines capable of exploiting this fact can be more effective in hyperlink modeling, crawling, and finding authoritative sources.

2.5 Putting it All Together: Personalization

Consolidation of many of the above ideas underlies the creation of personalization solutions on the Internet. One of the problems with web sites is that they tend to be too feature-rich. A user may probably use only a small fraction the functionality of an application on the web. If however, the entire set of users is considered, then there is definitely someone who needs each of these features. Software that configures itself to individual needs entails the scope of personalization. Personalization refers to the automatic adjustment of information content, structure, and presentation to individual users. It can be described as any action that makes the web experience of a user personalized to the user's taste [MCS00].

Recommender systems provide personalization by resolving the conflict between having too much information and a clean interface. Well developed personalization reminds a user about important dates like birthdays of friends and family and recommends gifts according to their age and profile. It can improve the success of the web site and increase the return on marketing investment by providing one-to-one marketing [PR97]. Recommender systems have been classified as non-personalized, ephemeral, and persistent depending on the degree of personalization provided [SKRar].

Personalization is not limited to e-commerce. It has been used to personalize e-mails and online newspapers. For example, Anatagonomy [KSK97], a personalized newspaper on the web, learns user profiles using both implicit and explicit ratings to create personalized newspaper pages. Depending on which news articles have high scores, noticeable and large screen spaces are allotted to them. Similarly, an engine to generate TV guides as HTML web pages personalized for the viewing of individual users has also been proposed [SC00]. Other sites such as Yahoo! provide a My Yahoo! page [MPR00] to personalize its content and layout. CNN's `mycnn.com` allows users to tailor newsfeeds by specifying criteria on sources and topics. Anupam et al. [ABFK99] give a proposal to build a personalization service that allows users to create personalized pages that integrate information from more than one web site. Ramakrishnan [Ram00] describes a programmatic framework to design a personalization system around a specific collection of web sites.

2.6 Limitations of Existing Approaches

A main limitation of current research efforts in recommender systems is that they do not provide systematic mechanisms to evaluate the efficiency and effectiveness of different recommender system algorithms. Traditional information retrieval evaluation metrics such as *precision* and *recall* are not suited here since satisfaction with a product/service cannot be modeled functionally without a human element. In an attempt to shy away from the HCI aspects of recommender systems research, existing evaluation metrics treat recommendation as a classification or function approximation problem.

The paper by Breese et al. [BHK98], one of the most widely cited in the recommender systems literature, is typical. The authors present various algorithms for recommender systems and (later) apply the standard cross-validation procedure of dividing the original dataset into two parts — a *training* set and a *test* set. The training set serves as the collaborative filtering database and is used to predict the ratings of the users in the test set. The predictions are then compared to the original test set data to obtain absolute errors in predictions of ratings.

Thus, recommendation often is cast as a task of function approximation or learning mappings [AWWY99, BP98a, BHC98, GRGP00, GSK+99, HKBR99, HSRF95, KFV00, KMM+97, PHLG00, SKKR00b, SM95, SN99, SKKR00a, SKR99, THA+97]. Even approaches that focus on clustering (e.g. [UF98]) view it primarily as a pre-processing step for functional modeling or as a technique to ensure scalability. Such simplifications miss many desirable properties of the recommendation process, namely:

- **The social view of a recommender system as an indirect way of bringing people together.** Social network theory [WF94] refers to a recommendation system of people versus artifacts as an *affiliation network* and distinguishes between a *primary mode* (e.g., people) and a *secondary mode* (e.g., movies), where a *mode* refers to a 'distinct set of entities on which structural variables are measured [WF94].' The purpose of the secondary mode is viewed as serving to bring entities of the primary mode together (i.e., it isn't treated as a *first-class* mode).

- **Emphasis on the connections used in making the recommendations, not just on the (predicted) rating for the artifact.** In many situations, users would like to request recommendations *purely* based on local and global constraints on the nature of the specific connections explored. Functional modeling techniques are inadequate since they embed the task of learning a mapping from people to predicted values of artifacts in a general-purpose learning system such as neural networks and bayesian classification [BHK98]. A notable exception is the work by Hofmann and Puzicha [HP99] that concentrates on functional modeling using a restricted set of 'connection structures.' Closely related to this aspect is

- **The ability of the recommender system to explain the choices and selections made for the user** (in terms and constructs that are natural to the user/application domain). It is nearly impossible to convince the user of the quality of a recommendation obtained by black-box techniques such as neural networks.

- **The mathematical models of social networks in which recommendations are usually delivered.** In a recommender system, the rating patterns of people on artifacts induce an implicit social network and influence the connectivities in this network. Little study has been done to understand how such rating patterns influence recommendations and how they can be advantageously exploited.

This thesis proceeds to address these issues in the next chapters.

Chapter 3

A Model of Jumping Connections

To address the issues identified in the previous chapter, we develop a novel way to characterize algorithms for recommender systems. Algorithms will be distinguished, not in terms of (predicted) ratings of services/artifacts, but in terms of the combinations of people and artifacts that they bring together. If two algorithms work in qualitatively different ways but, for any given input graph, bring together exactly the same set of nodes, they are considered equivalent. This distinction emphasizes the central role of a recommender system as a mechanism for bridging entities in a social network. We refer to this approach of studying algorithms as 'jumping connections' (JC). Notice that the JC framework does not emphasize how the recommendation is actually made and effected (e.g., is the rating squashed? linearly transformed?). Almost all prior research in recommender systems has focused on the prediction aspect, with implicit assumptions of connection-jumping in the design of the system. By restricting its scope to exclude the actual aspect of making ratings and predictions, the JC framework provides a systematic and rigorous way to study recommender systems.

Of course, the choice of 'how to jump connections' will be driven by the (often conflicting) desire to reach almost every node in the graph (i.e., recommend every product for somebody, recommend some product for everybody) and the strength of the jumps enjoyed when two nodes are brought together. It should be emphasized that JC does not imply that algorithms only exploit local structure of the social network. Any mechanism — local or global — could be used to jump connections.

Notice also that when an algorithm 'brings together' person X and artifact Y, it could imply either a positive recommendation or a negative one. Such differences are, again, not captured by our framework unless the mechanism for making connections restricts its jumps, for instance, to only those artifacts for which ratings satisfy some threshold. In other words, thresholds for making recommendations could be abstracted into the mechanism for jumping.

Table 3.1: Summary of the symbols used in the JC model.

Symbol	*Meaning*
$\mathcal{R}$	Recommender dataset
P	Set of all people in $\mathcal{R}$
N_P	Number of people in $\mathcal{R}$
M	Set of all movies in $\mathcal{R}$
N_M	Number of movies in $\mathcal{R}$
E	Set of all edges in $\mathcal{R}$
$A_{\mathcal{R}}$	Adjacency matrix of $\mathcal{R}$
$\mathcal{J}$	Jump function
G_s	Social network graph
A_{Gs}	Adjacency matrix of G_s
G_r	Recommender graph
A_{Gr}	Adjacency matrix of G_r
w	Hammock width
κ	Minimum rating constraint
l_{pp}	Shortest path length in G_s
l_r	Shortest path length in G_r
l_{pm}	Shortest path length from people to movies in G_r

JC satisfies all four issues outlined in the previous chapter. It is based on the social-network model, and thus, emphasizes connections rather than prediction. The nature of connections jumped also aids in explaining the recommendations. The graph-theoretic nature of JC allows the inclusion of mathematical models (e.g., random graphs) of social networks in which recommender systems are designed.

3.1 The JC Construction

Before developing the model of jumping connections, we define some terminology and notation that will be useful. Furthermore, we specifically develop our theory using a movie recommender system as an application domain. This choice was primarily driven by the availability of public-domain datasets for validation purposes. It does not restrict the range of applicability of JC and is introduced here only for ease of presentation. Table. 3.1 summarizes the various symbols and notations that will be used in this thesis.

Definition 3.1 A *recommender dataset* $\mathcal{R}$ is a bipartite graph $G(P \cup M, E)$ where P is the set of all people, M is the set of all movies, and E contains edges between elements of P and elements of M (Fig. 3.1). Recall the definition of a mode as a 'distinct set of entities

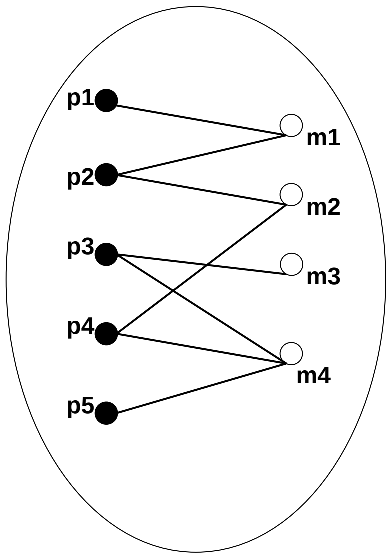

Figure 3.1: A bipartite graph of people and movies.

on which structural variables are measured [WF94].' Thus, $\mathcal{R}$ can be viewed as a two-mode (people and movies) network. E could indicate ratings, viewings, preferences, experiences, or constraints on movie recommendations. Assuming $|P| = N_P$, $|M| = N_M$, the bipartite graph of people and movies represented by $\mathcal{R}$ is then given by the $(N_P + N_M) \times (N_P + N_M)$ adjacency matrix:

$$A_{\mathcal{R}} = \left[\begin{array}{c|c} 0 & \mathcal{R} \\ \hline \mathcal{R}^T & 0 \end{array} \right]$$

In this thesis, we assume that the entries in $A_{\mathcal{R}}$ are either 0 or 1, though in the general case, they could be arbitrary numeric values corresponding to quantitative and/or qualitative information about ratings and preferences.

One can view M as a secondary mode in the bipartite graph or affiliation network and attempt to understand the influence of M on the connectivity among members in P. We thus arrive at our second definition.

Definition 3.2 A *jump* is a function $\mathcal{J} : \mathcal{R} \mapsto S, S \subseteq P \times P$ that takes as input a recommender dataset $\mathcal{R}$ and returns a set of (unordered) pairs of elements of P. Intuitively, this means that the two nodes described in a given pair can be reached from one another, by a single jump. Notice that this definition does not prescribe how the mapping should be performed, or whether it should use all the information present in $\mathcal{R}$. We also make the simplifying assumption that jumps can be composed in the following sense: if node B can be reached from A in one jump, and C can be reached from B in one jump, then C is reachable from A in two jumps.

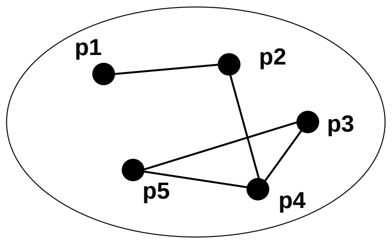

Figure 3.2: Social network graph for the recommender dataset shown in Fig. 3.1

Since the only edges in $\mathcal{R}$ go between two modes (people and movies), the simplest jump is the *skip*, which connects two members in P if they have at least one movie in common. We refer to graphs induced in this manner (by using various jump functions) as *social network graphs*. Fig. 3.2 shows the social recommender graph induced from the example in Fig. 3.1 using a skip jump.

Definition 3.3 The social network graph of a recommender dataset $\mathcal{R}$ induced by a given jump $\mathcal{J}$ is a unipartite undirected graph $G_s(P, E_s)$, where the edges are given by $E_s = \mathcal{J}(\mathcal{R})$. Notice that the induced graph could be disconnected based on the strictness of the jump function. The adjacency matrix A_{Gs} corresponding to G_s is of size $N_P \times N_P$ with entries from $\{0, 1\}$. In addition, since G_s is undirected:

$$A_{Gs}{}^T = A_{Gs}$$

The *recommender system* viewpoint is to qualify the paths (and path lengths) needed to make a recommendation of a particular movie for a particular person. To model this, we view the unipartite social network of people as a directed graph and reattach movies (seen by each person) such that every movie is a sink (reinforcing its role as a secondary mode). Shortest path algorithms through this graph can then be used to provide the basis for recommendations. We refer to a graph induced in this fashion as a *recommender graph* (Fig. 3.3). Since the outdegree of every movie node is fixed at zero, paths through the graph run from people to movies (through more people, if necessary).

Definition 3.4 The recommender graph of a recommender dataset $\mathcal{R}$ induced by a given jump function $\mathcal{J}$ is a directed graph $G_r(P \cup M, E_{sd} \cup E_{md})$, where E_{sd} is an ordered set of pairs, listing every pair from E_s (of the social network graph induced by $\mathcal{J}$ on $\mathcal{R}$) in both directions, and E_{md} is an ordered set of pairs, listing every pair from E in the direction pointing to the movie mode. The $(N_P + N_M) \times (N_P + N_M)$ adjacency matrix for G_r provided by this construction is thus:

$$A_{Gr} = \left[\begin{array}{c|c} A_{Gs} & \mathcal{R} \\ \hline 0 & 0 \end{array}\right]$$

Figure 3.4 illustrates the process of generating the social network and recommender graphs for our example recommender dataset using the skip jump function.

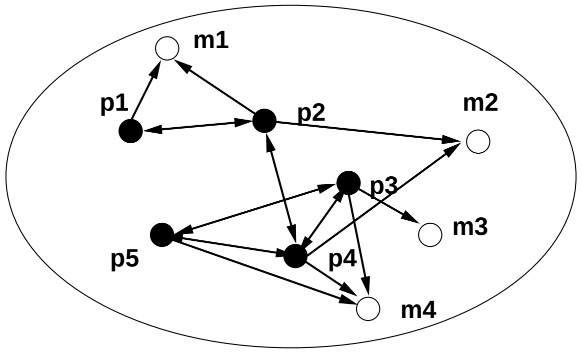

Figure 3.3: Recommender graph obtained by rendering the social network graph with bidirectional edges and reattaching the movies.

