Self Recommendation in Peer to Peer Systems

Agostino Forestiero
Institute for High Performance Computing and Networking, ICAR-CNR,
Via Pietro Bucci, 41C, 87036 Rende (CS), Italy

Keywords: Recommendation systems, Peer to Peer.

Abstract: Recommendation system aims to produce a set of significant and useful suggestions that can be meaningful for a particular user. This paper introduces a self-organizing algorithm that by exploiting of a decentralized strategy builds a distributed recommendation system. The available resources are represented by a string of bits namely *describer*. The describers are obtained by exploiting of a locality preserving hash function that maps similar resources into similar strings of bits. Each pear works independently with the aim to locate the similar describer in neighbor peers. The peer decisions are based on the application of ad-hoc probability functions. The outcome will be a fast recommendation service thanks to the emergent sorted overlay-network. Preliminaries experimental results show as the logical reorganization can improve the recommendation operations.

1 INTRODUCTION

The task of recommendation systems is to create a list of interesting items for the users when it has to make a choose in a given contest. In other words, it use the past opinions e/o the behaviors of whole community to help the users of the same community to more efficiently make a new choice. These systems can be built for movies, books, communities, news, articles etc. Recommendation systems have become an important research topic and much work have been proposed both in the industry and academia on developing new approaches in this field. The companies collect a large amount of transactional data that allows a careful analysis of how an user interacts with the set of available choices. This can be a way to automatize the generation of recommendations based on data analysis. The way used to analyze the data and develop the concepts of affinity between users and items distinguishes the recommendation systems.

The usefulness of an item or product is generally represented by a rating, which indicates how a given user liked a particular item. The items or products having an high value of rate are presented as recommendations for the user. The recommendation systems can be categorized as (Balabanović and Shoham, 1997): (i) *Collaborative Filtering* (CF) where an item or product is recommended to the user according to the past ratings of all users, i.e. systems are based on historical interactions; (ii) *Content-based recommending* where the item or product is recommended if it is similar in content to items or products the user has chosen in the past, or matched with given attributes of the user; (iii) *Hybrid approaches* in which collaborative and content-based approaches are combined. In collaborative filtering approach – the term was introduced in first commercial recommendation system known as Tapestry (Goldberg et al., 1992) – the utility of the item i for the user u is estimated based on the utilities assigned to item i by those users v who are "similar" to user u. Practically, this kind of system try to predict the utility of items for a given user based on the items previously rated by other similar users. For example, in a music recommendation system, to recommend music to user u, the collaborative system finds the "similar" of user u, i.e., other users that have similar tastes in music. Then, the music that are liked by the similar of user u would be recommended. To compute the similarity between two users have been used various approaches, where, often, the similarity is based on their ratings of items that both users have rated. The most popular are correlation and cosine similarity. In the correlation-based approach, the Pearson correlation coefficient is used to compute the similarity (Resnick et al., 1994), (Shardanand and Maes, 1995):

$$simlarity(u,v) = \frac{\sum_{i\varepsilon I}(r_{u,i} - \bar{r_u})(r_{v,i} - \bar{r_v})}{\sqrt{\sum_{i\varepsilon I}(r_{u,i} - \bar{r_u})^2 \sum_{i\varepsilon I}(r_{v,i} - \bar{r_v})^2}}$$

where **I** is the set of all items rated by both users u

and *v*. The value of the rating *r* for user *u* and item *i* is computed as an aggregate of the ratings of some other users for the same item *i*. The cosine-based approach (Breese et al., 1998), (Sarwar et al., 2001), uses two vectors in *n-dimensional* space to represent the users *u* and *v*, and *n* will be $|I|$. The cosine of the angle between two vectors can be computing to measure the similarity between them:

$$similarity(u,v) = cos(\overrightarrow{u}, \overrightarrow{v}) = \frac{\overrightarrow{u} \cdot \overrightarrow{v}}{|\overrightarrow{u}|_2 \times |\overrightarrow{v}|_2} = \frac{\sum_{i\varepsilon I} r_{u,i} r_{v,i}}{\sqrt{\sum_{i\varepsilon I} r_{u,i}^2}\sqrt{\sum_{i\varepsilon I} r_{u,i}^2}}$$

where $\overrightarrow{u} \cdot \overrightarrow{v}$ indicates the *dot-product* between the vectors $\overrightarrow{u}$ and $\overrightarrow{v}$.

