On the Limitations of Browsing Top-N Recommender Systems

Klaus Seyerlehner
Dept. of Computational Perception
Johannes Kepler University
Linz, Austria
klaus.seyerlehner@jku.at

Arthur Flexer
Austrian Research Institute for Artificial Intelligence
Vienna, Austria
arthur.flexer@ofai.at

Gerhard Widmer
Dept. of Computational Perception
Johannes Kepler University
Linz, Austria
gerhard.widmer@jku.at

ABSTRACT

To exploit the enormous potential of niche products, modern information systems must support users in exploring digital libraries and online catalogs. A straight-forward way of doing so is to support browsing the available items, which is in general realized by presenting a user the top-N recommendations for each item. However, recent research indicates that most of the niche products reside in the so-called *Long Tail*, and simple collaborative filtering-based recommender systems all alone do not allow to explore these niche products. Although there have been first attempts to extend collaborative filtering recommenders, e.g. by combining collaborative filtering with content-based techniques, we will show that it is not only a popularity problem related to the collaborative filtering approach that makes a portion of the elements of a digital library inaccessible via browsing, but also a consequence of the top N-recommendation approach itself.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search process, Selection process; G.2.2 [**Graph Theory**]: Graph algorithms

General Terms

Algorithms, Measurement, Theory

Keywords

recommender systems, browsing, long tail, reachability

1. INTRODUCTION

The importance of recommender systems specifically for digital libraries has increased dramatically in the last few years. This development is driven by the explosive growth of digital items stored in modern databases making millions of digital items like e-books, audio-books, music or videos available. Now that even the most obscure digital items are available the problem is to filter and present a user the *right* products. Chris Anderson has already foreseen, in his book [1], that the future of business will be selling less of more by exploiting the niche product market. Thus, the main question is how to assist users in finding exactly those niche products they are interested in.

One key concept to support people in finding new products is to give them the possibility to explore a digital library via *browsing*. There are several reasons why browsing will be one of the most important features of digital libraries in the future. First of all it is quite obvious that no single recommendation will ever contain the one *"correct"* item a user might be looking for, but it will in general take several recommendation steps and each step might point the user close to what he is looking for. Secondly, in many cases users even have no well-defined motive or search goal. Herocker et al. [9] discovered in a survey that many people use sites like *Amazone.com* when they have no purchase imminent, but they simply find it pleasant to browse, which of course makes the search process very unfocused. This seems to be especially true for digital libraries containing e-books, audio-books, music or videos, where many users are only looking for entertainment. In such a scenario users often have no precise idea of what they are looking for, but are just looking for some new unknown items. Interestingly the search or browsing process itself seems to be satisfying to many users. Last but not least flat-rate offers based on a monthly fee, make it even more attractive to explore online movie, music or e-book collections. Thus, many people simply want to explore *"what's out there"*, encouraged by the fact that no additional costs arise from testing and trying items of online catalogs.

Therefore we will — in contrast to many other papers dealing with recommender systems in terms of a single query only — interpret recommendation as a continuous process, where a user navigates from recommendation to recommendation, slowly refining her search goal. Such a process can then be described by the sequence of recommendations that a user steps through, and whenever a user is exploring a database via such a sequence of recommendations we speak of *browsing*.

The easiest and of course most widespread way to support browsing is to present a user the list of the top-N recommendations for each item in a library such that she can

browse through the whole library. Systems that generate the top-N recommendations for each item are called *top-N recommender systems* [11, 13]. The goal of this paper is to analyze the ability of top-N recommender systems to assist users in exploring digital libraries via browsing. The ability of a top-N recommender system to be useful in exploring a collection cannot be measured by accuracy-based evaluation criteria only; other criteria beyond accuracy will be necessary. In this paper we will focus on one of these evaluation criteria, by investigating the *catalog coverage* via complex network analysis.