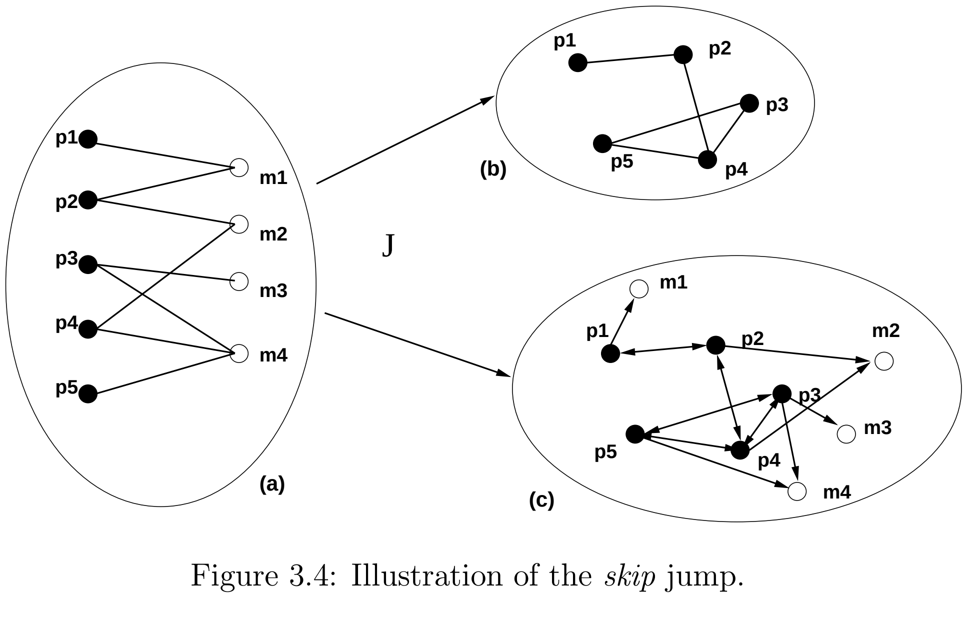

Figure 3.4: Illustration of the *skip* jump.

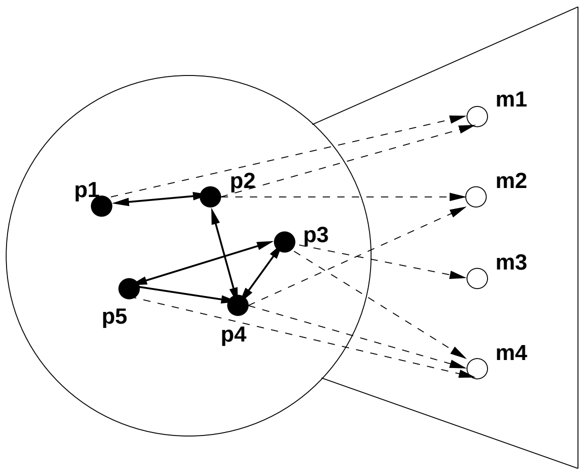

Figure 3.5: The JC construction produces a half bow-tie graph G_r.

Assuming that the jump construction does not cause G_r to be disconnected, the portion of G_r containing only people is its strongest component: every person is connected to every other person. The movies constitute vertices which can be reached from the strongest component, but from which it is not possible to reach the strongest component (or any other node, for that matter). Thus, G_r can be viewed as a 'half bow-tie,' (see Fig. 3.5) as contrasted to the full bow-tie nature of the web [BKM+99]. The circular portion in the figure depicts the strongly connected directed component derived from G_s. Links out of this portion of the graph (people) go to sinks (movies).

3.2 The Many Ways of Jumping

For a given bipartite graph, there are many ways of inducing the social network graph and the recommender graph. The simplest and most common has been illustrated in Fig. 3.4. Note that this idea now provides a systematic way to characterize recommender systems algorithms in the literature. In this section, we do just that — we outline the nature of connections used by different algorithms to induce the two secondary graphs. In turn, this will help us understand the kinds of nodes (and graph subparts) that will be brought together.

3.2.1 Two-Step Linear Paths

Variations of the skip can be observed in the LikeMinds [Lik], GroupLens [KMM+97], and Firefly [SM95] algorithms. All of these algorithms assume there is an edge between two people (in the social network graph) based on the extent of agreement between the ratings of the people on the common movies. GroupLens and Firefly use correlation coefficients

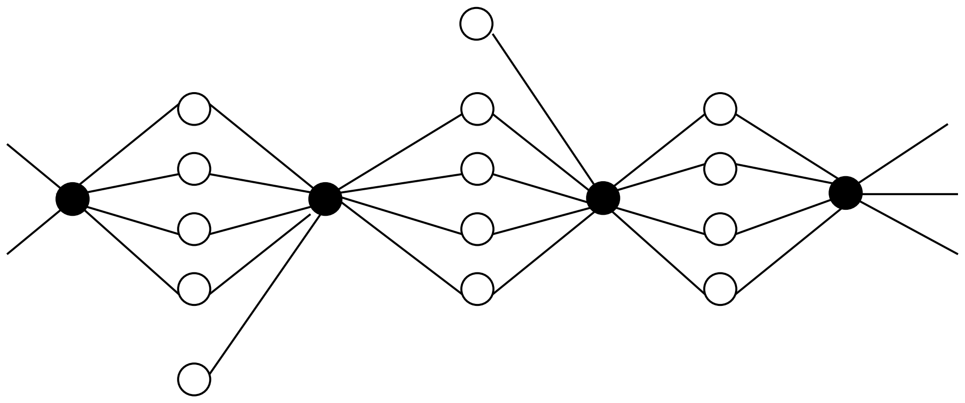

Figure 3.6: Illustration of a sequence of *hammock* jumps, with a *hammock width* $w = 4$.

and LikeMinds uses vector-norms, reminiscent of early information retrieval research. In the recommender graph, all three algorithms require a path (from a person to a movie) to not involve more than one intermediate person, hence the phrase *two-step*. In addition, two people could be brought together if they have even one artifact in common, thus forming a *linear path* between the two people. Notice that there are other differences between the algorithms based on how exactly a rating is computed for a given movie, but the JC framework does not distinguish between these aspects.

3.2.2 Hammocks

The *hammock* jump [Kel00] brings two people together in G_s if they have at least w (called *hammock width*) artifacts in common in $\mathcal{R}$. Formally, a pair (p_1, p_2) is in $\mathcal{J}(\mathcal{R})$ iff the $(p_1, p_2)^{th}$ entry of $A_{\mathcal{R}} \times A_{\mathcal{R}} \geq w$. Thus a hammock jump with $w = 1$ corresponds to the skip jump. The idea of 'horting' [AWWY99] can be viewed as employing the hammock jump. First, it posits that two people are connected in the social network graph, only if they have seen w movies together (see Fig. 3.6) and have similar ratings (achieved by characterizing a linear transformation) for these w movies. Thus in implementations of this jump, only those sequences of hammocks are allowed, whose ratings satisfy an agreement criterion. In addition, this criterion is defined in a undirectional manner, so even the social network graph is directed (the jump function, in this case, would produce *ordered* pairs of nodes from P). Paths in the social network graph would, thus, correspond to 'sequences of hammocks' in the original bipartite graph. In the recommender graph, no constraints are imposed on the number of intermediate nodes needed to reach a movie from a particular person. This is one of the ways the technique addresses sparsity of connections. In one implementation of this algorithm in the IBM Recommendation Engine, a limit on the maximum length is imposed, assuming that a majority of the nodes are reachable within this constraint.

3.2.3 Attribute-Value Based Techniques

Attribute-value techniques such as neural networks and bayesian classification have been used in recommender systems [BHK98]. The underlying theme is that recommendation is cast as a learning problem. These approaches conduct no direct modeling relevant to the social network graph. The recommender graphs induced consist only of a functional mapping from people to movies. It is thus difficult to characterize the kinds of connections these algorithms jump, since they are heavily dependent on the original topology of the network. For example, if people in certain age groups rate certain movies in a similar manner, a neural network can be expected to capture this pattern (given sufficient data), and hence will connect two people if they fall in the same age group. In other words, inducing connections for the social network graph is implicit (and captured by the similarity of activations for certain inputs), but making recommendations (using a recommender graph) is explicit in the neural network topology.

3.2.4 Sites Based on Social Network Navigation

Although not directly relevant to recommender systems, we wish to point out an analogy of JC to a particular design involving web pages. A surprising number of web sites base their design on a social metaphor of navigating links through a multi-mode network to identify information. The Internet Movie Database at `http://www.imdb.com` while providing basic search facilities, models the network connecting actors, actress, movies, directors, songs etc. Users are able to systematically jump connections to find answers to queries such as 'Which was the movie that first introduced the lead actor in Titanic?' Another well known example is the DBLP bibliography web site (`http://www.informatik.uni-trier.de/~ley/db/`) that models the network of authors, papers, major computer science journals, and conferences. These can be subsumed in the JC framework by including *types* of links in the bipartite graph and restricting jumps to concentrate on predefined combinations involving types.

3.2.5 Exploiting Global Structure

All of the above algorithms jump connections based on local properties, such as commonality of ratings and agreement of ratings among *neighboring* nodes. No effort is made to jump connections by taking into account the global nature of connections. Techniques such as Latent Semantic Indexing [Fol90] achieve precisely this effect, by rendering the bipartite graph as a matrix, performing a principal component analysis, and dropping lower order terms. This practice is typically justified as either (i) removing noise, (ii) reducing the effective dimension (rank) to make computations tractable, or (iii) addressing information retrieval problems such as synonymy. Nevertheless, two people who have not seen any movie in common could be brought together by being clustered into the same principal component.

The exact nature of jumps is difficult to characterize since this practice is sensitive to the actual distribution of principal components and the number of terms dropped. A very recent study [JL00] promises to shed some light in this area, by arriving at a general formulation of matrix computations in information retrieval.

3.2.6 Probabilistic Dyadic Models

Finally, we look at the model proposed by Hofmann, et al. [HP99] which is interesting for two reasons. First, there is no explicit modeling of a social network graph, akin to other attribute-value techniques discussed above. To make jumps, the authors use an intermediate mode (besides people and movies) which is intended to capture a *latent variable* that has an effect on ratings and recommendations. A recommendation of a movie for a person is made if an expectation-maximization (EM) applied to this three-mode graph implies a strong dependence between the two nodes. The second point is that it allows the designer to explicitly incorporate prior information about the latent variables in the EM algorithm. This means that different portions of the graph can support different forms of jumps. In this sense, this is a unique model.

3.2.7 Generality of the JC Model

It is worth investigating if JC covers all possible designs for recommender systems. At first glance, content-based designs look like they cannot be subsumed under this model; but realize that one can model the features (of artifacts) as a new mode and connect the presence/absence of features to the existence of edges in the starting graph (recommender dataset $\mathcal{R}$). Thus, information filtering can also be abstracted in this framework. This brings us to the issue of how one can design new algorithms with this abstraction. The common aspect among all the above algorithms is best expressed in terms of an *anti-condition*: 'Algorithms in JC never bring together nodes from disconnected components of $\mathcal{R}$.' We state this without proof, but it is easy to see why it is true in algorithms such as GroupLens and Horting. Proving this property for algorithms such as LSI is beyond the scope of this thesis, but could help in the acceptance of this model.

3.3 Approach

The most immediate implication of jumping for recommender systems is the reduction in the length of the path used to route a recommendation. In the 1980s, this problem of reducing the number of intermediaries was referred to as *information routing*[1]. In this thesis, we restrict

[1]Information filtering branched off this trend [BC92], with its emphasis on removing information.

our attention to the (undirected) hammock jump. Recall that the hammock jump models the 'strength' of recommendations, i.e., if more people agree with you, the recommendation you get through them should be stronger. Other motivations for selecting the hammock derive from the experimental context in which recommender systems are usually designed. Typically, people are convinced to rate at least κ objects, in order to avail of a recommendation service. In many commercial sites and systems, the value of κ is hardwired and they enforce this constraint strictly. The reason is economic viability — people are notorious for *free-riding* on recommendations [AZ97], without providing any input/feedback to the system. Free-riding is a serious issue for the feasibility of a recommender system. Several options have been studied to address it; providing incentives, hiring full-time raters, subscription sites, and membership revocation are popular commercial solutions. The hammock is the most natural jump to investigate the effect of κ on recommender system algorithms, since connectivity is based on commonality of ratings. We thus arrive at the following working question:

> (*) Given a recommender dataset $\mathcal{R}$, what is the relationship between the hammock width w and the average length of the path used to route recommendations?

Intuitively, this problem helps us connect the metrics of 'how much data to collect' (κ, which in turn affects the possible hammock width w) and 'how small a length is desired' (for routing recommendations). We address this problem by adopting random graph models of the jump-induced graphs G_s and G_r and using generating functions to calculate properties such as the average path length. This provides a systematic and rigorous basis for characterizing recommender system algorithms.

