Probabilistic Collaborative Filtering with Negative Cross Entropy

Alejandro Bellogín[a,c], Javier Parapar[b], Pablo Castells[c]
[a]Information Access, Centrum Wiskunde & Informatica
[b]Information Retrieval Lab, Department of Computer Science, University of A Coruña
[c]Information Retrieval Group, Department of Computer Science, Universidad Autónoma de Madrid
alejandro.bellogin@cwi.nl, javierparapar@udc.es, pablo.castells@uam.es

ABSTRACT

Relevance-Based Language Models are an effective IR approach which explicitly introduces the concept of relevance in the statistical Language Modelling framework of Information Retrieval. These models have shown to achieve state-of-the-art retrieval performance in the pseudo relevance feedback task. In this paper we propose a novel adaptation of this language modeling approach to rating-based Collaborative Filtering. In a memory-based approach, we apply the model to the formation of user neighbourhoods, and the generation of recommendations based on such neighbourhoods. We report experimental results where our method outperforms other standard memory-based algorithms in terms of ranking precision.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Information Search and Filtering

General Terms: Algorithms, Experimentation, Performance

Keywords: Recommender systems, Collaborative Filtering, Relevance Models

1. INTRODUCTION AND MOTIVATION

Personalised recommendation is a fertile research area with roots in the eighties, which started to attract wider attention in the mid-nineties when the first works on Collaborative Filtering came to light [6]. Three classical approaches to recommendation are distinguished: *content-based recommendation* (CB), based on the user's history and data (descriptions, content, features) of the items; *collaborative filtering* (CF), based on the history of similar users and items; and *hybrid approaches*, based on combining CB and CF recommendation. One of the earliest and most popular CF methods is the so-called k nearest-neighbours (kNN) approach, which is still being used today in many commercial systems.

A common formulation of a user-based kNN method is [6]:

$$\hat{r}(u,i) = C \sum_{v \in N_k(u)} \mathrm{sim}(u,v) r(v,i) \tag{1}$$

where the known preference of a user u for a particular item i is given by a numeric rating $r(u,i)$, and $\hat{r}(u,i)$ denotes the system's estimate of this value for an item for which the degree of user interest is unknown to the system. To provide that estimation, the method takes into account the ratings $r(v,i)$ provided by the k users v who are most *similar* to u, commonly referred to as neighbours and denoted here as $N_k(u)$. The function $\mathrm{sim}(u,v)$ measures the similarity between two users u and v, and the constant C is a normalisation factor. Thus, the predicted rating of user u for item i is computed as a weighted average of u's neighbours' ratings (deviations with respect to the user and neighbour average can also be considered [6]), where the weights are the similarities between u and her neighbours v.

Relevance-Based Language Models [3] (or Relevance Models for short, RM), on the other hand, are among the best-performing ranking techniques in text retrieval. In this paper, we study how RM can be adapted to rating-based recommendation, and whether they may lead to enhancements in terms of ranking-based metrics such as precision and nDCG. We do so by establishing a mapping between the variables involved in RM and the ones in user-based CF. The mapping is non-trivial as, to begin with, RM is formulated in text IR on a triadic space (query, documents, words), whereas the CF space is typically dyadic (users and items). Based on a proposed mapping, we bring the adaptation to computable terms, giving rise to a workable and effective recommendation framework. The resulting approach comprises two separable subcomponents: a neighbour selection approach and a ranking function, which can be used on their own or together.

In the remaining of this paper, we will answer the following research questions: (**RQ1**) Are the Relevance-Based Language Models useful to identify neighbours in recommendation? (**RQ2**) Is there any advantage in using a complete probabilistic representation of the problem? In other terms, does it make sense to replace the CF weighted average by the RM cross entropy as used in IR, and does this approach work well in practice?

After presenting our proposed method in detail in the next section, we address the research questions experimentally on MovieLens data, testing the effectiveness of our overall approach, and comparing the separate effect of the two subcomponents. As we shall see, the empirical results validate the approach, showing performance improvements over state of the art memory-based alternatives.