Content-based recommenders provide recommendations by comparing representations of content that interests the user to representations of content describing an item. Recommended items have associated textual content, such as books, web pages, and movies. The web pages should be associated to contents like descriptions and user reviews. Information retrieval (IR) technics address this problem, where the content associated can be handled as a query, and the unrated documents marked with a similarity value to this query (Balabanović and Shoham, 1997). Otherwise, the documents can be converted into word vectors, and then averaged to obtain a prototype vector of each category for a user, as showed in (Lang, 1995).

Collaborative and content-based approaches use the same cosine measure from information retrieval. But, in content-based recommendation systems measures the similarity between vectors of weights, whereas, in collaborative systems measures the similarity between vectors of the actual ratings specified of the users. Other approaches to the recommendation consist in handling of the problem as a classification task. Each pattern represents the content of an item, and a user's past ratings are used as labels for these patterns. For example, text from fields such as title, author, synopses, reviews, and subject terms are used by (Mooney and Roy, 2000) to recommend books. Several classification algorithms have been used to content-based recommend: decision trees, k-nearest neighbor, and neural networks (Pazzani and Billsus, 1997).

In this paper, a self organizing algorithm for building a peer to peer recommendation system, is proposed. The algorithm is able to distribute and cleverly organize the "descriptor of resources" in order to improve discovery operations. In peer to peer systems, bit vectors or keys, are often exploited to describe the resources and with different meanings. The presence or absence of a given topic can be represented thorough a bit (Crespo and Garcia-Molina, 2002)(Platzer and Dustdar, 2005). With resource like documents, it is particularly appropriate, because it is possible to recognize the topics. An hash function was employed to map the resources with strings of bits in (Cai et al., 2004) (Oppenheimer et al., 2005). With an hash function locality preserving, similar describers are assigned to similar resources and then placed in the same region of the network. Thus, it is very probable find similar and appreciated describers close to the target describer (recommendations). In the rest of the paper a preliminary version of the algorithm is introduced in section 2 and an initial experimental analysis is showed in section 3.

2 SELF ORGANIZING RECOMMENDATION SYSTEM

In this section a distributed self-organizing algorithm for building a recommendation system, is introduced. The aim of the algorithm is to logically reorganize the describers that describe the available and recommendable resources to allow recommendation operations faster. The algorithm cleverly disseminates the describers on the network with the aim of spatially sort them. Thanks to this spatial reorganization, similar describers, representing similar resources, will be placed in neighbor hosts. A set of similar describers representing similar resources can be detected in the neighborhood of the target resource and suggested to the user. The sorting process is progressively and continuously realized by each peer achieving simple and local operations. Probability functions steer the operations of sending and depositing of the describers. These simple operations are performed in local and a sort of global intelligence emerges from work of unaware peers. Two probability functions, P_{send} to evaluate the probability to send the describers, and $P_{deposit}$ to evaluate the probability to deposit the describers, are employed. The probability functions derive from the formulas exploited by biological systems for self-organizing their behavior (Lumer and Faieta, 1994). In this systems, unaware entities independently and locally work in order to produce a global intelligent behavior.