Our formal mathematical framework is adapted from the work of Newman, Strogatz, and Watts (NSW) [NSW00] which provides a random graph model for datasets with pre-specified degree distributions. This will allow us to incorporate prior knowledge about rating patterns into the analysis. The basic idea in random graph theory is to describe properties of a given family of graphs in terms of characteristics of the probability distribution from which the graphs are drawn. Besides the NSW model, there are several models popular in the literature, most notably the Erdös-Rényi [ER59] and the Aiello-Chung-Lu [ACL00] models. However, the NSW model is the only model that characterizes the family of graphs in terms of the degree distributions of the vertices of the graph. The Erdös-Rényi model's assumptions dictate a Poisson distribution of degrees and the Aiello-Chung-Lu model is motivated by a power-law distribution. As we will show later in the thesis, graphs induced by the hammock jump depict a variety of degree distributions, thus rendering the NSW model most appropriate for our purposes. From the original bipartite graph, $\mathcal{R} = G(P \cup M, E)$, we develop two models, one for the social network graph G_s and one for the recommender graph G_r.

3.3.1 Modeling the Social Network Graph

Recall that the social network graph $G_s(P, E_s)$ is undirected and E_s is induced by a jump function $\mathcal{J}$ on $\mathcal{R}$. Our analysis below is drawn from [NSW00] and we will use the same notation to be consistent. We can write a generating function $G_0(x)$ for the probability distribution of the vertex degrees in G_s:

$$G_0(x) = \sum_{k=0}^{\infty} p_k x^k \tag{3.1}$$

where p_k is the probability that a randomly chosen vertex in G_s has degree k, so that:

$$G_0(1) = \sum_{k=0}^{\infty} p_k = 1. \tag{3.2}$$

Notice that for a randomly chosen vertex in this graph, $G_0(x)$ also gives us the distribution of the immediate neighbors of that vertex. Thus, the average number of first neighbors of a given node is the average degree z:

$$z = \sum_k k p_k = G_0^{'}(1) \tag{3.3}$$

This is one of the main advantages of using generating functions. Statistical properties of sequences (such as vertex degrees) can be expressed in terms of derivatives of the generating function. Let us compute the distribution of the second neighbors of a randomly chosen vertex. We achieve this by first characterizing the random process involved and expressing it in mathematical notation. We start at a randomly chosen vertex, follow the edges at this vertex to reach its k nearest neighbors, and from each of these vertices, choose edges to arrive at the second nearest neighbor (taking care to ensure that we don't use the edge that we originally came along). A given vertex has degree k with probability p_k. If we arrived at this vertex using one of the k edges, we have $k-1$ choices of outgoing edges. We thus, get the generating function of outgoing edges leaving a vertex reached by following a randomly chosen edge:

$$G_1(x) = \frac{\sum_k k p_k x^{k-1}}{\sum_k k p_k} \tag{3.4}$$

The numerator can be interpreted as follows; there are kp_k ways of arriving at a vertex which has $k-1$ outgoing edges. The denominator is designed to normalize the distribution. This expression can be simplified as:

$$G_1(x) = \frac{G_0^{'}(x)}{G_0^{'}(1)} = \frac{G_0^{'}(x)}{z} \tag{3.5}$$

We can then compose the G_0 function over G_1 to obtain the distribution of second neighbors of a randomly chosen vertex:

$$G_0(G_1(x)) \quad = \quad \sum_k p_k [G_1(x)]^k \tag{3.6}$$

Differentiating this new generating function and evaluating it at $x = 1$ produces the average number of second neighbors:

$$\begin{aligned} z_2 &= \sum_k kp_k \frac{1}{z} \sum_k k(k-1)p_k \\ &= G_0^{'}(1)G_1^{'}(1) \end{aligned} \tag{3.7}$$

Similarly, the distribution of third neighbors is $G_0(G_1(G_1(x)))$. The average number of third neighbors will then be given by:

$$z_3 = G_0^{'}(1)G_1^{'}(1)G_1^{'}(1) \tag{3.8}$$

and so on. In the general case, the average number of nodes at a distance m is given by:

$$z_m = \left(\frac{z_2}{z_1}\right)^{m-1} z_1 \tag{3.9}$$

Newman, Strogatz, and Watts [NSW00] illustrate how we can use this expression to estimate the average length l_{pp} of shortest paths (in the graph G_s). In one step, we can reach $z_1 = z$ vertices. In two steps, we can reach $z_1 + z_2$ vertices. In l_{pp} steps, we could reach all the vertices N_P (number of people) of the graph G_s, if:

$$1 + \sum_{m=1}^{l_{pp}} z_m = N_P \tag{3.10}$$

where the constant 1 counts the vertex under consideration. Unfolding this expression, we get:

$$\left(\frac{z_2}{z_1}\right)^{l_{pp}} = \frac{(N_P - 1)(z_2 - z_1) + z_1^2}{z_1^2} \tag{3.11}$$

or

$$l_{pp} = \frac{log[(N_P - 1)(z_2 - z_1) + z_1^2] - log[z_1^2]}{log[z_2/z_1]} \tag{3.12}$$

This formula will be our primary means of characterizing the distances between pairs of people in G_s, induced by a jump function $\mathcal{J}$. Notice that since our empirical evaluation will proceed with actual datasets (from which measurements of p_k could be made), calculation of z_1 and z_2 is simplified considerably, by casting p_k as the fraction of vertices in the graph having degree k.

3.3.2 Modeling the Recommender Graph

Recall that the recommender graph $G_r(P \cup M, E_{sd} \cup E_{md})$ is directed, and hence the generating function for vertex degrees should capture both indegrees and outdegrees:

$$G(x, y) = \sum_{j=0,k=0}^{j=\infty,k=\infty} p_{jk} x^j y^k \tag{3.13}$$

where p_{jk} is the probability that a randomly chosen vertex has indegree j and outdegree k. From the JC construction, we know that movie vertices have outdegree 0 (the converse is not true, vertices with outdegree 0 could be people nodes 'stranded' as a result of a severe jump constraint). Notice also that by using the joint distribution p_{ij}, independence of the indegree and outdegree distributions is *not* implied. We will show in the next chapter that this feature is very useful. In addition, the average number of edges entering (or leaving) a vertex is zero. Thus:

$$\sum_{jk}(j-k)p_{jk} = \sum_{jk}(k-j)p_{jk} = 0 \tag{3.14}$$

If we forget the directed nature of the graph, the average degree z is given by:

$$\begin{aligned} z &= \left.\frac{\partial G}{\partial x}\right|_{x=1,y=1} && (3.15)\\ &= \sum jp_{jk} && (3.16)\\ &= \sum kp_{jk} && (3.17)\\ &= \left.\frac{\partial G}{\partial y}\right|_{x=1,y=1} && (3.18)\end{aligned}$$

where Eq. 3.17 is obtained by using Eq. 3.14. In addition to counterparts of the generating functions $G_0(x)$ and $G_1(x)$ for the outdegrees in G_r, we can obtain generating functions $F_0(x)$ and $F_1(x)$ for the indegrees, as well. Newman, Strogatz, and Watts [NSW00] state that these are given by:

$$\begin{aligned} F_0(x) &= G(x,1) && (3.19)\\ G_0(y) &= G(1,y) && (3.20)\\ F_1(x) &= \left.\frac{1}{z}\frac{\partial G}{\partial y}\right|_{y=1} && (3.21)\\ G_1(y) &= \left.\frac{1}{z}\frac{\partial G}{\partial x}\right|_{x=1} && (3.22)\end{aligned}$$

We thus obtain new expressions for z_1 and z_2:

$$\begin{aligned} z_1 &= G_0'(1)\\ &= \sum_{jk} kp_{jk} && (3.23)\\ z_2 &= G_0'(1)G_1'(1)\\ &= \sum_{jk} jkp_{jk} && (3.24)\end{aligned}$$

The average path length l_r can be calculated as before:

$$l_r = \frac{log[(N_P+N_M-1)(z_2-z_1)+z_1^2]-log[z_1^2]}{log[z_2/z_1]} \tag{3.25}$$

where $N_P + N_M$ denotes the size of the recommender graph G_r (again, assuming that the graph is one giant component), with N_M denoting the number of movies. Notice that l_r includes paths from people to movies, as well as paths from people to people. The average length of only movies from people l_{pm} can be expressed as:

$$l_{pm} = \frac{(l_r(N_P(N_P-1) + N_P N_M) - l_{pp} N_P(N_P-1))}{N_P N_M} \tag{3.26}$$

3.4 Caveats with the NSW Equations

There are various problems with using the above formulas in a realistic setting [Hea01]. First, unlike most results in random graph theory, the formulas do not come up with any guarantees and/or confidence levels. Second, all the equations above are obtained over the ensemble of random graphs that have the given degree distribution, and hence assume that all such graphs are equally likely. The specificity of the JC construction implies that the G_s and G_r graphs are poor candidates to serve as a 'typical' random instance of a graph.

In addition, the equations utilizing N_P and N_M assume that all nodes are reachable from any starting vertex (i.e., the graph is one giant component). This will not be satisfied for very strict jumping constraints. In such cases, Newman, Strogatz, and Watts suggest the substitution of these values with measurements taken from the largest component of the graph. Expressing the size of the components of the graph using generating functions is also suggested [NSW00]. However, the complexity of jumps such as the hammock can make estimation of the cluster sizes extremely difficult, if not impossible (in the NSW model). We leave this issue to future research.

Finally, the NSW model is fundamentally more complicated than traditional models of random graphs. It has a potentially infinite set of parameters (p_k), doesn't address the possibility of multiple edges, loops and, by not fixing the size of the graph, assumes that the same degree distribution sequence applies for all graphs, of all sizes. These observations hint that we cannot hope for more than a qualitative indication of the dependence of the average path length on the jump constraints. In the next chapter, we describe how well these formulas perform on two real-world datasets.

Chapter 4

Experimental Results

We devote this chapter to an investigation of two actual datasets from the movies domain; namely the EachMovie dataset, collected by the Digital Equipment Corporation (DEC) Systems Research Center, and the MovieLens [RK] dataset developed at the Univ. of Minnesota. Both these datasets were collected by asking people to log on to a website and rate some movies. The time spent rating movies was paid off by procuring predictions of ratings for other movies not yet seen, which the recommendation engines calculated based on the submitted ratings and some other statistical information.

The datasets contain some basic demographic information about the people (age, gender, etc) as well the movies (title, genre, release date, etc). Associated with each person and movie are unique *id*s. The rating information (on a predefined numeric scale) is provided as a set of $3-$tuples: the person *id*, the movie *id*, and the rating given for that movie by that person. Some statistics for both the datasets are provided in Table 4.1. Notice that only a small number of actual ratings are available (as a fraction of all possible combinations), and yet the bipartite graphs of people versus movies are connected, in both cases.

4.1 Preliminary Investigation

Upon closer inspection of EachMovie and MovieLens, it can be realized that both these datasets exhibit what we can refer to as a *hits-buffs* structure. Specifically, there are some

Table 4.1: Some statistics for the EachMovie and MovieLens datasets.

Dataset	*Number of people*	*Number of movies*	*Sparsity*	*Connected?*
MovieLens	943	1,682	93.70%	Yes
EachMovie	61,265	1,623	97.63%	Yes

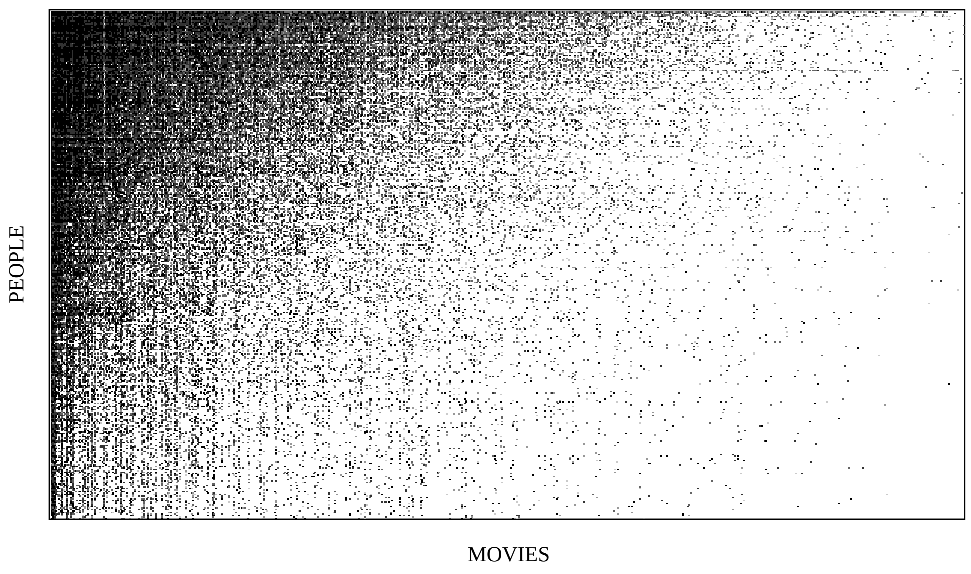

Figure 4.1: Hits-buffs structure of the (reordered) MovieLens dataset.

people (the *buffs*) who see (and rate) almost all movies and there are some movies (the *hits*) that are seen by nearly all people. Assume that people are ordered according to a *buff index* b: A person with buff index 1 has seen the most number of movies, and so on. For example, in the EachMovie dataset, the person with buff index 1 has seen $1,455$ movies from the total of $1,623$. These $1,455$ movies have, in turn, been seen by $61,249$ other people. Thus, within two steps, a total of $62,705$ nodes in the graph can be walked; with other choices of the intermediate 'buff' node, the entire graph can be shown to be connected in, at the most, two steps. The MovieLens dataset satisfies a similar property.