2. RELEVANCE MODELLING FOR RECOMMENDATION

Relevance-Based Language Models, as first proposed in [3], view the original user search query Q as a short sample of words obtained from an underlying relevance model R. If one aims to add more words from R to the query then it is reasonable to choose those words with the highest estimated probability given a sample of observed words generated by the relevance model for the query. In [3] two different estimations for RM were originally presented, namely RM1 and RM2 from which, in our present approach, we adopt the former. In RM1 it is assumed that the query words $q_i \in Q$ and the words w in the relevant documents are sampled identically and independently from a unigram distribution (*i.i.d. sampling*).

RecSys'13, October 12–16, 2013, Hong Kong, China.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2409-0/13/10 ...$15.00.
http://dx.doi.org/10.1145/2507157.2507191.

RM for Retrieval	RM for Recommendation
query q	target user u
query words $q_1 \dots q_n$	items rated by user $I(u)$
pseudo relevance set PRS	positively rated items $PRS(u)$
candidate expansion terms	candidate user's neighbours

Figure 1: Correspondence between the elements involved in Relevance-Based Language Models for document retrieval and its adaptation to item recommendation

Denoting by R_Q the underlying relevance model from which the words in the query Q were sampled, the relevance language model (i.e. the distribution of words given R_Q) is computed as in Eq. 2:

$$p(w|R_Q) \propto \sum_{d\in\mathcal{C}} p(d)p(w|d) \prod_{i=1}^{|Q|} p(q_i|d) \qquad (2)$$

where $\mathcal{C}$ is the set of all documents in the search space (the collection). For computational reasons, the top N documents from the initial result set are taken for the estimation, rather than the whole collection $\mathcal{C}$. This set of documents is usually called pseudo relevance set (PRS). Finally, to build the final (expanded) query, the terms with the highest estimated probabilities of relevance with respect to the query are selected, and a second document ranking is produced using the negative cross entropy retrieval function.

2.1 Space Mapping

In order to adapt this framework to the recommendation task, we need to identify the random variables to play the part of queries, documents and words in the above formulation. Our proposed mapping equates, at the top level, users to queries and items to the documents to be retrieved (see Figure 1). Now, words are equated to users or to items, depending on the role they are playing: in assimilating a user u to a query Q, the constituent elements of the latter (the query terms $q_1 \dots q_n$) are mapped to the set of items rated by u – which we shall denote as $I(u)$. However, the words with which queries are expanded are mapped to other users, with similar tastes to the target user, as we shall see. Items as documents play an additional part: that of pseudo-relevant documents. As we just recalled, in RM instead of estimating the Relevance Model R_Q over the whole collection, only a certain top set of documents is commonly used for such task: the pseudo relevant set PRS. In our proposed approach this role is played by the items positively rated by the target user u, which we denote by $PRS(u)$.

Note this last aspect of our approach introduces an interesting difference with respect to RM in IR: while an initial ranking is needed in the IR formulation to select the pseudo relevant documents from, in our adaptation PRS is extracted from readily available data without needing to compute an initial recommendation. This means the resulting approach works as a standalone recommendation algorithm, rather than a query expansion technique.

Other mappings are possible besides the one we investigate here, e.g. words for expansion could be mapped to items instead of users. We leave this possibility as future work. Ratings could also be used as a first-class variable making up a triadic space, apparently easier to match to the IR framework. We think however ratings would be a bad – or at least unpredictable – analogue for words (let alone the query or documents), since ratings are rather a natural equivalent of relevance grades in IR. Log-based recommendation, on the other hand, has a more direct equivalence [8], which is however not applicable to rating data.

The proposed approach allows to decompose the task as follows: a) we compute a relevance model for each user in order to capture how relevant any other user would be as a potential neighbour; and then b) we replace the rating prediction by weighted average in CF with the well-established scoring function (negative cross entropy) used in IR to incorporate the information learnt from the RM. We describe these two parts of our approach in the next two sections.

2.2 Neighbour Selection

The equivalent estimation to RM1 for neighbour selection according to our proposed mapping goes as follows. Modelled after RM1, the probability of a neighbour v under the relevance model R_u for a given user u is estimated as:

$$p(v|R_u) = \sum_{i\in PRS(u)} p(i)p(v|i) \prod_{j\in I(u)} p(j|i) \qquad (3)$$

where $p(i)$ is the probability of the item i in the collection, $p(v|i)$ is the probability of the neighbour v given the item i, and $p(i|j)$ is the conditional probability of item i given another item j. As presented in Figure 1, $I(u)$ corresponds to the set of items rated by user u, and $PRS(u) \subset I(u)$ is the subset of items rated by u above some specific threshold, i.e. items with a favorable rating value indicating the user likes them.