The rest of the paper is structured as follows: In the next section we will discuss some related work and the related *Long Tail* phenomenon. In section 3 we will analyze a real world content-based recommender system and a very simple collaborative filtering based recommender system. We will experimentally show for both types of recommender systems that a non-negligible portion of the items in the database is not accessible when browsing the top-N recommendations only. In section 4, finally, we prove, using a random graph model, that via browsing the top-N recommendations always a certain percentage of all items will be inaccessible — unless the recommender systems is especially designed. Finally in section 5, wegive an outlook on future research directions.

2. RELATED WORK

To exploit the potential associated with the sale of niche products, recommender systems will have to help users in finding the hidden products within the item catalog by providing recommendations to those niche items. However, recent research on recommender systems reveals that the very popular strategy of generating recommendations based on collaborative filtering is not well suited to this [5, 6]. The main reason is that such recommender systems follow a popularity rule, recommending the best selling product to all users, whereas unpopular or newly introduced items having just a few or no ratings are not or wrongly recommended because of the lack of reliable ratings. The typical popularity distribution of such an item-catalog (e.g. measured in item-purchases or item-downloads) is in general characterized by a head containing the few popular items and a long tail consisting of all unknown or unpopular items. This specific pattern motivated the phrase *Long Tail* [3, 1, 4]. Exploiting the niche product market essentially means selling those items sitting in the Long Tail. Recent research has been focusing on how to extend recommender systems to also recommend niche products. There seem to be two main reasons why niche products can stay hidden in the Long Tail:

- The recommendations for a niche product are bad. Because of the lack of reliable information about a niche product, the niche product is recommended in a wrong context, where users are most likely not interested in such a product.
- A niche product is hardly or not at all accessible via recommendation. It cannot be reached by any sequence of recommendations, but only by a direct query, which is of course unlikely because it is a niche product and not well-known.

To alleviate the Long Tail problem, Park et al. [17] have already addressed the former reason. They tried to improve the recommendation quality of niche products by clustering the items in the tail part and then generating recommendations for the tail items based on the ratings of the corresponding cluster.

In contrast, we will focus on the latter reason and try to identify items that are not reachable via recommendation. The number of items that are not reachable can be interpreted as a quality measure related to the so-called *catalog coverage* [9]. More specifically, *Coverage* measures the percentage of a dataset that the recommender system is able to provide recommendations for, whereas *Catalog coverage* measures what percentage of items a recommender system ever recommends to users. In practice catalog coverage is usually measured on a set of recommendations formed at a single point in time. For instance, it might be measured by taking the union of the top 10 recommendations for each user in the population. In this paper we will however follow the approach of Celma et al. [5, 6] and try to analyze the network structure of top-N item-based recommender systems by identifying the number of unreachable items in a recommendation network, which can be considered the inverse of the catalog coverage. Celma et al. mainly focused on the popularity bias of recommender systems by comparing three different recommendation networks, one based on collaborative filtering, a content-based system and an expert-based recommendation network. They found that the collaborative filtering approach is prone to popularity bias, whereas content-based and expert-based network are not. This finding supports the idea of hybrid recommender systems. Donaldson [7] proposed combining a collaborative filtering strategy with a content-based strategy in the domain of music recommendation, which might help make items in the Long Tail accessible since content-based recommender systems are not prone to the popularity problem. There is also some related work on the network analysis of content-based recommender systems, especially in music information retrieval. Aucouturier [2] found that some content-based similarity networks based on timbral similarity resemble scale-free networks and might contain extreme hubs (i.e., items that are among the N nearest neighbors of a large number of other items). However, recent research on content-based similarity [19, 10] reveals that these findings do not generalize to all content-based (audio) similarity measures, which we will further investigate in section 3.1.

In the following, we will present an alternative approach to analyzing catalog coverage, based on complex network analysis.