Furthermore, the relationship between the buff index b and the number of movies seen by the buff $P(b)$ follows a power-law distribution:

$$P(b) \propto b^{-\beta}$$

A similar trend can be observed for the *hits*[1]. This feature of both datasets appears to have gone unnoticed by prior recommender systems research (including the designers and collectors of both datasets). We show how such structures can be exploited advantageously to achieve compelling recommendations. To better demonstrate the structure, we reorder the people and movie *id*s, so that the relative positioning of the *id*s denotes the extent of a person being a buff, or a movie being a hit. For example, person *id* 1 refers to the person with buff index 1 and movie *id* 1 refers to the movie with hit index 1. Figure 4.1 illustrates the hits-buffs structure of the MovieLens dataset.

[1]Graphs with such power-law distributions can thus also form small-worlds [ASBS00], as evidenced by the short length between any two people in both MovieLens and EachMovie.

4.2 Outline of Setup

We describe our experimental setup below. The goal of our experiments was to investigate the effect of the hammock width w on the average characteristic path lengths of the induced G_s and G_r graphs for the above datasets. The pseudocode for these experiments is provided in Appendix A. Both the datasets were sanitized by first removing the rating information; in other words, we explored a purely connection-oriented jump. For various values of the hammock width w, we formed the social network and recommender graphs and calculated their degree distributions (for the largest component, when the graphs were disconnected). Using parallel implementations of Djikstra's and Floyd's algorithms, we computed the average path length for (the largest component of) both the secondary graphs and compared the results with the predictions of the equations from Chapter 3.

4.2.1 MovieLens

Fig. 4.2 describes the number of components in G_r as a result of imposing increasingly strict hammock jump constraints. Up to about $w = 17$ the graph remains in one piece and rapidly disintegrates after this threshold point. The high value of this threshold for transition is not surprising, since the designers of MovieLens insisted that every participant rate at least $\kappa = 20$ movies! As observed from our experiment results, after the threshold point and up to $w = 28$, there is still only one giant component with the separated people nodes stranded as *islands* (Fig. 4.3). Specifically, the degree distributions of the MovieLens social network graphs for $w > 17$ show us that the disconnected people nodes which are not part of the giant component do not form any other connected components and are left by themselves. We say that a jump shatters a set of nodes if the vertices not part of the giant component do not have any edges. This aspect of the formation of a giant component is, of course, well known from random graph theory [Bol85]. Since the JC construction views the movies as a secondary mode, we can ensure that only the strictest hammock jumps shatters the N_M movie nodes. Fig. 4.4 demonstrates that the movie nodes are not stranded as a result of hammock constraints up to $w = 29$.

We then compared the lengths l_{pp} (Fig. 4.5) and l_r (Fig. 4.6) using both actual computations and the formula predictions. The increase in length up to the threshold point is understandable, since with links being removed from the giant component, paths of greater length have to be traversed to reach other nodes. After the threshold, the relative stability of the length indicates that the only edges being lost are those that are associated with the stranded people nodes. Notice also that the values of the lengths lie between 1 and 2, emphasizing the role of the hits-buffs structure. While the formulas capture the qualitative behavior of the effect of the hammock width, it is obvious from Fig. 4.5 that they postulate significantly less clustering than is actually observed. A possible explanation is given later in the chapter.

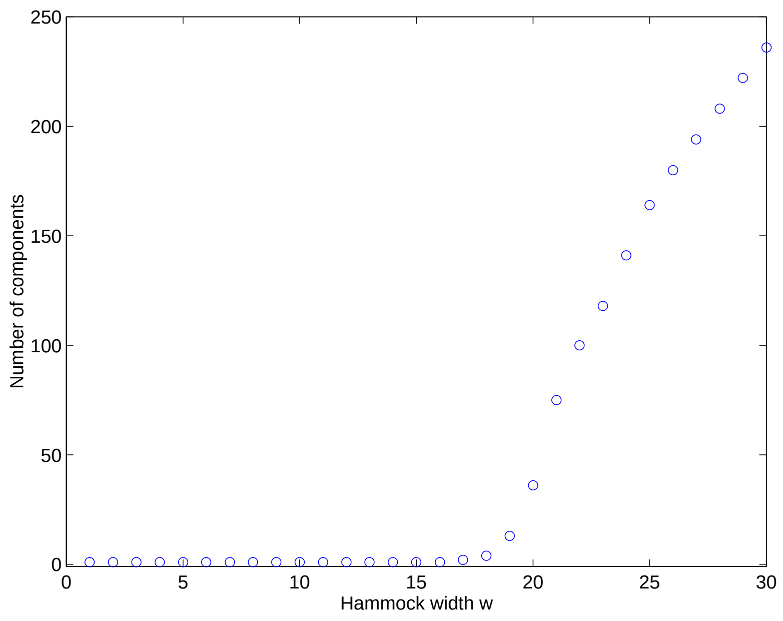

Figure 4.2: Effect of the hammock width on the number of components in the G_r graph induced from the MovieLens dataset.

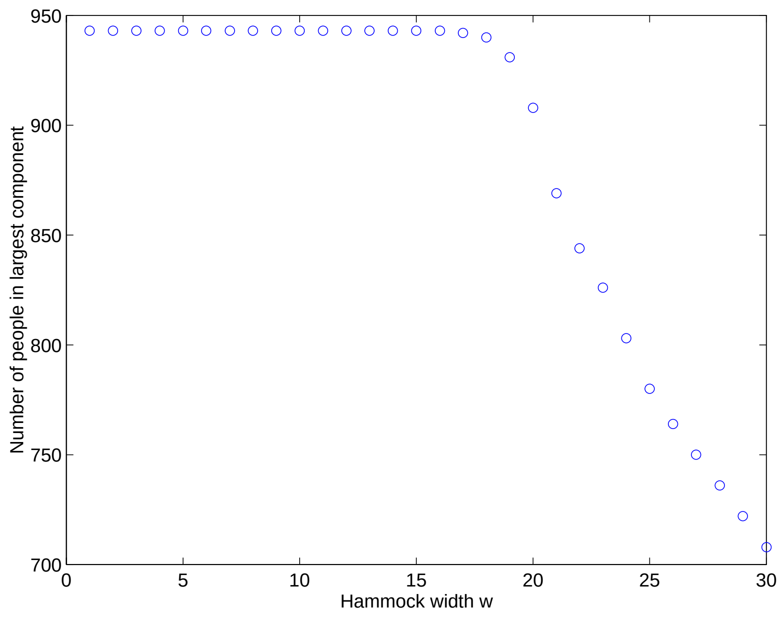

Figure 4.3: Effect of the hammock width on the number of people in the largest components in the G_r graph (MovieLens).

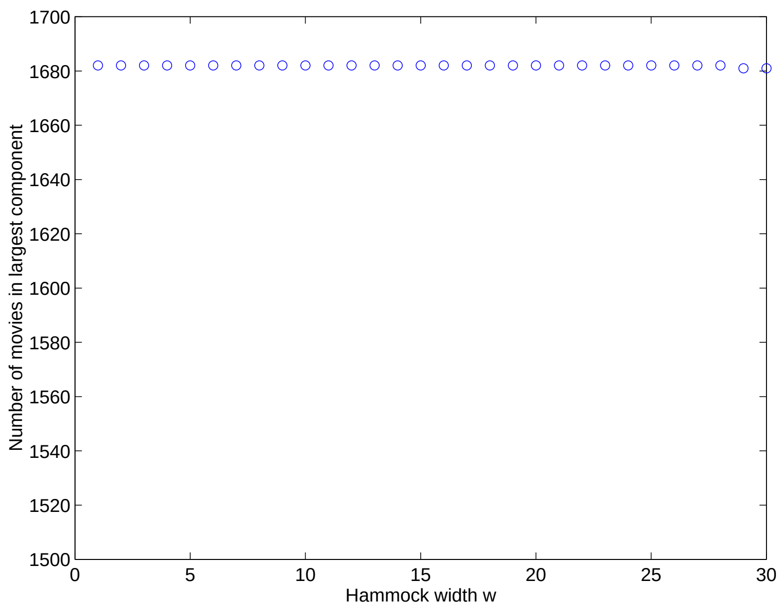

Figure 4.4: Effect of the hammock width on the number of movies in the largest components in the G_r graph (MovieLens).

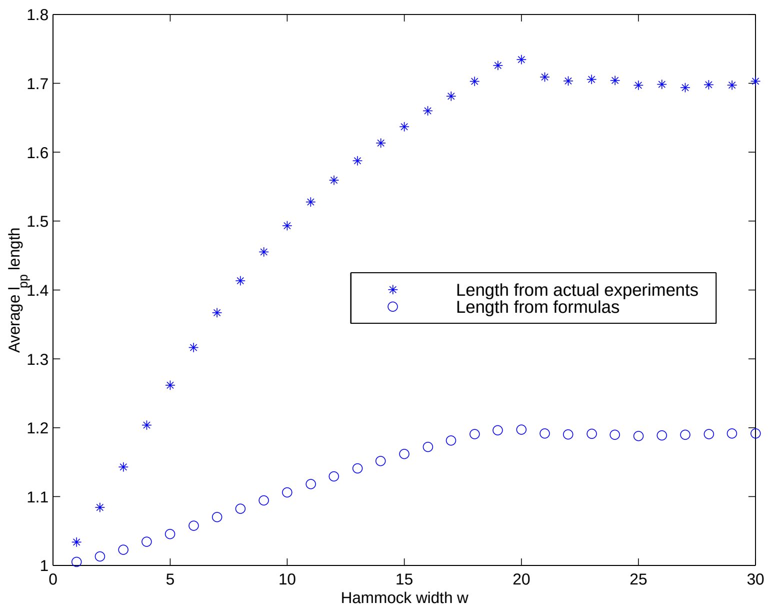

Figure 4.5: Comparison of the l_{pp} measure (MovieLens) from actual computations and from the formulas.

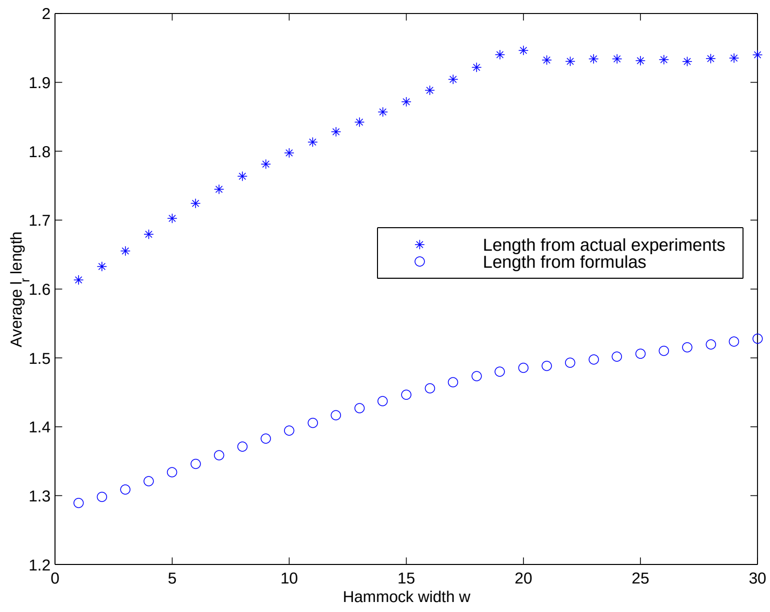

Figure 4.6: Comparison of the l_r measure (MovieLens) from actual computations and from the formulas.

Finally, we present comparisons of the l_r measurements and formula estimations (Fig. 4.6). In this case, there is substantially better agreement between the two values, as well as tracking of the qualitative change. Once again, the formulas assume significantly less clustering than real data. In other words, the higher values of lengths from the actual measurements indicate that there is some source of clustering, not captured by the degree distribution alone (which is used as input to the formulas).