Besides folding a triadic space into a dyadic one, in our formulation the probabilistic RM framework is turned upside down to some extent. The ground probabilistic model upon which RM are formulated in text IR reflects a process in which words are sampled according to a certain (indirectly observed) distribution. While the relevance model steers the generation of words in text IR, in our approach it drives the sampling of users, or to be more specific, user profiles (i.e. a history of item ratings). In this perspective, $p(v|R_u)$ can be read as the probability to observe the ratings entered by user v if her underlying tastes are defined by R_u.

2.3 Item Ranking

In document retrieval, once the terms with the highest estimated probability under the relevance model are selected for expansion, they are used to produce a second ranking by means of the negative cross entropy scoring function. In this paper we propose to use this very same method for ranking the item collection with respect to the preferences of the user u, that is, $\hat{r}(u,i) = H(p(\cdot|R_u); p(\cdot|i)) = \sum_{v\in\mathcal{C}_U} p(v|R_u)\log p(v|i)$. With this formulation, we rank the items according to the distance between the item and user probability models, so that the closest – more similar and (hypothetically) more relevant – items are ranked higher. Now, following the kNN CF strategy, we propose to restrict the sum over users v to a subset $N_k(u) \subset \mathcal{C}_U$ of k nearest neighbours with most similar tastes to user u. We propose the use of $p(v|R_u)$ as the similarity function, such that we select the neighbourhood for a given user u as the set of k users in the collection with the highest probability to share the user relevance model R_u. The resulting ranking function is:

$$\begin{aligned} \hat{r}(u,i) &= H(p(\cdot|R_u); p(\cdot|i)|N_k(u)) = \qquad (4) \\ &= \sum_{v\in N_k(u)} p(v|R_u)\log p(v|i) \end{aligned}$$

The result of this restriction is that we avoid the residual effect of a long tail of users with a low $p(v|R_u)$ (Eq. 3), whose contribution to the prediction of user tastes does not pay off the incurred computational cost. This conforms to the principle of neighbour selection in kNN CF, but also of keyword selection (a cutoff of most probable words given the relevance model) in query expansion by RM.

2.4 Parameter Estimation

Finally, some estimation details remain to be defined. We initially consider $p(i)$ and $p(u)$ as uniform priors, i.e. every neighbour $v \in N_k(u)$ has the same probability of being sampled, and same for every item in the collection. The conditional distribution $p(j|i)$ is computed by the maximum likelihood estimate $p_{ml}(j|i) = |U(j) \cap U(i)|/|U(i)|$, where $U(i)$ is the set of users who rated the item i. The probability of a user given an item is computed by Bayesian inversion $p(u|i) = p(i|u)p(u)/p(i)$, and the probability of an item given a user is estimated by maximum likelihood smoothed with the probability in the collection (background collection model), using the Jelinek-Mercer smoothing [10]:

$$p_\lambda(i|u) = (1-\lambda)p_{ml}(i|u) + \lambda p(i|\mathcal{C}) \tag{5}$$

where $p_{ml}(i|u)$ and $p(i|\mathcal{C})$ are estimated as:

$$p_{ml}(i|u) = \frac{r(u,i)}{\sum\limits_{j \in I(u)} r(u,j)}, \;\; p(i|\mathcal{C}) = \frac{\sum\limits_{u \in \mathcal{C}_U} r(u,i)}{\sum\limits_{j \in \mathcal{C}_I, u \in \mathcal{C}_U} r(u,j)} \tag{6}$$

where $\mathcal{C}_U$ ($\mathcal{C}_I$) is the set of users (items) in the collection.

3. EMPIRICAL EVALUATION

3.1 Evaluation Methodology

We test the proposed approach on two publicly available datasets[1]. The first one is *MovieLens 100K*, which contains $100,000$ ratings on $1,682$ items by 943 users. We perform a 5-fold cross validation using the data splits contained in the public distribution, which retain 80% of the data for training, and the rest for testing. Using the terminology in [2], we report the results obtained following the *TestItems* evaluation approach described in [1]. In the TestItems methodology, a ranking is generated for each user by predicting a score for every item that has a rating in the test set. We can then compute any standard IR metric on the ranking, such as precision, nDCG or MRR. We report here the values for precision at 5 and 50, and nDCG at 5 and 10. We also report the *user space coverage* metric (cvg) as defined in [7], that is, the number of users for which the system is able to recommend at least one item.