3. REACHABILITY: AN EMPIRICAL STUDY

To measure the catalog coverage of item-based top-N recommender systems we analyze the recommendation network of the respective recommender system. Any item-based top-N recommender system can be transformed into an equivalent *recommendation network* or *recommendation graph*. Consider a recommender system presenting a user a list of precisely l recommendations for each item in the database. Then the recommendation graph of this recommender system can be represented as a directed graph $G = (V, E)$, where each vertex $v \in V$ in the graph corresponds to a item

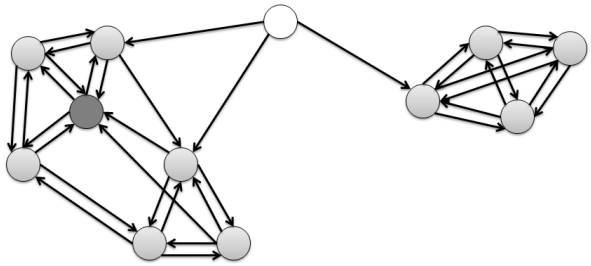

Figure 1: *The recommendation graph of a top-N item-based recommender. Each vertex has exactly l outgoing edges. In this example $l = 3$. The number of incoming edges is an emergent property of the network. There might be hubs, i.e., vertices with a high in-degree (dark), and sources, i.e., vertices with an in-degree of zero (white).*

in the database. Furthermore each vertex v in this graph has exactly l outgoing edges pointing to those items in the database that are recommended for the specific item v. Thus each vertex v has an out-degree of l ($\deg^+(v) = l$). Figure 1 visualizes such a directed recommendation graph.

The graph representation makes it easy to identify items that are not reachable via recommendation. Obviously any vertex v having an in-degree of zero ($\deg^-(v) = 0$) corresponds to an item that is not reachable via recommendation. In graph theory a vertex having an in-degree of 0 is called a *source*[1]. Vertices with a very high in-degree are called *hubs*. In the graph in figure 1 there is an example of both, a *hub* and a *source*. We call items corresponding to sources *source-items*. The number of sources in a recommendation graph is an inverse measure of the catalog coverage, as it measures the number of items that are not reachable, while catalog coverage informs about the number of items that are ever recommended by a recommender system.

In the literature it is quite common to categorize recommender systems according to the strategy used to generate the recommendations. The most prominent classes are *collaborative-filtering* and *content-based* systems. In the following we will analyze the graph structure of a top-N recommender systems of each of these two categories and report on the number of sources for both types of recommender systems.

3.1 Content-based Techniques

As an example of a content-based recommender system we analyze a real world music recommender attached to an Austrian music portal. The FM4 Soundpark[2] is an internet platform of the Austrian public radio station FM4, that allows artists to present their music to the general public. All interested parties can download the music free of any charge. At the moment this music collection contains about 10000 songs and is steadily growing. At the time of our experiments, 7665 songs were available.

[1]Identifying sources can be done in linear time ($O(n)$).
[2]`http://fm4.orf.at/soundpark/main`

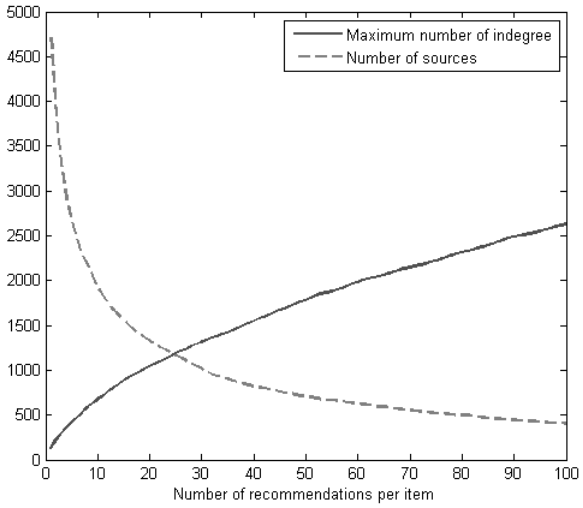

Figure 2: *Analysis of the FM4 Soundpark. For small recommendation lists, the number of sources is extremely large. It decreases with an increasing number of recommendations per query, whereas the maximum in-degree over all vertices in each graph increases. For a result set size of 100, there is one song that appears in the recommendation list of 2628 other songs, or in 34.29% of all recommendation lists.*