4.2.2 EachMovie

The evaluation of the EachMovie data was more tricky owing to inconsistencies in data collection. For example, the dataset was collected in two phases, with entirely different rating instructions and scales in the two situations, and contained duplicate ratings. We concentrated on the portion of the data collected in 1997 and created a synthetic dataset that has the same characteristics as this reduced dataset. Specifically, we created a 500 person, 75 movie dataset such that the person with buff *id* b has seen the *first* $\lceil 75b^{-\epsilon} \rceil$ movies (recall that the movies are also ordered according to their hit *id*). An $\epsilon = 0.7$ produces a dataset with a minimum rating of 1 movie (95.5% sparse), while $\epsilon = 0.27$ produces a minimum rating of 15 movies (with sparsity 76.63%). The choice of ϵ thus provides a systematic way to analyze the effect of the minimum rating constraint κ (see Fig. 4.7). In addition, for each (person,movie) edge of these synthetic graphs, we generate a uniform (discrete) random variate in $[0, 10]$ and rewire the movie 'endpoint' of the edge if this variate is < 2. This device

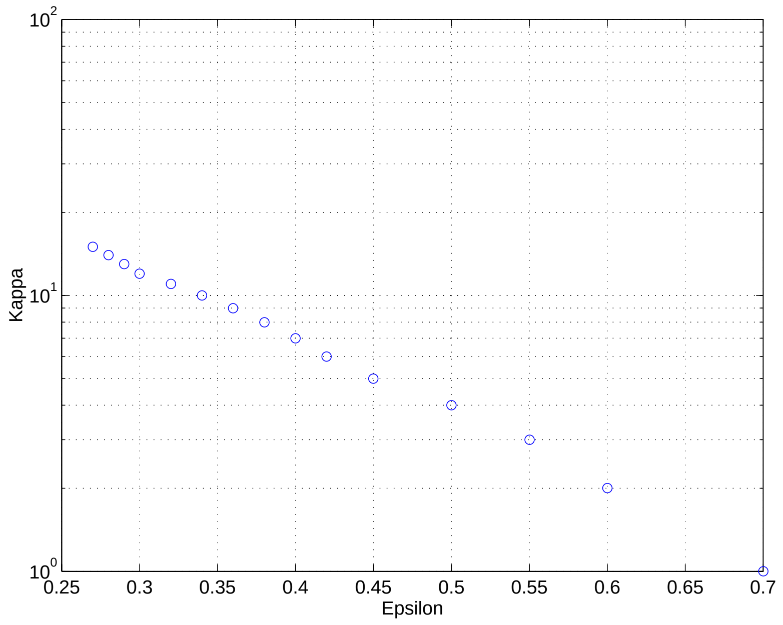

Figure 4.7: Calibrating the value of ϵ in the synthetic model of EachMovie produces datasets with required specifications on minimum rating κ.

models deviations from a strict hits-buffs distribution. We generated 15 such graphs and ensured that they were connected (sometimes a few bits were manually turned on to ensure connectedness). These served as the starting points for our analysis. For each of these 15 graphs, we varied the hammock width w from 1 to 25 and repeated the length calculations (using both formulas and actual measurements) for the social network and recommender graphs.

Like the MovieLens experiment, the formulas for EachMovie predict shorter l_{pp} lengths (and consequently, lesser clustering) than observed from actual experiments. Fig. 4.8 depicts the L_∞ metric for the discrepancies between the measured and actual length values for 25 values of the hammock width w, for each of the 15 graphs.

Notice the relatively linear growth of the discrepancy as κ increases. In the considered κ range, the hammock constraints shatter the graph into many small components, so the formulas are applied over ever-decreasing values of n, rendering them ineffective. For example, for $\kappa = 15$, a hammock width $w = 25$ shatters the graph into 53 components. Since the hammock width w was varied in $[1, 25]$ over a range of $[1, 15]$ for κ, we have reason to believe the this graph will taper off when extrapolated to higher values of κ. While this experiment does not provide any new insight, it hints at a fairly bounded growth in the L_∞ discrepancies for l_{pp} lengths.

The comparisons for the l_r lengths tell a different story (see Fig. 4.9). At low values of κ, the l_r length calculated from formulas is highly erroneous for even medium values of hammock

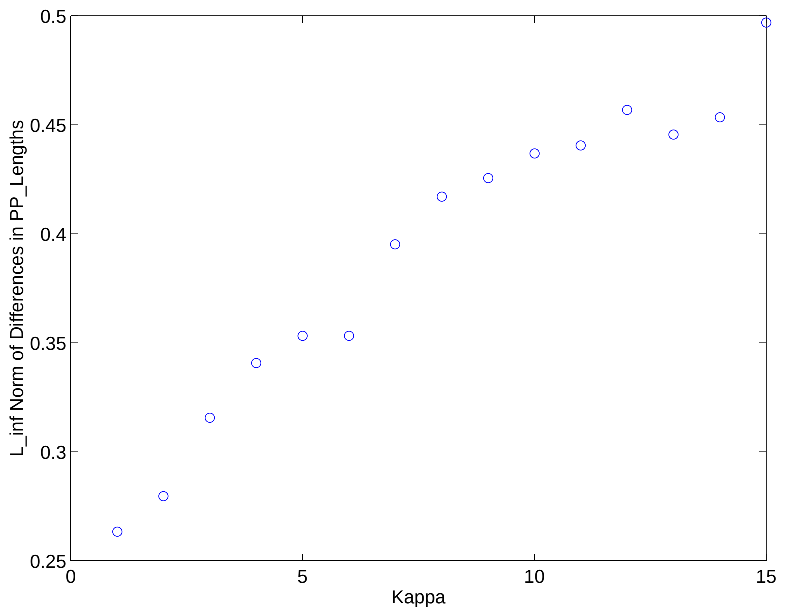

Figure 4.8: Comparison of the l_{pp} measure (EachMovie) from actual computations and from the formulas for varying values of κ.

width w (Fig. 4.9, left) whereas we see the standard 'less clustering credit' picture for high values (≥ 12) of κ (Fig. 4.9, right). In particular, for $\kappa = 1, w = 25$, the formulas predict an average l_r length of 4.24! This is counter-intuitive given that the largest component for this shattering itself consists of only 4 people nodes (and 71 movie nodes).

This problem arises because the NSW model does not prohibit a multiplicity of edges between a pair of nodes. Let us look at this situation more closely. For $\kappa = 1, w = 25$, the largest component has the degree distribution given in Table. 4.2. Notice that the nodes with outdegree 0 are obviously the movie nodes and the nodes with non-zero outdegree are the people nodes. Thus, two of the people are connected to two people (each), and two of the people are connected to three people (each). A graph that satisfies this property is shown in the left of Fig. 4.10. The origin of the value of 4.24 can be traced back, rather to the NSW model's postulation of the unlikely 'culprit' graph shown in the right of Fig. 4.10. Notice that this graph satisfies the exact same distribution but by allowing multiple edges, many movie nodes would be counted more than once, and with greater (ever increasing) path lengths. One cycle between two people can thus effect two extra hops in the length calculations, rendering the estimates inaccurate. As observed from our results, as κ increases, the largest component increases in size. For example, when $\kappa = 15$ and $w = 25$, the largest component has 53 people whereas for $kappa = 1$ and $w = 25$, there are only 4 people in the largest component. With increase in largest component size for higher values of κ, the proportion of such pathological graphs decreases; hence the observed (qualitative) agreement between actual and predicted values.

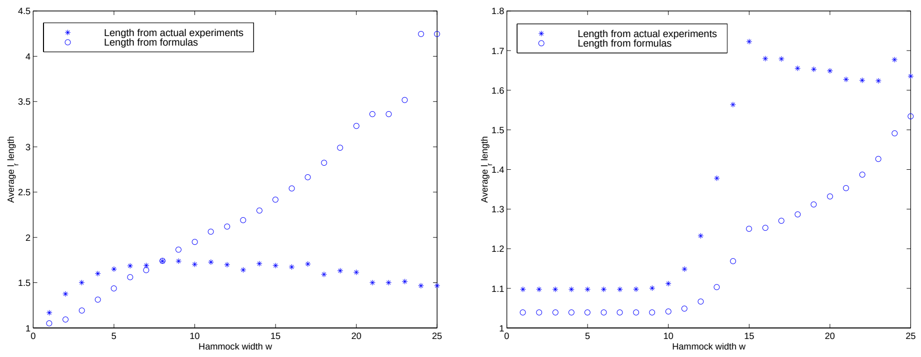

Figure 4.9: Comparison of the l_r measure (EachMovie) from actual computations and from the formulas for minimum rating constraint $\kappa = 1$ (left) and minimum rating constraint $\kappa = 15$ (right).

Table 4.2: Joint degree distribution for the largest component of the recommender graph (EachMovie) when $\kappa = 1, w = 25$. Only the non-zero entries are shown.

$kappa = 1$	w	Size of largest component
1	0	23
2	0	16
3	0	13
4	0	19
2	31	1
2	65	1
3	37	1
3	47	1

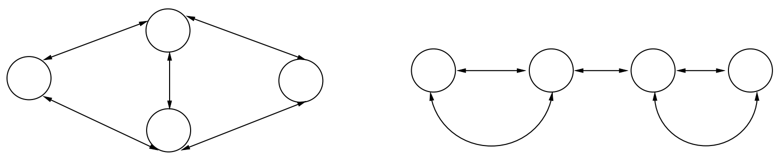

Figure 4.10: Two graphs that satisfy the degree distribution given in Table. 4.2. For simplicity, only the people nodes are shown.

4.3 Discussion of Results

We can make several preliminary observations from the results so far:

1. As Newman, Strogatz, and Watts point out [NSW00], the random graph model defined by degree distributions makes strong qualitative predictions of actual lengths, using only local information about the number of first and second nearest neighbors (z_1 and z_2 from Chapter 3). We have shown that this holds true even for graphs induced by hammock jumps.

2. For sufficiently large values of κ, the relationship between hammock width w and average l_{pp} length follows two distinct phases: (i) In the first regime, there is a steady increase of l_{pp} up to a threshold $< \kappa$, where only edges are lost. (ii) In the second phase, nodes are shattered but without much effect on the average l_{pp} values. This two-phase phenomenon can thus serve as a crucial calibration mechanism for connecting the number of people to be brought together by a recommender system and the average l_{pp} length. For example, one can define a parametric study such as discussed here for a new domain and proceed to first demarcate the phases. Choices of hammock width w can then be made, depending on the feasibility of realizing an appropriate κ (in real-life) and any desired constraints on l_{pp}.

3. The average l_r lengths are within the range $[1, 2]$ in both predicted results and actual measurements, highlighting that the hits-buffs structure is captured in the NSW model. Caution has to be exercised whenever the graph size n gets small, as Fig. 4.10 shows. In our case, this happens when the shattering constraint of hammock width w falls above the limits posed by the minimum rating constraint κ. Of course, this behavior will also be observed for other forms of 'strict jumps.'

4. Both l_{pp} and l_r lengths postulate consistently less clustering than observed in the real data. We attempt to address this below. The typical random graph model has a Poisson distribution of edges [Bol85], whereas, as seen earlier, real datasets depict a power-law distribution [FFF99, CD00]. The power-law feature is sometimes described as 'scale-free' or 'scale-invariant' since a single parameter (the exponent of the law) captures the size of the system at all stages in its cycle (the x-axis of the law). A log-log plot of values would thus produce a straight line, an effect not achievable by traditional random graph models. Barabási and Albert [BA99] provide two sufficient conditions for this property: *growth* and *preferential attachment*. Growth refers to the ability of the system to dynamically add nodes; thus random graph models that fix the number of nodes are unable to 'expand.' Preferential attachment refers to the phenomenon that nodes that have high degrees have a greater propensity of being linked to, by new nodes. In our case, a movie that is adjudged well by most people is likely to become a hit when additional people are introduced. Barabási refers to this as a 'rich get richer' effect.

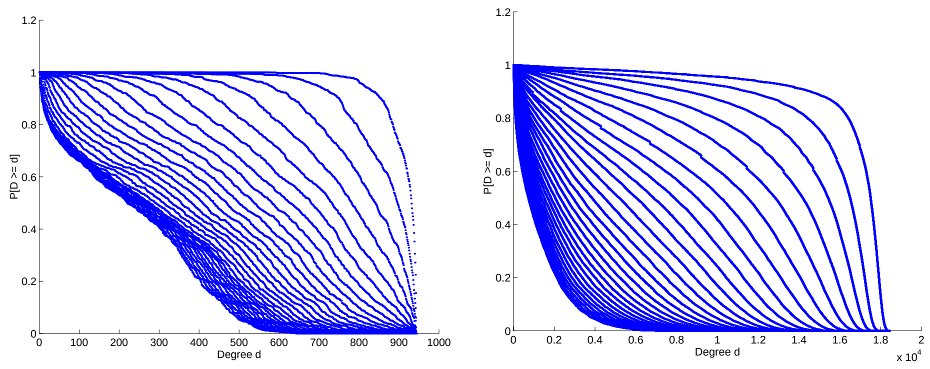

Figure 4.11: Cumulative frequency distribution (of degrees) as a function of the degree for (left) MovieLens and (right) EachMovie datasets, for hammock widths from 1 to 30.

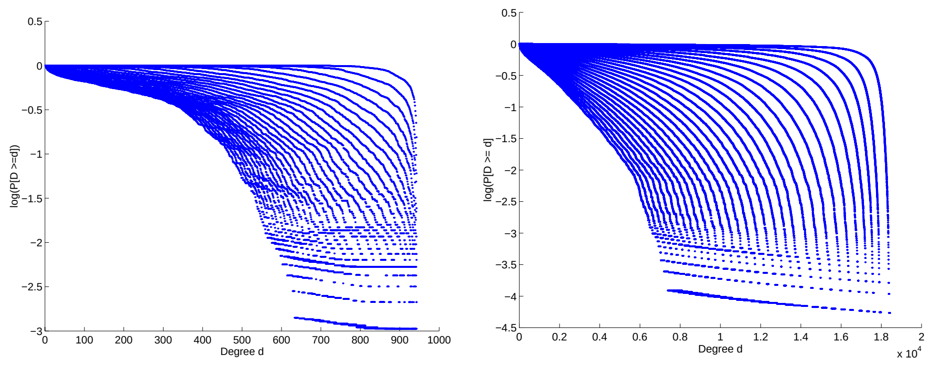

Figure 4.12: Logarithm of the cumulative frequency distribution (of degrees) as a function of the degree for (left) MovieLens and (right) EachMovie datasets, for hammock widths from 1 to 30.