Each peer evaluates the probability function P_{send} for each stored describer, so as to decide whether or not to gather describers and send them to a neighbor peer; the probability function $P_{deposit}$ is evaluated by a peer when a set of describers arrives from a neighbor peer. Peer' decisions, i.e. probability functions, are based on a similarity function that measures the average similarity of a describer *des* with all the describers located in the local region. All the hosts reachable from the current host with a given number of hops represent the local region. Here, the similarity function S for the describer $\underline{des}$ in local *Region* is reported in formula(1):

$$S = \frac{1}{N} \sum_{des \varepsilon Region} N_h \cdot \left(1 - \frac{1 - cos(des, \bar{des})}{\alpha}\right) \quad (1)$$

where, N is the overall number of describers in the *Region*, N_h is the number of describers maintained in each host, while $cos(des, \bar{des})$ is the cosine distance between des and $\bar{des}$. The parameter α is the similarity scale and here it is set to 2. The value of S assumes values ranging between -1 and 1, but negative values are fixed to 0. The bulk of describers is propagated across the network until all describers are dropped. Each peer that receives the set of describers from a neighbour evaluates the $P_{deposit}$ function for each describer and in case take it. The probability to send a describer must be inversely proportional to the similarity of this describer with those located in the visibility region, so that dissimilar describer are sent away of the region. When similar describer being to be accumulated the initial equilibrium is broken and a reorganization of describers is increasingly driven. The probability function to gather for sending a describer is defined in formula(2):

$$P_{send} = \left(\frac{k1}{k1 + S}\right)^2 \quad (2)$$

the parameter k1, whose value is comprised between 0 and 1, can be tuned to modulate the degree of similarity. Here k1 is set to 0.1 (Bonabeau et al., 1999). The local region accumulates similar describers because the dissimilar describers will be sent away.

Whenever a bulk of describers gets to a new host, the probability function $P_{deposit}$,is evaluated. It is directly proportional to the similarity function S , i.e., to the average similarity of this describer with the describers maintained in the current visibility region.

$$P_{deposit} = \left(\frac{S}{k2 + S}\right)^2 \quad (3)$$

the parameter k2 is set to 0.5 (Bonabeau et al., 1999).

An algorithm for exploiting the logical reorganization and then obtain a set of describers representing resource that can be suggested, is very simple and immediate. A query will be issued by an host (user) to search a *target describer* representing the wished resource and it will be forwarded through the peer to peer network to collect as many target describers as possible. Thanks to the logical reorganization a, the queries can be forwarded towards the host with the maximum value of similarity between *representative describer* and the target describer. The representative describer is a virtual describer, for each host, built by averaging of the values of all describers located in a current host. When a query is issued by an host for a target resource, a virtual target describer is created. The query will be forwarded towards the neighbor peer with the maximum value of similarity, based on formula 1, between the virtual target describer of the query and the representative describer of the host. The same operation will be done by each host that received a query and has to forward it to one of its neighbors. When the query gets to an host with a representative describer equal to the virtual target describer, or the maximum number of query hops admissible is finished, the query will be directly forwarded to the host that has issued the request. The query going across the network collects a set of describers similar to the virtual target describer, which can be exploited to produce a list of suggestion/recommendation to the user. The discovery algorithm is very simple and needs very little computing and memory resources, it is very efficient as it exploits the continuous work of the algorithm that organize the describers.

3 EXPERIMENTAL RESULTS

An event-based simulator was implemented to evaluate the performance of the algorithm. A P2P network with number of hosts equal to 2,500 was considered where each peer is linked to 4 hosts on average. The number of resources published by each host is equal to 15 on average and indexed with a prefixed string of bits obtained using a locality preserving hash function to guarantee that describer give similar keys. Exploiting the algorithm of Albert and Barabasi (Barabási and Albert, 1999)a scale free topology network was built. In this way, the characteristics of real networks are careful considered. A graphical description of the logical reorganization is reported in Figure 1. Here, each describer is associated to a color. A part of the network is photographed: (a) at *Time* = 0 sec, when the process is starting and the describers are randomly distributed and (b) at *Time* = 50,000 time units when the process is in a steady situation. Notice that similar describers are located in the same region and between near region the color change gradually, which proves the spatial sorting on the network.