Furthermore, for assessing the robustness of the methods across collections, we took the optimal parameter values for every method in terms of P@5 in the MovieLens 100K dataset, and tested the methods with those values on a second (and disjoint) public dataset, *MovieLens 1M*, containing $1,000,209$ ratings by $6,040$ users to $3,900$ items. We do 5-fold cross validation again in this larger dataset – not with the aim of training the parameters but to enhance the randomisation of the data split, following standard methodology in the evaluation of recommender systems.

3.2 Baselines

We compare our methods against different well-known state-of-the-art recommenders. We take a user-based CF where ratings are predicted as in Eq. 1, with Pearson correlation as similarity measure (**UB** [6]). We also test our methods against the graph partitioning method [2] based on Normalised Cut (NC) with Pearson similarity (**NC+P**) which has demonstrated important improvements for neighbour selection. This method basically clusters the users in the collection by finding the optimal cut (NC) of the computed graph, where Pearson similarity is used to weight the edges between items. Then, it selects a neighbourhood $NC_k(u)$ that outputs those users who belong to the same cluster as the target user u among the k clusters found by the algorithm; finally, the predicted rating is produced like in Eq. 1 with $N_k(u) = NC_k(u)$.

Furthermore, we compare our approach to related work on recommendation algorithms that adapt IR models. Specifically, we compare our methods against the relevance model for log-based CF (**UIR**) proposed in [8], as formulated in the Eq. 16 of that paper:

$$\hat{r}(u,i) \propto \sum_{\substack{v \in U(i) \\ c(u,v)>0}} \log\left(1 + \frac{(1-\lambda)p_{ml}(v|u,r)}{\lambda p(v|r)}\right) + |U(i)| \log \lambda \tag{7}$$

where $c(u,v)$ is the number of items rated by both users u and v and the rest of probabilities are estimated as follows:

$$p_{ml}(v|u,r) \propto \frac{c(v,u)}{c(u)}, \;\; p(v|r) \propto c(v)$$

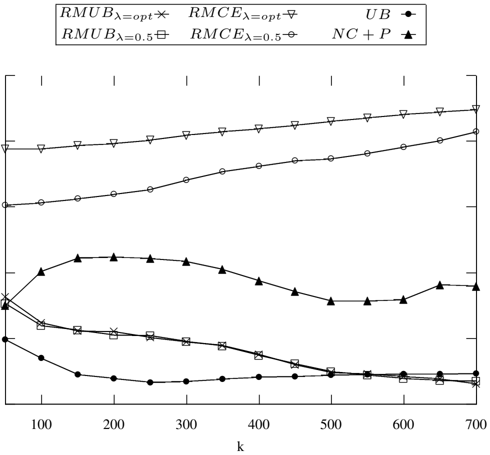

Figure 2: Evolution of the performance of the compared methods in terms of $P@5$ when varying k on the MovieLens 100K collection

We also include as a baseline the unified user-based model (**URM**) presented in [9], which allows for introducing ratings in the probability estimations. More specifically, we use the Eq. 40a from [9] which goes as follows:

$$\hat{r}(u,i) = \frac{\sum_{v \in U(i)} r(v,i) e^{-\frac{1-\cos(u,v)}{h_u^2}}}{\sum_{v \in U(i)} e^{-\frac{1-\cos(u,v)}{h_u^2}}} \tag{8}$$

where $\cos(u,v)$ is a cosine kernel based similarity measure between users u and v represented as vectors in an item space, where the missing ratings can be replaced by a constant value of 0, or by the average rating value. This approach requires a prior learning of the value h_u (the kernel bandwidth window parameter) by expectation-maximisation [9]. In order to provide a fair comparison, we use here the best value $h_u^2 = 0.79$ reported in [9], which was tuned on the very same collection.