The recommender system attached to the FM4 Soundpark music portal is based on a standard similarity measure for music audio files. Each song is modeled as a distribution of local spectral features, namely Mel Frequency Cepstrum Coefficients (MFCCs). MFCCs are a compact representation of the spectral envelope of a short audio frame and are one of the most widespread features used in the Music Information Retrieval (MIR) community. A single multivariate Gaussian distribution is used to model the distribution of MFCCs of a song. Recommendations can then be generated by comparing these distributions. This is commonly done by computing the Kullback-Leibler (KL) divergence [12] or relative entropy between the distributions of two songs. For more details on the feature extraction process and the generation of music recommendations we refer to [14, 15]. Using the recommender system of the FM4 Soundpark we can generate lists of recommended songs of a given length l, ordered according to the similarity to the query song.

Consequently we systematically created various recommendation graphs for $l = 1 \ldots 100$, where l is the number of recommended songs and defines the number of outgoing edges for each vertex in the recommendation graph. For each of these graphs we counted the number of sources and also identified the *"biggest"* hub, the vertex with the maximum in-degree in the graph. Figure 2 shows that for short recommendation lists the number of sources is extremely high. For example, in the recommendation graph of degree 5 (a recommendation graph where each vertex has 5 outgoing edges) there are 2661 sources, which implies that 34.72% of all the songs in the music collection are never recommended at all. By increasing the length of the recommendation list

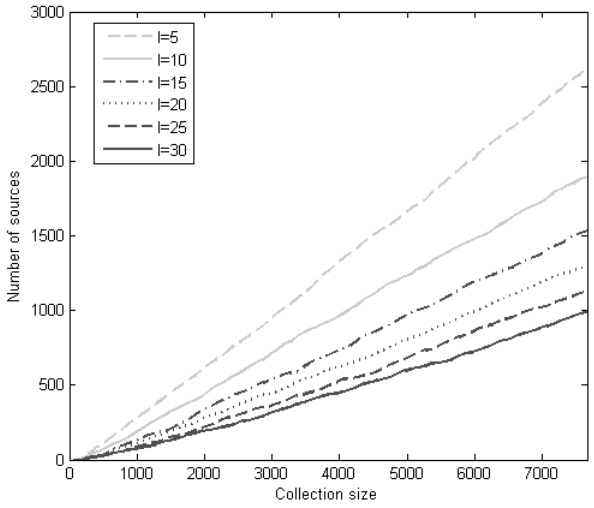

Figure 3: *The Analysis of the FM4 Soundpark reveals that the number of sources scales with the database size. Furthermore the number of sources depends on the length of the recommendation lists for each item. This is illustrated for fixed recommendation list length of* $l = 5, 10, 15, 20, 25, 30$.

the number of sources decreases, but even if we recommend the 20 most similar songs according to the similarity measure, there are still approximately 1320 sources in the resulting recommendation graph. Consequently still 17.22% all the songs in the collection are inaccessible. From figure 3 we can see how the number of sources scales with the collection size. To simulate different collection sizes songs were randomly removed from the collection. Figure 3 illustrates that the problem gets worse for increasing collection sizes, but fortunately in a linear way. In fact the analysis of the recommendation graph that corresponds to the online version of the FM4 Soundpark — there are only three recommendations per song — revealed that only 56,79% of all songs are reachable by recommendations, the remaining 43,21% of the songs are sources and are never recommended.