The traffic generated by the process of reorganization, that is the average number of bulks per second that are processed by an host, does not depend neither on the network size nor on the churn rate. It

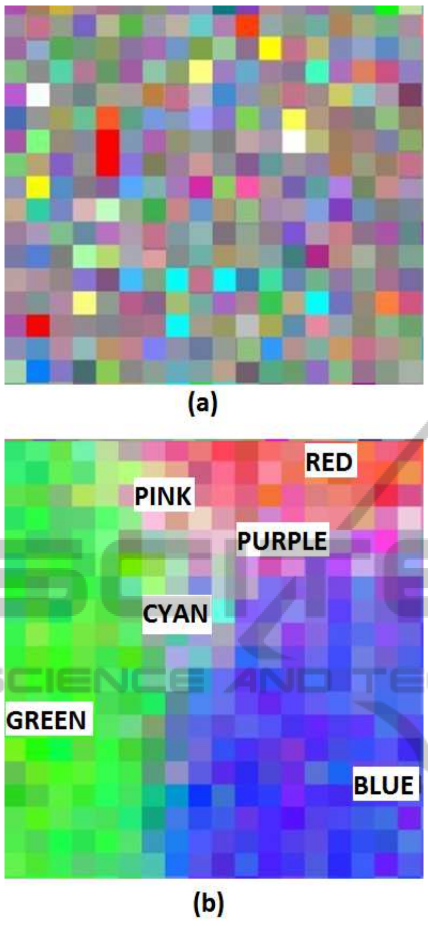

Figure 1: Snapshots of a part of the network when the process is starting (a), and when the process is in a steady situation (b).

only depends on the number of forwarding and their frequency across the network. In simulated scenario, each server processes about one send/deposit operation every 20 time units, which can be considered an acceptable load for the host. The traffic, that is the number of operations that a host elaborates per time units, was calculated and shown in Figure 2. We can see as the value of the traffic changes according to the value of maximum number of hops done within a single sending. In this figure the distance of the neighbor target peer, i.e. the number of hops achieved by bulks before the host evaluates the probability function, was varied. It was noted that the reorganization process is accelerated if distance of jump are longer, because they can scan the network more quickly. The maximum number of hops is a compromise between the traffic load tolerable and the rapidity and efficiency of the reorganization. It was possible to note during the simulations that the processing load does not depend on system parameters such as the average number of

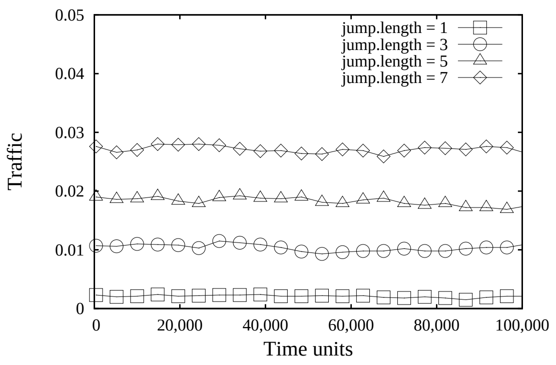

Figure 2: The traffic generated by the algorithm when the number of hops of each sending ranges from 1 to 7.

resources handled by a host or the number of hosts, which is a confirmation of the scalability properties of the algorithm.

A spatial index of *Similarity rate* was defined to evaluate the goodness of the algorithm. For each peer, the similarity among all the local describer within the local region, by averaging the cosine of the angle between every couple of describers, was calculated. The values of the Similarity rate has been averaged for all the hosts of the network. Our aim is to increase the Similarity rate value as more as possible. It would mean that similar describers are located into neighbor hosts and an effective sorting of describer is becoming. In Figure 3 the Similarity rate of the whole