3.3 Results and Discussion

We now assess the research questions RQ1 (are RM models useful to identify neighbours in recommendation?) and RQ2 (can we achieve better performance with a complete probabilistic representation of the CF problem?), raised at the beginning of the paper, in light of the results, summarised in Figure 2 and Table 1. Figure 2 shows how sensitive the evaluated methods are to the neighbourhood size (or number of clusters for NC+P). In Table 1 we present the results for different evaluation metrics using the two datasets described previously. For our approaches, we first test a hybrid combination of our neighbour selection approach using Eq. 3 followed by the standard user-based CF formulation with Pearson similarity (Eq. 1); we refer to this method as **RMUB**. Additionally, we denote by RMCE the combination of the relevance model (Eq. 3) followed by the cross entropy ranking function (Eq. 4).

To address **RQ1** we compare the performance of RMUB against that of the other baselines. We observe that the RMUB method clearly outperforms the UB, UIR and URM baselines for P@5, nDCG@5 and nDCG@10, in both MovieLens 100K (see Table 1a) and 1M (Table 1b). Its performance in terms of P@50, however, is similar to some of the baselines, showing that our method is able to rank higher than such baselines interesting items for the user,

[1] Both available at http://www.grouplens.org/node/73

Table 1: Summary of comparative effectiveness. Best values for each collection and metric are in bold. Statistical significant improvements w.r.t. UB, NC+P, UIR, URM, RMUB and RMCE are superscripted with a, b, c, d, e and f respectively (Wilcoxon Test with $p < 0.01$). Trained parameter values are $k = 50$; $k = 200$; $h_u^2 = 0.79$; $\lambda = 0.1$; $k = 50$ and $\lambda = 0.1$; and $k = 700$ and $\lambda = 0.9$ respectively.

(a) MovieLens 100K

Method	P@5	nDCG@5	nDCG@10	P@50	cvg
UB	0.049^{cd}	0.041^{cd}	0.047^{cd}	0.056^{ce}	100%
NC+P	0.111^{acde}	0.097^{acde}	0.095^{acde}	0.058^{ce}	83%
UIR	0.004	0.002	0.002	0.002	100%
URM	0.005	0.003	0.018	0.054^{ce}	100%
RMUB	0.081^{acd}	0.064^{acd}	0.062^{acd}	0.050^{c}	60%
RMCE	$\mathbf{0.224}^{abcde}$	$\mathbf{0.204}^{abcde}$	$\mathbf{0.204}^{abcde}$	$\mathbf{0.138}^{abcde}$	100%

(b) MovieLens 1M

Method	P@5	nDCG@5	nDCG@10	P@50	cvg
UB	0.035^{cd}	0.031^{cd}	0.031^{cd}	0.039^{cde}	100%
NC+P	0.037^{acd}	0.033^{acd}	0.036^{acd}	0.048^{acde}	99%
UIR	0.001	0.001	0.001	0.001	100%
URM	0.001	0.001	0.006	0.034^{c}	100%
RMUB	0.075^{abcd}	0.061^{abcd}	0.057^{abcd}	0.038^{c}	41.4%
RMCE	$\mathbf{0.187}^{abcde}$	$\mathbf{0.176}^{abcde}$	$\mathbf{0.168}^{abcde}$	$\mathbf{0.108}^{abcde}$	100%

at least until some reasonable cut-off, which in this case seems to be 50. Moreover, since this method takes two parameters (k and λ), we analyse now its performance sensitivity. Due to space constraints, we only explore in Figure 2 the neighbourhood size k, but we include the performance for two values of λ – the optimal ($\lambda = 0.1$) and neutral ($\lambda = 0.5$) configurations – where a negligible difference is obtained. We can also notice in the same figure that the baseline NC+P obtains a much better performance than RMUB, consistently with results reported in [2]. Table 1 also shows that the coverage results for the NC+P baseline are better than for RMUB in their optimal settings. We further observed (we omit the detailed results here for the sake of space) that the coverage of NC+P decreases with larger k's (as reported in [2]), whereas the coverage of RMUB increases, but at the expense of losing precision. All in all, our answer to RQ1 is that relevance models as a standalone method for neighbour selection are useful but not optimal.