In addition to the number of sources, we also computed the maximum in-degree over all vertices in each graph, visible in figure 2. It seems that extreme hubs and the number of observed sources in top-N recommendation graphs are related phenomena. Extreme hubs imply that there are many sources and vice versa. A very informal but intuitive explanation is that hub-vertices are some sort of attractors "stealing" the links from other vertices, which might even lead to a situation where a vertex has no incoming links at all. This phenomenon, known as the hub-problem in the MIR community, was first discovered by Aucouturier [2]. Hub-songs are songs which are similar to very many other songs in the collection according to the similarity measure, while they do not share any perceptual similarity. Recent research indicates that the hub problem might be related to the similarity measure used to compute song similarities, and that there are methods which are not or not that much affected by this problem. Therefore, we evaluated the recommendations generated by an alternative audio similarity measure, namely the vector quantization approach described in [19]. Figures 4 and 5 visualize the number of sources, produced by the Soundparks's Gaussian measure and our alternative method, respectively. It can be seen from figures 4 and 5 that the vector quantization approach also generates sources, but far fewer than the single Gaussian approach. Consequently, based on the results of this experiment, we can conclude that the number of sources depends on how the similarities between database items are estimated. This raises the question if other completely different ways of estimating item-to-item similarities e.g. collaborative filtering would probably produce no sources at all. In the next subsection we will analyze a simple collaborative filtering based recommender to find out if such a system will also generate sources.

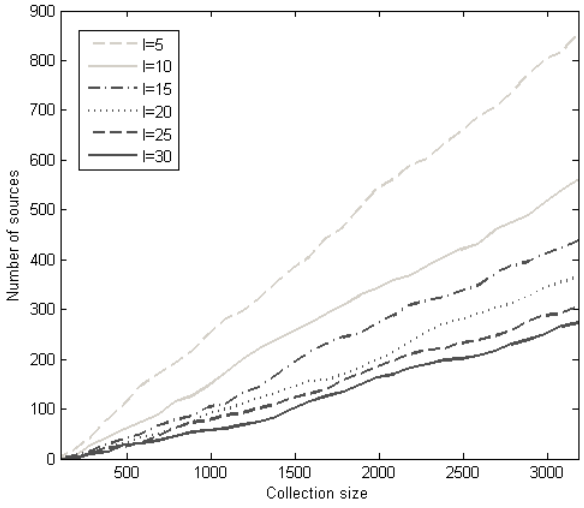

Figure 4: *The network analysis of the recommendation algorithm of the FM4 Soundpark based on a genre classification dataset consisting of 3180 songs. Results are shown for different collection sizes and for fixed recommendation list length of* $l = 5, 10, 15, 20, 25, 30$. *Compare these results to those obtained on the same dataset for the vector quantization approach (figure 5).*

3.2 Collaborative Filtering

Our prototype for the 'collaborative filtering-based recommender' is based on the well-known Movielens dataset[3], which contains 1 million ratings for 3900 movies by 6040 users. The recommendation approach that we simulate is based on the approach presented in [18]. In this system two items are thought of as two vectors in the m dimensional user space, where m is the number of users. The cosine distance is then used as a similarity measure between the n items. Given the $n \times m$ ratings matrix the similarity between two item vectors is given by equation (1), where "." denotes the dot-product of the two vectors.

$$\text{sim}(i, j) = \cos(\vec{i}, \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{||\vec{i}|| \; ||\vec{j}||}, \qquad (1)$$

[3]www.grouplens.org

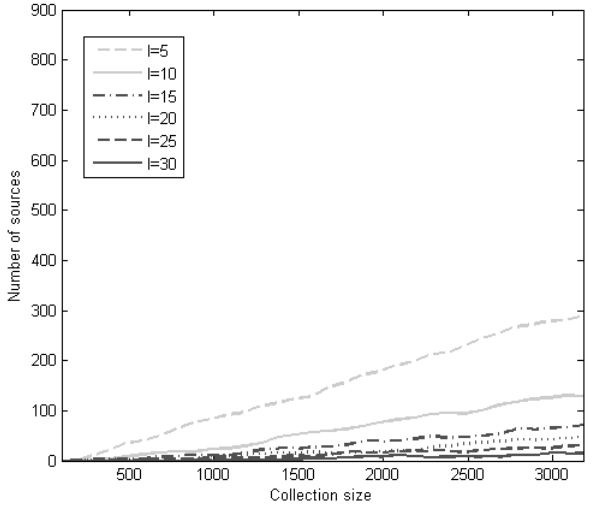

Figure 5: *The network analysis of a vector quantization approach. Results are shown for varying collection sizes and for fixed recommendation list length of $l = 5, 10, 15, 20, 25, 30$ for the same genre classification dataset as in figure 4. The vector quantization approach generates fewer sources-items.*