To characterize the possible causes in our domain, we consider the distribution of degrees in the social network graphs G_s of both MovieLens and (the actual) EachMovie datasets (see Fig. 4.11). In the figure, top to bottom indicates increasing hammock widths. For the entire range of the hammock width w, both datasets do not follow a strict power law. For low values of w, there is a small and steadily increasing power-law regime, followed by an exponential cutoff. For higher values of w, the graph 'falls' progressively earlier and the CDF resembles a gaussian or exponential decay, with no power-law behavior. This is evident if the logarithm of the CDF is plotted, as in Fig. 4.12 where top to bottom indicates increasing hammock widths. Notice the significant cusp in the left side of both graphs, depicting the qualitative change in behavior. These two extremes of deviations from power-law behavior (at low and high values of w) are referred to as (resp.) *broad-scale* and *single-scale* in [ASBS00] to distinguish them from the scale-free behavior of power-laws.

The authors of [ASBS00] also show that *aging* and *capacity* can cause such deviations from a pure power-law. Aging refers to the fact that after a certain point in time, nodes stop accumulating edges. Capacity refers to resource-bounded environments where cost and economics prevent the hits from becoming arbitrarily greater hits. They demonstrate these observations for environments such as airports (capacity limitations on runways), and natural networks (aging caused by people dying).

In our domain, recall that the G_s graphs model the connectivities among people, and are hence indirect observations from an underlying bipartite graph. One possible explanation for the deviations of the connectivities in G_s from from power-law behavior is suggested in a study done by Robalino and Gibney [RG99b]. The paper models the impact of movie demand on the social network underlying a 'word of mouth' recommendation. The two new factors introduced here are *expectation* and *homogeneity* (of the social network). The authors suggest that "some movies might end up having low demand, depending on the initial agents expectations and their propagation through the social network." Furthermore, they assume that negative information (ratings) obtained early in a movie's lifetime can have substantial effect in a "homogeneous" network (one where individuals trust each others opinions strongly than in other networks). At this point, we are unable to accept or reject this observation due to lack of information about the social dynamics in which the data was collected in MovieLens and EachMovie.

However, such an effect might not still model the shift in the emphasis from a broad-scale behavior to a single-scale behavior as w increases. Fortunately, this is easy to explain algorithmically from the JC construction. For higher values of w, the degree distributions resemble more and more a typical random graph (for smaller values of n), which has a connectivity characterized by a fast decaying tail (such as a Poisson). Insisting on greater values of w leads to higher and higher decays, such that for sufficiently large w (relative to κ), no power-law regime is visible [ASBS00].

Chapter 5

Concluding Remarks

This research makes two key contributions:

1. We have shown how algorithms for recommender systems can be characterized as jumping connections in a bipartite graph (or affiliation network). This view enables a new methodology to conduct experimental analysis and comparison of algorithms. Using this approach, algorithms can be distinguished by the pairs of nodes that are brought together.

2. We have demonstrated the application of the JC framework to a particular form of jump — the hammock. The two-phase phenomenon observed for the induced social network graph allows us to connect the minimum rating constraint κ (the approximate point at which the phase transition appears), the hammock width w (a factor parameterizing the jump), the size of the largest component (the number of people connected by the jump), and the average l_{pp} length (minimization of which is desirable).

The eventual success of the proposed methodologies relies on the expressiveness of the representations supplied to the recommender system builder and his/her ability to reason effectively with such representations. We have shown how our graph-theoretic model can help view recommendation as a process of making connections in a graph. Ideally, a recommender system builder will fix one or more of the variables among κ, w, average l_{pp} length, and the types and numbers of nodes brought together by the jump. An analysis for a particular degree distribution will help make estimates for other parameters.

In future, this work can be extended in several directions:

1. Our emphasis on experimental algorithmics has focused on a single jump with multiple parameterizations (of hammock width w). The JC framework can be explored to obtain strong theoretical results about the efficacies of qualitatively different jumps, such as outlined in Section 3.2. Many of the jumps presented there, besides the hammock,

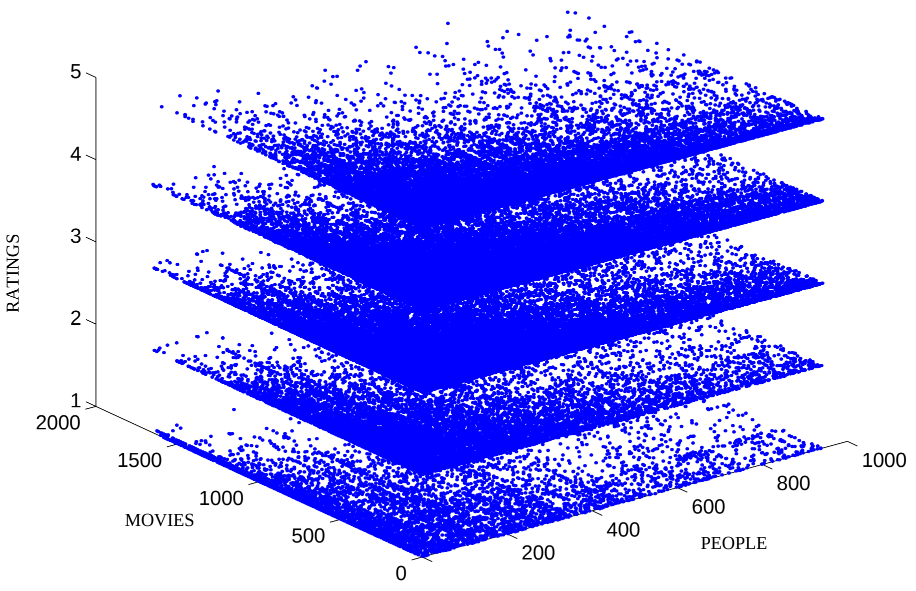

Figure 5.1: A stratified view of the MovieLens dataset demonstrates hits-buffs structures at all rating levels.

are of intrinsic commercial and economic value. Microsoft's E-Commerce Server 2000 series now provides algorithms for collaborative filtering. The recent interest in mixture models and latent variables [HCM$^+$00, HP99] coupled with the relational nature of their induced representations is a strong indicator for the applicability of JC in these situations. Inference of connections in these networks is usually very difficult; an algorithm (jump) that makes similar connections with significantly less overhead will be preferable to the complexity of using Bayesian/dependency networks for inference. In addition, ratings information can be included in the analysis of jumps. Fig. 5.1 shows the ratings information present in the MovieLens dataset. As can be seen, the hits-buffs structure gets stratified into multiple layers and can be used advantageously both within a ratings layer and across layers.

2. The formulas presented in this thesis for l_{pp} and l_{pm} are derived from the parameters of the induced G_s and G_r graphs. One possible direction of work is to cast these variables in terms of parameters of the original bipartite graph (dataset) $\mathcal{R}$. However, the NSW model is very difficult to analyze, for all but the simplest forms of jumps. Recently, Aiello et al. [ACL00] have introduced a random graph model for massive graphs that is modeled after power-laws. Unlike the NSW model and like traditional random graph models, their model has only two parameters (the intercept and slope of the power-law plotted on a log-log scale). Estimations of graph properties such as diameter have very recently [Lu00] been initiated for this new model and it appears to be a promising candidate for application to recommender systems.

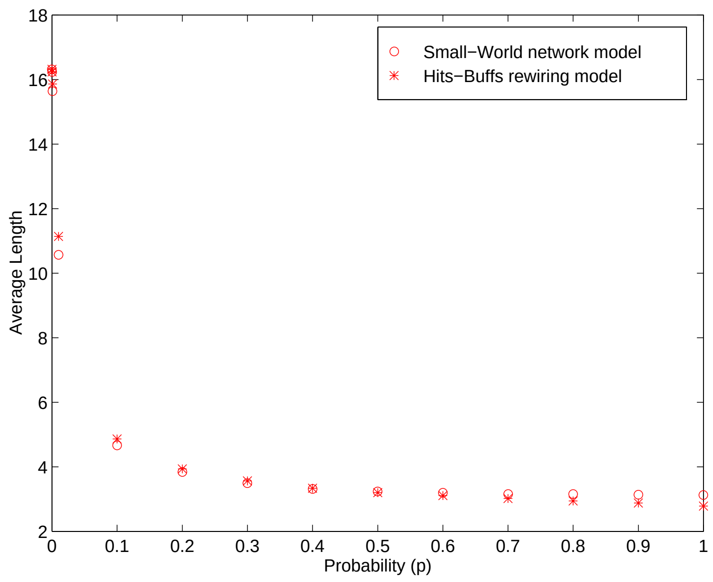

Figure 5.2: Effect of two rewiring models on the characteristic path length, starting from a regular wreath network.

3. The existence of ratings structures such as the hits-buffs distribution and our ability to exploit them (to minimize factors such as the average length) is very crucial to the success of recommender systems. An interesting question is how much of the hits-buffs structure should be present in a dataset? to provide high quality recommendations. The answer to this question has implications for the currently arcane process of data acquisition. Typically, movies (the secondary mode) are partitioned into two sets: (i) a *hot set*, that almost everybody is required to rate (to increase commonality of ratings) [AWWY99, GRGP00], and (ii) a *cold set* that is used to ensure adequate coverage [AWWY99]. A more detailed understanding of the hits-buffs structure would provide new methodologies to address this dichotomy.

   To study this question, we conducted a Watts-Strogatz analysis with a rewiring model that preferentially rewires to some nodes with greater probability. Thus, nodes that are hits or buffs have a greater propensity of being linked to. Fig. 5.2 shows the results for the average length in a one-mode graph. In the small α range, the results from a preferential rewiring are virtually indistinguishable from a random rewiring, so both of them serve to bring the length down to nearly identical limits. In the large α range, the effect of the hits-buffs structure is evident in the reduced length. It is thus unclear what a 'base structure' should be to explore the role of hits-buffs in length reduction. Recent research on the modeling of dynamic systems shows that power-laws such as hits-buffs are crucial [CD00] to ensure robustness; more work needs to be done to delineate connections to data collection and desired metrics in recommender systems.

4. We also plan to design an interactive graphical tool that allows the exploration of recommendation spaces. The user can be given an opportunity to define a jump and basic parameters of the graph. The system can then help estimate properties such as the average length, clustering coefficient, and the size of components induced by the jump. A more interesting question arises by inverting this process: given a collection of nodes that the user would like to bring together, what are the jumps that make this possible? We refer to this scenario as 'inverse personalization.' This is currently being used in a variety of web sites that do targeted marketing of products. For each product or set of products, a *niche* is identified that ensures that the products are the only examples satisfying all the desired characteristics. In our formulation, niches are defined by new ways of jumping.

5. Jumping connections also has implications for the privacy aspects of recommender systems. In many contexts (committee voting habits, for instance), the bipartite graph $\mathcal{R}$ is not directly observable; thus one of the modes is 'hidden.' However, people would be comfortable divulging information pertaining to the social network graph G_s induced by the skip jump (i.e., neighbors with whom they have at least one aspect in common). In some cases, this could be implicitly observed. The goal here is to get estimates (of l_{pp}, for instance) for other forms of jumps using such information (that appear deceptively harmless to provide). The validity of such estimates would have to be qualified using notions from random graph theory, in turn leading to a better understanding of the privacy implications of modeling connections.

Bibliography

[ABFK99] V. Anupam, Y. Breitbart, J. Freire, and B. Kumar. "Personalizing the Web using Site Descriptions". In *Proceedings of the Tenth International Workshop on Database & Expert Systems Aplications*, pages 732–738, 1999.

[ACL00] W. Aiello, F. Chung, and L. Lu. "A Random Graph Model for Massive Graphs". In *Proceedings of the ACM Symposium on Theory of Computing (STOC'2000)*, pages 171–180. ACM Press, 2000.

[Ada99] L. Adamic. "The Small World Web". URL: `http://www.parc.xerox.com/istl/groups/iea/www/smallworldpaper.html`, 1999.

[AFJM95] R. Armstrong, D. Frietag, T. Joachims, and T. Mitchell. "WebWatcher: A Learning Apprentice for the World Wide Web". In *Proceedings of the 1995 AAAI Spring Symposium of Information Gathing from Heterogeneous, Distributed Environments, Stanford, CA*, pages 6–12. AAAI Press, 1995.

[AKK98] J. Alspector, A. Kolcz, and N. Karunanithi. "Comparing Feature-Based and Clique-Based User Models for Movie Selection". In *Proceedings of the Third ACM Conference on Digital Libraries*, pages 11–18. ACM Press, 1998.

[ASBS00] L.N. Amaral, A. Scala, M. Bathelemy, and H.E. Stanley. "Classes of Behavior of Small-World Networks". *Proceedings of the National Academy of Science, USA*, Vol. 97:pp. 11149–11152, 2000.

[ASP97] M. Ackerman, B. Starr, and M. Pazzani. "The Do-I-Care Agent: Effective Social Discovery and Filtering on the Web". In *Proceedings of RIAO'97 (Computer-Assisted Information Searching on the Internet)*, pages 17–31, 1997.

[AT99] G. Adomavicius and A. Tuzhilin. "User Profiling in Personalization Applications Through Rule Discovery and Validation". In *KDD'99, Proceedings of the Fifth ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 377–381. ACM Press, 1999.

[ATar] G. Adomavicius and A. Tuzhilin. "Expert-Driven Validation of Rule-Based User Models in Personalization Applications". *Data Mining and Knowledge Discovery*, Vol. 5, 2001 (to appear).