To address **RQ2** we focus on the RMCE approach. In this case our method consistently achieves statistically significant improvements against all the baselines for every reported metric, achieving a 100% improvement with respect to the best baseline (NC+P, which already demonstrated performance superior to a standard Matrix Factorisation baseline [2]). Furthermore, RMCE does not suffer from the coverage problem, although it is highly sensitive to the smoothing parameter, as Figure 2 shows. For this approach, the optimal parameter in MovieLens 100K is 0.9, that is, a configuration which relies heavily on the background collection model. This makes sense since, in such a setting, RMCE promotes popular recommendations which are known to perform very well in this dataset. Furthermore, as the same figure shows, the method outperforms the baselines even for a neutral setting of the smoothing parameter. Thus, we may conclude with a positive answer for RQ2, since the combination of RM-based neighbours and negative cross entropy as scoring function (RMCE) results in important improvements over the existing state-of-the-art CF methods.

Finally, it is interesting to observe how differently the RMUB and RMCE approaches perform, taking into account that the neighbours used for both methods are the same. Our hypothesis is that the classical user-based CF formulation (Eq. 1) has no formal justification to generate item rankings, mainly because it was proposed to predict ratings, not to rank items according to this predictions, in agreement with [4]. By using negative cross entropy as the retrieval function, the ranking shifts from guessing rating values to assessing relevance distribution distances, which proves to reward relevance over rating value accuracy, as we may notice in terms of relevance-oriented ranking metrics; according to the results, we can conclude that like in IR, negative cross entropy maintains its good properties in recommendation tasks.

4. CONCLUSIONS AND FUTURE WORK

We have presented a new approach to collaborative filtering in Recommender Systems. Our approach adapts the negative cross entropy ranking principle from the Relevance-Based Language Models in document retrieval to the item recommendation problem, combined with a neighbour selection step, drawing from the kNN CF principle. We tested our proposal first only for neighbourhood selection and then also for producing the recommendation, finding that the largest improvements are achieved when using the complete probabilistic model. Comparisons of our approach with other highly performing baselines shows consistent significant improvements for every evaluation metric. As future work, we plan to explore the behaviour of our proposal on larger datasets, and study how further the improvements can go with alternative estimation formulations such as different smoothing methods and RM models. We have also researched further alternatives in how the IR and recommendation variables are mapped [5], although more research is still needed on this point.

5. ACKNOWLEDGMENTS

This work was funded by *Secretaría de Estado de Investigación, Desarrollo e Innovación* from the Spanish Government under projects TIN2012-33867 and TIN2011-28538-C02.

6. REFERENCES

[1] BELLOGÍN, A., CASTELLS, P., AND CANTADOR, I. Precision-oriented evaluation of recommender systems: an algorithmic comparison. In *RecSys* (2011), pp. 333–336.

[2] BELLOGÍN, A., AND PARAPAR, J. Using graph partitioning techniques for neighbour selection in user-based collaborative filtering. In *RecSys* (2012), pp. 213–216.

[3] LAVRENKO, V., AND CROFT, W. B. Relevance based language models. In *SIGIR* (2001), ACM, pp. 120–127.

[4] MCLAUGHLIN, M. R., AND HERLOCKER, J. L. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *SIGIR* (2004), pp. 329–336.

[5] PARAPAR, J., BELLOGÍN, A., CASTELLS, P., AND BARREIRO, A. Relevance-based language modelling for recommender systems. *Inf. Process. Manage. 49*, 4 (2013), 966–980.

[6] RESNICK, P., IACOVOU, N., SUCHAK, M., BERGSTROM, P., AND RIEDL, J. Grouplens: An open architecture for collaborative filtering of netnews. In *CSCW* (1994), pp. 175–186.

[7] SHANI, G., AND GUNAWARDANA, A. Evaluating recommendation systems. In *Recommender Systems Handbook*. 2011, pp. 257–297.

[8] WANG, J., DE VRIES, A., AND REINDERS, M. A user-item relevance model for log-based collaborative filtering. In *ECIR* (2006), Springer-Verlag, pp. 37–48.

[9] WANG, J., DE VRIES, A. P., AND REINDERS, M. J. T. Unified relevance models for rating prediction in collaborative filtering. *ACM Trans. Inf. Syst. 26*, 3 (June 2008), 16:1–16:42.

[10] ZHAI, C., AND LAFFERTY, J. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst. 22*, 2 (2004), 179–214.