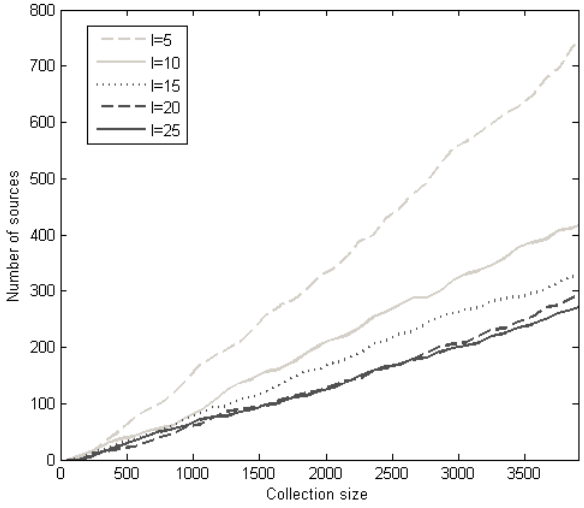

Figure 6: *Number of sources for a simple collaborative recommender system based on the Movielens dataset.*

Our experimental recommender system then simply proposes the top-N most similar items according to the cosine distance. Even if this is surely not a high quality recommender system, it will be useful as we are not striving for improved classification accuracy. Our sole goal is to figure out if we can also identify sources within this collaborative-filtering based recommendation graph, and thus learn something about catalog coverage.

As we did for the content-based approach, we simulated different collection sizes by randomly removing items from the dataset, before generating recommendations. Additionally we also generated recommendation graphs for recommendation lists of different length as we did before. The results of the analysis with respect to the number of sources are visualized in figure 6. Obviously collaborative filtering based recommender systems suffer from the same problem as content-based recommender systems do. We can observe many sources in the generated recommendation graphs, which implies that a significant portion of the movies is never recommended by this experimental recommender system. For example 19.23%, of all movies will never be recommended when browsing the top-5 recommendations only.

All our experiments indicate that there are sources independently of how we generate the recommendations. How many sources we observe seems to depend on two factors. First of all different recommendation strategies generate a different number of sources, as demonstrated for two content-based systems on one and the same dataset (see section 3.1). Secondly, the collection size and the length of the recommendation lists seem to have a major influence on the number of source. In the next section we propose to make use of a random graph model to investigate the theoretic limitations of browsing top-N recommender systems.

4. REACHABILITY: A THEORETICAL VIEW

In the last section we analyzed two top-N recommender systems based on two fundamentally different recommendation strategies and found that a significant portion of all items in the respective library is not reachable via browsing the generated recommendations. Obviously the way we estimate the item-to-item similarities determines the edges between the vertices in the graph and consequently determines the number of sources as well. Hence, a similarity function that is completely unbiased (with no preference of linking specific items) will produce the fewest sources in a recommendation graph. Clearly a random recommender is a realization of such a "fair" similarity function and represents a lower bound with respect to the number of sources. If even a random top-N recommender systems generates sources, then any top-N recommender system will generate sources. Therefore we will now analyze the recommendation graph of a random recommender.