[AWWY99] C. Aggarwal, J. Wolf, K. Wu, and P. Yu. "Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering". In *KDD'99, Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 201–212. ACM Press, 1999.

[AZ97] C. Avery and R. Zeckhauser. "Recommender Systems for Evaluating Computer Messages". *Communications of the ACM*, Vol. 40(3):pp. 88–89, March 1997.

[BA99] A.-L. Barabási and R. Albert. "Emergence of Scaling in Random Networks". *Science*, Vol. 286:pp. 509–512, October 1999.

[BC92] N. Belkin and B. Croft. "Information Filtering and Information Retrieval: Two Sides of the Same Coin?". *Communications of the ACM*, Vol. 35(12):pp. 29–38, December 1992.

[BHC98] C. Basu, H. Hirsh, and W. Cohen. "Recommendation as Classification: Using Social and Content-Based Information in Recommendation". In *Proceedings of the Fifteeth National Conference on Artifical Intelligence*, pages 714–720. AAAI Press, 1998.

[BHK98] J. Breese, D. Heckerman, and C. Kadie. "Empirical Analysis of Predictive Algorithms for Collaborative Filtering". In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 43–52. Morgan Kaufmann, 1998.

[BKM+99] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. "Graph Structure in the Web". In *Proceedings of the Ninth International World Wide Web Conference*, 1999.

[Bol85] B. Bollabas. *"Random Graphs"*. Academic, London, 1985.

[BP98a] D. Billsus and M. Pazzani. "Learning Collaborative Information Filters". In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 46–53. Morgan Kaufmann, 1998.

[BP98b] S. Brin and L. Page. "The Anatomy of a Large-Scale Hypertextual Web Search Engine". In *Proceedings of the Seventh International World Wide Web Conference*, pages 107–117. Elsevier Science, 1998.

[BS97] M. Balabanovic and Y. Shoham. "Fab: Content-Based, Collaborative Recommendation". *Communications of the ACM*, Vol. 40(3):pp. 66–72, March 1997.

[Bur99] R. Burke. "Integrating Knowledge-Based and Collaborative Filtering Recommender Systems". In *Proceedings of the Workshop on Artificial Intelligence for Electronic Commerce*, pages 69–72. AAAI Press, 1999.

[CBS92] J. Callen, C. Bruce, and H. Stephen. "The INQUERY Retrieval System". In *Proceedings of the Third International Conference on Database and Expert Systems Applications*, pages 78–83, 1992.

[CCH91] Y. Cai, N. Cercone, and J. Han. "Attribute-Oriented Induction in Relational Databases". In G. Piatetsky-Shapiro and W. Frawley, editors, *"Knowledge Discovery in Databases"*, pages 213–228. AAAI Press/MIT Press, 1st edition, 1991.

[CD00] J.M. Carlson and J. Doyle. "Highly Optimized Tolerance: A Mechanism for Power Laws in Designed Systems". Technical report, California Institute of Technology, 2000.

[CDRK$^+$99] S. Chakrabarti, B.E. Dom, S. Ravi Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Klienberg. "Mining the Web's Link Structure". *IEEE Computer*, Vol. 32(8):pp. 60–67, August 1999.

[Cen] IBM Almaden Research Center. "The CLEVER Project". URL: `http://www.almaden.ibm.com/cs/k53/clever.html`.

[CLR90] T. Cormen, C. Leiserson, and R. Rivest. *"Introduction to Algorithms"*. MIT Press, 1990.

[DIT98] J. Delgado, N. Ishii, and U. Tomoki. "Intelligent Collaborative Information Retrieval". In *Proceedings of the Sixth Ibero-American Conference on AI*, 1998.

[DR99] R. Driskill and J. Riedl. "Recommender Systems for E-Commerce: Challenges and Opportunities". In *Proceedings of the Workshop on Artificial Intelligence for Electronic Commerce*, pages 73–76. AAAI Press, 1999.

[ER59] P. Erdös and A. Renÿi. "On Random Graphs". *Publicationes Mathematicae*, Vol. 6:pp. 290–297, 1959.

[FFF99] M. Faloutsos, P. Faloutsos, and C. Faloutsos. "On Power-Law Relationships of the Internet Topology". In *Proceedings of the ACM SIGCOM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 251–262. ACM Press, 1999.

[FISS98] Y. Fruend, R. Iyer, R. Schapire, and Y. Singer. "An Efficient Boosting Algorithm for Combining Preferences". In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 170–178. Morgan Kaufmann, 1998.

[FLM98] D. Florescu, A. Levy, and A. Mendelzon. "Database Techniques for the World-Wide Web: A Survey". *SIGMOD Record*, Vol. 27(3):pp. 59–74, September 1998.

[Fol90] P. Foltz. "Using Latent Semantic Indexing for Information Filtering". In *Proceedings of the ACM Conference on Office Information Systems*, pages 40–47. ACM Press, 1990.

[Fon96] L. Foner. "A Multi-Agent Referral System for Matchmaking". In *Proceedings of the First International Conference on the Practical Applications of Intellignet Agent Technology*, pages 245–261, 1996.

[GNOT92] D. Goldberg, D. Nichols, B. Oki, and D. Terry. "Using Collaborative Filtering to Weave an Information Tapestry". *Communications of the ACM*, Vol. 35(12):pp. 61–70, December 1992.

[Gol] K. Goldberg. "Jester: The On-Line Joke Recommender". URL: `http://shadow.ieor.berkeley.edu/humor`.

[Gre] D. Greening. "Building Consumer Trust with Accurate Product Recommendations". URL: `http://ebusiness.macromedia.com/software/likeminds/whitepapers/building_consumer%20_trust%20.pdf`.

[GRGP00] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. "Eigentaste: A Constant Time Collaborative Filtering Algorithm". Technical Report M00/41, Electronic Research Laboratory, University of Berkeley, August 2000.

[GSK+99] N. Good, J. Scafer, J. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl. "Combining Collaborative Filtering with Personal Agents for Better Recommendations". In *Proceedings of the Sixteenth National Conference on Artifical Intelligence*, pages 439–446. AAAI Press, 1999.

[Han96] J. Han. "Data Mining Techniques". In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, page 545. ACM Press, 1996.

[HCM+00] D. Heckerman, D.M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. "Dependency Networks for Inference, Collaborative Filtering, and Data Visualization". *Journal of Machine Learning Research*, Vol. 1:pp. 49–75, 2000.

[Hea01] L. Heath. Personal Communication, 2001.

[HK70] E. Housman and E. Kaskela. "State of the Art in Selective Dissemination of Information". In *Proceedings of the IEEE Transaction on Engineering and Writing Speech*, pages 78–83, 1970.

[HKBR99] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. "An Algorithmic Framework for Performing Collaborative Filtering". In *Proceedings of the Twenty Second Annual International ACM SIGIR Conference*, pages 230–237. ACM Press, 1999.

[HP99] T. Hofmann and J. Puzicha. “Latent Class Models for Collaborative Filtering”. In *Proceedings of the 16th International Joint Conference on Artificial intelligence (IJCAI’99)*. IJCAI Press, 1999.

[HSRF95] W. Hill, L. Stead, M. Rosenstein, and G. Furnas. “Recommending and Evaluating Choices in a Virtual Community of Use”. In *Proceedings of the CHI’95-Human Factors in Computing Systems*, pages 194–201. ACM Press, 1995.

[JL00] F. Jiang and M.L. Littman. “Approximate Dimension Equalization in Vector-Based Information Retrieval”. In *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann, 2000.

[Kar] G. Karypis. “Family of Multilevel Partioning Algorithms”. URL: `http://www-users.cs.umn.edu/∼karypis/metis/metis.html`.

[Kau] H. Kautz. “ReferralWeb”. URL: `http://www.cs.washington.edu/homes/kautz/referralweb/`.

[KB96] B. Krulwich and C. Burkey. “Learning User Information Interests Through Extraction of Semantically Significant Phrases”. In *Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access*, pages 110–112. AAAI Press, 1996.

[Kel00] B. Keller. Personal Communication, 2000.

[KFV00] B. Kitts, D. Freed, and M. Vrieze. “Cross-Sell: A Fast Promotion-Tunable Customer-Item Recommendation Method Based on Conditional Independent Probabilities”. In *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining*, pages 437–446. ACM Press, 2000.

[KK95] G. Karypis and V. Kumar. “A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs”. Technical Report 95-035, Computer Science Department, University of Minnesota, 1995.

[Kle98] J. Kleinberg. “Authoritative Sources in a Hyperlinked Environment”. In *Proceedings of the Ninth ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677. ACM Press, 1998.

[Kle99] J. Kleinberg. “The Small-World Phenomenon: An Algorithmic Perspective”. Technical Report 99-1776, Cornell Computer Science, October 1999.

[KM99] A. Kohrs and B. Merialdo. “Clustering for Collaborative Filtering Applications”. In *Proceedings of Computational Intelligence for Modelling, Control and Automation*. IOS Press, 1999.

[KMM+97] J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordan, and J. Riedl. "GroupLens: Applying Collaborative Filtering to Usenet News". *Communications of the ACM*, Vol. 40(3):pp. 77–87, March 1997.

[Koh] A. Kohrs. "Active Web Museum Login". URL: `http://abyss.eurecom.fr:1111/AWM/login.html`.

[KSK97] T. Kamba, H. Sakagami, and Y. Koseki. "ANATAGONOMY: A Personalized Newspaper on the World Wide Web". *International Journal of Human-Computer Studies*, Vol. 46(6):pp. 789–803, 1997.

[KSS97] H. Kautz, B. Selman, and M. Shah. "ReferralWeb: Combining Social Networks and Collaborative Filtering". *Communications of the ACM*, Vol. 40(3):pp. 63–65, March 1997.

[Lee00] W. Lee. "Online Clustering for Collaborative Filtering". Technical Report TRA8/00, National University of Singapore, 2000.

[Lep] D. Leppik. "Genre-Based Partitions for a Movie Recommender". URL: `http://www-users.cs.umn.edu/~leppik/genre_partitions.html`.

[Lie95] H. Lieberman. "Letizia: An Agent that Assists Web Browsing". In *Proceedings of the Fourteenth International Joint Conference on Artifical Intelligence*, pages 924–929, 1995.

[Lik] Macromedia LikeMinds. "Macromedia LikeMinds Personalization and Performance". URL: `http://ebusiness.macromedia.com/software/likeminds/whitepapers/lm_white_3rd.pdf`.

[Lin00] W. Lin. *"Association Rule Mining for Collaborative Recommender Systems"*. PhD thesis, Worchester Polytechnic Institute, 2000.

[Lu00] L. Lu. "The Diameter of Random Massive Graphs". In *Proceedings of the Twelth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 912–921. ACM/SIAM Press, 2000.

[Mac67] J. MacQueen. "Some Methods for Classification and Analysis of Multivariate Observations". In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. University of California Press, 1967.

[MCS00] B. Mobasher, R. Cooley, and J. Srivastava. "Automatic Personalization Based on Web Usage Mining". *Communications of the ACM*, Vol. 43(8):pp. 142–151, August 2000.

[ME95] D. Maltz and K. Ehrlich. “Pointing the Way: Active Collaborative Filtering”. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI95)*. ACM Press, 1995.

[MPR00] U. Manber, A. Patel, and J. Robison. “The Business of Personalization: Experience with Personalization of Yahoo!”. *Communications of the ACM*, Vol. 43(8):pp. 35–39, September 2000.

[MR00] R. Mooney and L. Roy. “Content-Based Book Recommending Using Learning for Text Categorization ”. In *Proceedings of the Fifth ACM conference on Digital Libraries*, pages 195–204. ACM Press, 2000.

[MRK97] B. Miller, J. Riedl, and J. Konstan. “Experiences with GroupLens: Making Usenet Useful Again”. In *Proceedings of the 1997 Usenix Winter Technical Conference*, pages 219–231, 1997.

[NSW00] M. Newman, S. Strogatz, and D. Watts. “Random Graphs with Arbitrary Degree Distribution and their Applications”. Technical Report 0007042, Santa Fe Institute, 2000.

[OH99] M. O’Conner and J. Herlocker. “Clustering Items for Collaborative Filtering”. In *Proceedings of the ACM SIGIR Workshop on Recommender Systems*, 1999.

[Pay98] D. Payton. “Discovering Collaborators by Analyzing Trails through an Information Space”. In *Proceedings of the AAAI Fall Symopsium on Artificial Intelligence and Link Analysis*, pages 84–87. AAAI Press, 1998.

[PHLG00] D. Pennock, E. Horvitz, S. Lawrence, and C. Giles. “Collaborative Filtering by Personality Diagnosis: A Hybrid Memory- and Model-Based Approach”. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 473–480. Morgan Kaufmann, 2000.

[PMB96] M. Pazzani, J. Muramatsu, and D. Billsus. “Syskill & Webert: Identifying Interesting Web Sites”. In *Proceedings of the Thirteeth National Conference on Artifical Intelligence*, pages 54–61. AAAI Press, 1996.

[PR97] D. Peppers and M. Rogers. *“The One to One Future: Building Relationships One Customer at a Time”*. Bantam Doubleday Dell Publishing, 1997.

[Ram99] N. Ramakrishnan. “Experiences with an Algorithm Recommender System”. In P. Baudisch, editor, *“Proceedings of the CHI’99 Workshop on Interacting with Recommender Systems”*. ACM SIGCHI Press, 1999.