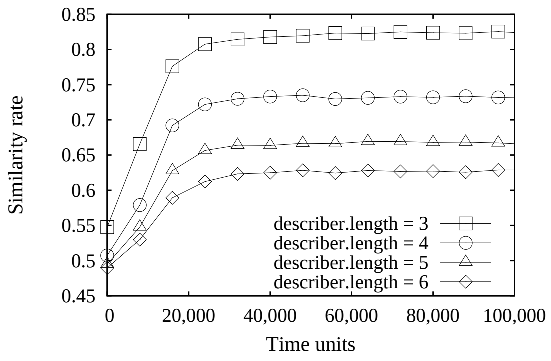

Figure 3: Similarity rate of the whole network when the length of bit strings ranges from 3 to 6.

network when the length of describer bit strings that represent the resources, is varied. It is possible to note how the logical reorganization is achieved independently of the length of bit strings. The scalability of the algorithm is confirmed analyzing its behavior when the network size is varied. Figure 4 reports the values of Similarity rate when the network size ranges from 1,000 to 16,000 hosts. Notice that the number of the involved hosts in the logical reorganization, has no

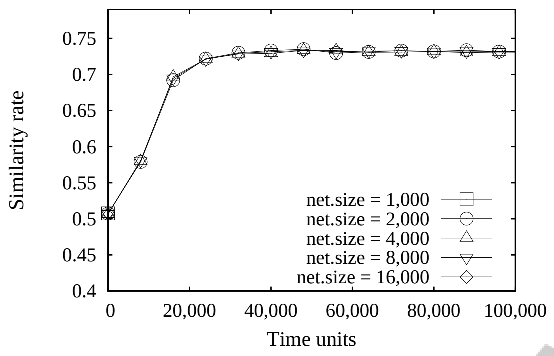

Figure 4: Similarity rate of the whole network when the network size ranges from 1,000 to 16,000 hosts.

detectable effect on the Similarity rate value.

4 CONCLUSION

A self-organizing algorithm to build a distributed recommendation system, was introduced. The available and recommendable resources in a network are described by means *describer* and logically reorganized. The describer are arranged as bit strings obtained by the application of a locality preserving hash function that allows to map similar resources into similar strings. Hosts autonomously send/deposit describer exploiting probability functions. Preliminary experimental results showed as the algorithm achieves an effective reorganization of descriptors. The emerging logical overlay allows to improve discovery and recommendation operations.

REFERENCES

Balabanović, M. and Shoham, Y. (1997). Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *science*, 286(5439):509–512.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*, volume 4. Oxford university press New York.

Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc.

Cai, M., Frank, M., Chen, J., and Szekely, P. (2004). Maan: A multi-attribute addressable network for grid information services. *Journal of Grid Computing*, 2(1):3–14.

Crespo, A. and Garcia-Molina, H. (2002). Routing indices for peer-to-peer systems. In *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, pages 23–32. IEEE.

Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70.

Lang, K. (1995). Newsweeder: Learning to filter netnews. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339. Citeseer.

Lumer, E. D. and Faieta, B. (1994). Diversity and adaptation in populations of clustering ants. In *Proceedings of the Third International Conference on Simulation of Adaptive Behavior : From Animals to Animats 3: From Animals to Animats 3*, SAB94, pages 501–508. MIT Press.

Mooney, R. J. and Roy, L. (2000). Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204. ACM.

Oppenheimer, D., Albrecht, J., Patterson, D., and Vahdat, A. (2005). Design and implementation tradeoffs for wide-area resource discovery. In *High Performance Distributed Computing, 2005. HPDC-14. Proceedings. 14th IEEE International Symposium on*, pages 113–124. IEEE.

Pazzani, M. and Billsus, D. (1997). Learning and revising user profiles: The identification of interesting web sites. *Machine learning*, 27(3):313–331.

Platzer, C. and Dustdar, S. (2005). A vector space search engine for web services. In *Web Services, 2005. ECOWS 2005. Third IEEE European Conference on*, pages 9–pp. IEEE.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM.

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM.

Shardanand, U. and Maes, P. (1995). Social information filtering: algorithms for automating word of mouth. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217. ACM Press/Addison-Wesley Publishing Co.