Random graphs have been intensively studied in complex network analysis [16]. There exist a variety of random graph models. The most prominent one is the Poisson random graph independently proposed by Erdős and Rényi [8] and Solomonoff and Rapoport [20]. The graph model we study in this paper differs from this standard graph model in that the graph is directed and in that we assume a fixed out-degree for each vertex. Consider a directed graph $G = (V, E)$, where each vertex $v \in V$ points to precisely l other items ($\deg^+(v) = l$). Furthermore, assume that the recommended items are chosen randomly. Thus each item in the recommendation graph will point to l other random items, but not to itself. Suppose that there are N items in total in the database. Then the recommendation graph will have N vertices and lN random edges. The probability of an item in this graph to be referenced by exactly k other items is

then given by the binomial distribution:

$$P(X = k) = \binom{l(N-1)}{k} p^k (1-p)^{l(N-1)-k}, \qquad (2)$$

where $p = \frac{1}{N-1}$ as each item will equally likely point to any other item, but will not reference itself. The probability of not being referenced by any other item in the graph is given by equation (3).

$$P(X = 0) = (1 - \frac{1}{N-1})^{l(N-1)} \qquad (3)$$

Knowing the probability for a single item not to be referenced by any other item, we can now estimate the *expected number of sources* $E(X)$ (items that are not referenced) for the whole system containing N items.

$$E(X) = NP(X = 0) = N(1 - \frac{1}{N-1})^{l(N-1)} \qquad (4)$$

To validate our model, we created a random recommender and computed the number of sources for this particular random recommender. In figure 7 the number of sources of this specific random recommender is plotted against the expected number of sources from the model. The model perfectly fits the observed numbers. It essentially depends on the two parameters l and N. The number of sources seem to linearly increase with the collection size (see figure 7). Depending on l, the number of sources grows more or less slowly. If N approaches infinity ($N \to \infty$) the expected number of sources ($E(X) \to \infty$) approaches infinity as well. Deriving the limit, for N approaching infinity, of the probability for an item to be a source-item (equation 3), we find that the probability converges:

$$\lim_{N\to\infty} P(X = 0) = \lim_{N\to\infty} (1 - \frac{1}{N-1})^{l(N-1)} = e^{-l} \qquad (5)$$

In combination with equation 4 this implies that the percentage of sources will stay constant and that the number of sources increases linearly with the database size. More importantly the percentage of sources exponentially decays with the length of the recommendation lists. Thus it is in general desirable to have long recommendation lists as this implies high catalog coverage. In practice, however, the number of recommended items is constrained by usability reasons and is typically rather low (usually below ten). Furthermore the random recommender model is optimal in the sense that it will create the fewest sources that one can expect. For real world recommender systems the percentage of items inaccessible via recommendation can be much higher. The main point of our theoretical result is that even in the best case, and independent of the recommendation approach, there will always be a percentage of items that will be inaccessible when browsing a top-N recommender system — unless it is specifically designed (see below).

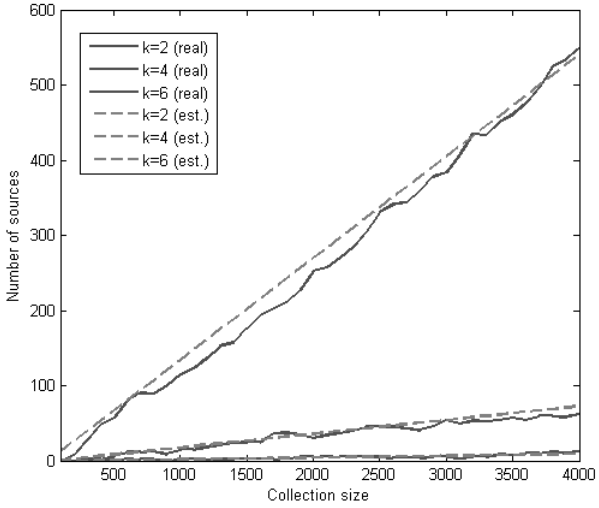

Figure 7: *Number of sources observed for a random recommender and the predictions from the proposed model.*

5. CONCLUSIONS AND FUTURE WORK

In this paper we have analyzed item-based top-N recommender systems with respect to catalog coverage. Experiments with two recommender types revealed that independent of the recommendation approach many items of the respective catalog are not recommended at all, which implies that they are not reachable by recommendation-based browsing. Therefore just using the top-N recommendations to support browsing is not an ideal solution. To improve the browsability it seems promising to take a more holistic approach that also considers the structure of the whole recommendation network. To guarantee important network properties like, e.g., reachability or connectedness, the recommendations for an item cannot be totally independent of all other recommendations, but have to be coordinated in such a way that on the one side the recommendation accuracy is optimized and on the other side essential network properties, e.g. reachability, are ensured. Recommender systems should be especially designed to optimize both accuracy and browsability. Another insight from our investigations is that sources are not a consequence of a specific recommendation strategy, but are a consequence of the top-N recommendation approach itself.