[Ram00] N. Ramakrishnan. “PIPE: Web Personalization by Partial Evaluation”. *IEEE Internet Computing*, Vol. 4(6):pp. 21–31, Nov/Dec 2000.

[RG99a] N. Ramakrishnan and A. Grama. "Data Mining: From Serendipity to Science". *IEEE Computer*, Vol. 32(8):pp. 34–37, August 1999.

[RG99b] D. Robalino and M. Gibney. "A Model of the Impact on Movie Demand of Social Networks and Word of Mouth Recommendation". URL: `http://www.metaculture.net/Searches/movies`, 1999.

[Ric98] L. Richard. "Automatic Information Extraction from Documents: A Tool for Intelligence and Law Enforcement Analysts". In *Proceedings of the AAAI Fall Symopsium on Artificial Intelligence and Link Analysis*, pages 63–67. AAAI Press, 1998.

[RIS$^+$94] P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom, and J. Riedl. "GroupLens: An Open Architecture for Collaborative Filtering of Netnews". In *Proceedings of the 1994 Computer Supported Collaborative Work Conference*, pages 175–186. ACM Press, 1994.

[RK] J. Riedl and J. Konstan. "GroupLens Research". URL: `http://www.cs.umn.edu/Research/GroupLens/`.

[RP97] J. Rucker and M. Polanco. "Siteseer: Personalized Navigation for the Web". *Communications of the ACM*, Vol. 40(3):pp. 73–76, March 1997.

[RS97] D. Rus and D. Subramanian. "Customizing Information Capture and Access". *ACM Transactions on Information Systems*, Vol. 15(1):pp. 67–101, 1997.

[RV97] P. Resnick and H. Varian. "Recommender Systems". *Communications of the ACM*, Vol. 40(3):pp. 56–58, March 1997.

[SC00] B. Smyth and P. Cotter. "Enabling Technologies: A Personalized Television Listings Service". *Communications of the ACM*, Vol. 43(8):pp. 107–111, August 2000.

[Sha94] U. Shardanand. *"Social Information Filtering for Music Recommendation"*. PhD thesis, Massachusetts Institute of Technology, 1994.

[SKB$^+$98] B. Sarwar, J. Konstan, A. Borchers, J. Herlocker, B. Miller, and J. Riedl. "Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System". In *Proceedings of ACM Conference on Computer Supported Cooperative Work, Sharing Information and Creating Meaning*, pages 345–354. ACM Press, 1998.

[SKKR00a] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. "Analysis of Recommendation Algorithms for E-Commerce". In *Proceedings of the Second ACM Conference on Electronic Commerce*, pages 158–167. ACM Press, 2000.

[SKKR00b] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. “Application of Dimensionality Reduction in Recommender System – A Case Study”. Technical Report CS-TR 00-043, Computer Science and Engineering Dept., University of Minnesota, July 2000.

[SKR99] J. Schafer, J. Konstan, and J. Riedl. “Recommender Systems in E-Commerce”. In *Proceedings of the ACM Conference on Electronic Commerce*, pages 158–166. ACM Press, 1999.

[SKRar] J. Schafer, J. Konstan, and J. Riedl. “Electronic Commerce Recommender Applications”. *Journal of Data Mining and Knowledge Discovery*, 2001 (to appear).

[SM83] G. Salton and M. McGill. *“Introduction to Modern Information Retrieval”*. McGraw-Hill, New York, 1983.

[SM95] U. Shardanand and P. Maes. “Social Information Filtering: Algorithms for Automating ‘Word of Mouth’”. In *Proceedings of CHI’95 – Human Factors in Computing Systems*, pages 210–217. ACM Press, 1995.

[SN99] I. Soboroff and C. Nicholas. “Combining Content and Collaboration in Text Filtering”. In *Proceedings of the IJCAI’99 Workshop on Machine Learning in Information FIltering*, pages 86–91, 1999.

[SS98] D. Swanson and N. Smalheiser. “Link Analysis of MedLine Titles as an Aid to Scientific Discovery”. In *Proceedings of the AAAI Fall Symopsium on Artificial Intelligence and Link Analysis*, pages 94–97. AAAI Press, 1998.

[SW93] M.F. Schwartz and D.C.M. Wood. “Discovering Shared Interests Using Graph Analysis”. *Communications of the ACM*, Vol. 36(8):pp. 78–89, August 1993.

[THA+97] L. Terveen, W. Hill, B. Amento, D. McDonald, and J. Creter. “PHOAKS: A System for Sharing Recommendations”. *Communications of the ACM*, Vol. 40(3):pp. 59–62, March 1997.

[Tri] Tripsmith. “Welcome to Trabble”. URL: `http://www.trabble.com/`.

[Tur] D. Turnbull. “Augmenting Information Seeking on the World Wide Web Using Collaborative Filtering Techniques”. URL: `http://donturn.fis.utoronto.ca/research/augmentis.html/`.

[UF98] L. Ungar and D. Foster. “Clustering Methods for Collaborative Filtering”. In *Proceedings of the Workshop on Recommendation Systems at the Fifteenth National Conference on Artificical Intelligence*, pages 112–125. AAAI Press, 1998.

[WF94] S. Wasserman and K. Faust. *“Social Network Analysis: Methods and Applications”*. Cambridge University Press, New York, 1994.

[WS98] D. Watts and S. Strogatz. “Collective Dynamics of ‘Small-World’ Networks”. *Nature*, Vol. 393(6):pp. 440–442, June 1998.

Appendix A

Pseudocode for Experiments

Given below is a skeleton of the procedures used to conduct our experiments. Statements enclosed in /* and */ are comments. We start with the adjacency matrix of people and movies A_R, and then generate the G_s and the G_r graphs (GENERATE-GRAPHS), for various values of hammock width w. The largest connected component of people, LC, in the G_s graph is determined. The largest component in the G_r graph is then obtained by attaching the movies to the nodes in LC and rendering the original edges bidirectional (this procedure is described in Chapter 3). The length of the two largest components is calculated. To determine the actual lengths of the G_s and G_r graphs (ACTUAL-LENGTHS), we used Floyd's and Djikstra's algorithms [CLR90]. To determine the estimates of these lengths by the NSW model, the average degree distributions of both G_s and G_r are determined and used as input to the formulas described in Chapter 3 (FORMULA-LENGTHS-G_s and FORMULA-LENGTHS-G_r respectively).

The MovieLens and the artificial EachMovie (GENERATE-DATASET) adjacency matrices are represented by M_R and E_R respectively. Similarly M_{Gs}, M_{Gr} and E_{Gs}, E_{Gr} represent the adjacency matrices of the social network and the recommender graphs for the MovieLens and EachMovie datasets respectively.

```
procedure MAIN ()
begin
    /* MovieLens experiments */
    $N_P \leftarrow 943$; $N_M \leftarrow 1623$;
    $M_R \leftarrow$ MovieLens dataset adjacency matrix of size $(N_P + N_M) * (N_P + N_M)$;
    for hammock width $w \leftarrow 1$ to 30 do
        $(M_{Gs}, M_{Gr}) \leftarrow$ GENERATE-GRAPHS $(M_R, N_P, N_M, w)$;
        $l_{pp}exp \leftarrow$ ACTUAL-LENGTHS ($M_{Gs}$);
        $l_{pp}form \leftarrow$ FORMULA-LENGTHS-$G_s$ ($M_{Gs}$);
        $l_r exp \leftarrow$ ACTUAL-LENGTHS ($M_{Gr}$);
        $l_r form \leftarrow$ FORMULA-LENGTHS-$G_r$ ($M_{Gr}$);
    end for
    /* EachMovie experiments */
    for minimum rating constraint $kappa \leftarrow 1$ to 15 do
        $N_P \leftarrow 500$; $N_M \leftarrow 75$;
        $\epsilon$ = LOOKUP ($kappa$);
        $E_R \leftarrow$ GENERATE-DATASET ($N_P, N_M, \epsilon$);
        for hammock width $w \leftarrow 1$ to 25 do
            $(E_{Gs}, E_{Gr}) \leftarrow$ GENERATE-GRAPHS ($E_R, N_P, N_M, w$);
            $l_{pp}exp \leftarrow$ ACTUAL-LENGTHS ($E_{Gs}$);
            $l_{pp}form \leftarrow$ FORMULA-LENGTHS-$G_s$ ($E_{Gs}$);
            $l_r exp \leftarrow$ ACTUAL-LENGTHS ($E_{Gr}$);
            $l_r form \leftarrow$ FORMULA-LENGTHS-$G_r$ ($E_{Gr}$);
        end for
    end for
end begin
```

```
procedure GENERATE-GRAPHS ($A_R, N_P, N_M, w$)
begin
    $A_R^2 \leftarrow A_R * A_R$;
    /* Entries in $A_R^2$ give us the number of paths of length 2 between two nodes */
    for rows $i \leftarrow 1$ to $N_P + N_M$ do
        for columns $j \leftarrow 1$ to $N_P + N_M$ do
            if $i \leq N_P$ then
                if $j \leq N_P$ then
                    if $A_R^2[i][j] \geq w$ and $i \neq j$ then
                $A_{Gs}[i][j] \leftarrow 1$;
                $A_{Gr}[i][j] \leftarrow 1$;
            **else**
                $A_{Gs}[i][j] \leftarrow 0$;
                $A_{Gr}[i][j] \leftarrow 0$;
            **end if**
        **else**
            $A_{Gr}[i][j] \leftarrow A_R[i][j]$;
        **end if**
    **else**
        $A_{Gr}[i][j] \leftarrow 0$;
    **end if**
  **end for**
**end for**
$LC \leftarrow$ LARGEST-COMPONENT ($A_{Gs}$);
**for** rows $i \leftarrow 1$ **to** $N_P$ **do**
    **if** person $i$ is not in $LC$ **then**
        **for** columns $j \leftarrow 1$ **to** $N_P + N_M$ **do**
            **if** $j \leq N_P$ **then**
                $A_{Gs}[i][j] \leftarrow 0$;
                $A_{Gr}[i][j] \leftarrow 0$;
            **else**
                $A_{Gr}[i][j] \leftarrow 0$;
            **end if**
        **end for**
    **end if**
**end for**
**return** ($A_{Gs}, A_{Gr}$);
```

```
end begin
```

```
procedure ACTUAL-LENGTHS ($A_G$)
begin
    Calculate $length$ of graph $A_G$ using Floyd/Djikstra algorithms;
    return $length$;
end begin
```

```
procedure FORMULA-LENGTHS-$G_s$ ($A_{Gs}$)
begin
    Find $z_1$ using Equation 3.3;
    Find $z_2$ using Equation 3.7;
    Find $length$ using Equation 3.12;
    return $length$;
end begin
```

```
procedure FORMULA-LENGTHS-$G_r$ ($A_{Gr}$)
begin
    Find $z_1$ using Equation 3.23;
    Find $z_2$ using Equation 3.24;
    Find $length$ using Equation 3.25;
    return $length$;
end begin
```

```
procedure GENERATE-DATASET ($N_P, N_M, \epsilon$)
begin
    for rows $i \leftarrow 1$ to $N_P$
        for columns $j \leftarrow 1$ to $N_M$ do
            if $j \leq N_M * i^{-\epsilon}$
                $E_R[i][j] \leftarrow 1$;
            else
                $E_R[i][j] \leftarrow 0$;
            end if
        end for
    end for
    /* Randomly rewire some movie endpoints */
    for rows $i \leftarrow 1$ to $N_P$
        for columns $j \leftarrow 1$ to $N_M$ do
            if $E_R[i][j] = 1$ then
                Generate a random number $r1$ between 1 and 10;
                if $r1 < 2$ then
                    Generate another random number $r2$ between 1 and 75;
                    $E_R[i][j] = 0$;
                    $E_R[i][r2] = 1$;
                end if
            end if
        end for
    end for
    Ensure $E_R$ is connected (if needed, change some edges manually);
    return $E_R$;
end begin
```

```
procedure LARGEST-COMPONENT ($A_{Gs}$)
begin
    Find the largest connected component of people $LC$;
    return $LC$;
end begin
```

```
procedure LOOKUP ($k$)
```

```
begin
    $epsilon[1] = 0.7$; $epsilon[2] = 0.6$; $epsilon[3] = 0.55$;
    $epsilon[4] = 0.5$; $epsilon[5] = 0.45$; $epsilon[6] = 0.42$;
    $epsilon[7] = 0.4$; $epsilon[8] = 0.38$; $epsilon[9] = 0.36$;
    $epsilon[10] = 0.34$; $epsilon[11] = 0.32$; $epsilon[12] = 0.3$;
    $epsilon[13] = 0.29$; $epsilon[14] = 0.28$; $epsilon[15] = 0.27$;
    return $epsilon[k]$;
end begin
```

Vita

Batul J. Mirza was born in 1977 in Hyderabad, India. She jumped around from one school to another within different states in India till finally her parents settled down in Kuwait. However Mr. Saddam Hussein had other plans for her and she landed right back in Hyderabad for another two years. Fate got her back to Kuwait and she completed her high school from Carmel School, Kuwait, in 1994 and obtained a Bachelor of Science degree in Computer Science in 1997 from Kuwait American Center of Education, an affiliate of Bharathiar Univeristy, Coimbatore, India. She joined Virginia Polytechnic Institute and State University in Spring 1999 and will complete her Master of Science degree in Computer Science and Applications in the Spring of 2001.