Future research directions will include the development of algorithms that modify or generate recommendation graphs that simultaneously optimize recommendation accuracy and desirable network properties. Another interesting research direction will be to study *recommendation novelty*, by analyzing browsing sequences within recommendation networks. For instance, when a user is browsing by moving from recommendation to recommendation, what is the overlap of two consecutive recommendation lists — how many novel items are proposed from one step to the next on average, and how likely is it to get stuck in a subpart of the recommendation graph?

Acknowledgments

This research was supported by the Austrian Research Fund (FWF) under grant L511-N15, and by the Austrian Research Promotion Agency (FFG) under project number 815474-BRIDGE.

6. REFERENCES

[1] C. Anderson. *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion, July 2006.

[2] J.-J. Aucouturier and F. Pachet. A scale-free distribution of false positives for a large class of audio similarity measures. *Pattern Recogn.*, 41(1):272–284, 2008.

[3] E. Brynjolfsson, Y. J. Hu, and M. D. Smith. Consumer surplus in the digital economy: Estimating the value of increased product variety at online booksellers. *Management Science*, 49(11), 2003.

[4] E. Brynjolfsson, Y. J. Hu, and M. D. Smith. From niches to riches: Anatomy of the long tail. *Sloan Management Review*, 47(4):67–71, 2006.

[5] O. Celma and P. Cano. From hits to niches? or how popular artists can bias music recommendation and discovery. In *2nd Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition (ACM KDD)*, Las Vegas, USA, August 2008.

[6] P. Celma, O.;Herrera. A new approach to evaluating novel recommendations. In *2008 ACM Conference on Recommender Systems*, Lausanne, Switzerland, 23/10/2008 2008.

[7] J. Donaldson. A hybrid social-acoustic recommendation system for popular music. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 187–190, New York, NY, USA, 2007. ACM.

[8] P. Erdős and A. Rényi. On random graphs. I. *Publ. Math. Debrecen*, 6:290–297, 1959.

[9] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, January 2004.

[10] M. Hoffman, D. Blei, and P. Cook. Content-based musical similarity computation using the hierarchical dirichlet process. In *In Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR'08)*, pages 349–354, Philadelphia, USA, Sept. 14-18, 2008.

[11] G. Karypis. Evaluation of item-based top-n recommendation algorithms. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 247–254, New York, NY, USA, 2001. ACM.

[12] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[13] S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 393–402, New York, NY, USA, 2004. ACM Press.

[14] M. Levy and M. Sandler. Lightweight measures for timbral similarity of musical audio. In *AMCMM '06: Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, pages 27–36, Santa Barbara, California, USA, 2006.

[15] M. Mandel and D. Ellis. Song-level features and svms for music classification. In *In Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR'05*, London, UK, 11-15th September 2005.

[16] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[17] Y.-J. Park and A. Tuzhilin. The long tail of recommender systems and how to leverage it. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, pages 11–18, New York, NY, USA, 2008. ACM.

[18] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 285–295, New York, NY, USA, 2001. ACM.

[19] K. Seyerlehner, G. Widmer, and P. Knees. Frame level audio similarity - a codebook approach. In *In Proceedings of the 11th International Conference on Digital Audio Effects (DAFx'08)*, pages 349–354, Espoo, Finland, Sept. 1-4, 2008.

[20] R. Solomonoff and A. Rapoport. Connectivity of random nets. *Bulletin of Mathematical Biology*, 13(2):107–117, June 1951.