Querying Similarity in Metric Social Networks

Jan Sedmidubský, Stanislav Bartoň, Vlastislav Dohnal, and Pavel Zezula

Masaryk University
Brno, Czech Republic
[xsedmid, xbarton, dohnal, zezula]@fi.muni.cz

Abstract. In this paper we tackle the issues of exploiting the concepts of social networking in processing similarity queries in the environment of a P2P network. The processed similarity queries are laying the base on which the relationships among peers are created. Consequently, the communities encompassing similar data emerge in the network. The architecture of the presented metric social network is formally defined using the acquaintance and friendship relations. Two version of the navigation algorithm are presented and thoroughly experimentally evaluated. Finally, learning ability of the metric social network is presented and discussed.

1 Introduction

The area of similarity searching is a very hot topic for both research and commercial applications. Current data processing applications use data with considerably less structure and much less precise queries than traditional database systems. Examples are multimedia data like images or videos that offer query-by-example search, product catalogs that provide users with preference-based search, scientific data records from observations or experimental analyses such as biochemical and medical data, or XML documents that come from heterogeneous data sources on the Web or in intranets and thus does not exhibit a global schema. Such data can neither be ordered in a canonical manner nor meaningfully searched by precise database queries that would return exact matches.
This novel situation is what has given rise to similarity searching, also referred to as content-based or similarity retrieval. The most general approach to similarity search, still allowing construction of index structures, is modeled in metric space. Many index structures were developed and surveyed recently [13]. However, the current experience with centralized methods [6] reveals a strong correlation between the dataset size and search costs. Thus, the ability of centralized indexes to maintain a reasonable query response time when the dataset multiplies in size, its *scalability*, is limited. The latest efforts in the area of similarity searching focus on the design of distributed access structures which exploit more computational and storage resources [2, 7, 10, 4, 3]. Current trends are optimizing and tuning the well known distributed structures towards better utilization of the available resources.
Another approach to design the access structure suitable for large scale similarity query processing that is introduced in this paper emerges from the notion of

social network. A social network is a term that is used in sociology since the 1950s and refers to a social structure of people, related either directly or indirectly to each other through a common relation or interest [11]. Using this notion, our approach places the peers of the distributed access structure in the role of people in the social network and creates relationships among them according to the similarity of the particular peer's data. The query processing then represents the search for the community of people – peers related by common interest – similar data.

Using this data point of view our designed metric social network is a cognitive knowledge network according to the terminology stated in [9] that is described as *who thinks who knows what* where it is not who you know but it is what who you know knows. This means that the network links are created on the basis of the particular peers' knowledge – stored data – rather than on being acquainted with other peers. As for the navigation, social networks exhibit the *small world network topology* [12] where most pairs of nodes are reachable by a short chain of intermediates – usually the average pairwise path length is bound by a polynomial in $\log n$. Therefore it is anticipated that a small amount – around six – of transitions will be needed to find the community of peers holding the answer to a query posed at any of the participating peers in the network.

Unlike the usual access structures that retrieve a total answer to each query, the presented approach focuses on retrieving the *substantial part* of the answer yet with *partial costs* compared to the usual query processing. The concepts of social networking towards the approximative query processing in large scale data have already been introduced in related works [1, 8].

2 Architecture

Our social network comprises of entities as usual – vertices and edges representing relationships among them. The relationships between two entities are of two types: the friendship and a relation of acquaintance. The relationships identified among the entities in the graph always relate to a particular query processed by the network and its retrieved answer. In our social network the vertices are the leaf nodes of the M-tree [5] created on the given dataset. This means that firstly the dataset is indexed using the M-tree indexing structure and the contents of leaf nodes of the M-tree are used as the peers in the network. This represents the initial state of the social network.

2.1 Vertices of the Social Network

As we mentioned the leafs of the M-tree created on the dataset represent the vertices in our social network. The vertex itself, besides the assigned piece of data, remembers also the history of the queries that it has been asked. To each query the recognized set of friends and acquaintances is also remembered for future optimization of a similar query processing.

Then a network peer P is $P = (D, H)$ where $D = \{o_1, \dots o_l\}$ represents the assigned piece of data and $H = \{h_1, \dots h_m\}, h_i = (Q, L_P^{Acq}(Q), L_P^{Fri}(Q))$ represents the history of queries with the pair of ordered lists of retrieved acquaintances and friends regarding the particular query Q processed. For example as a query a usual range query can be considered: $R(q, r)$ where q is the query object and r the predefined range.
The peer which is asked to answer the query in the social network is denoted as P_{start}. Consequently, the answer $A(Q)$ is passed from the network to P_{start}. This answer comprises of partial answers of the peers of the network $A_{P_i}(Q)$. So the answer $A(Q) = \bigcup_{i=1}^{n} A_{P_i}(Q)$ where n denotes the number of peers that participate on the answering.

2.2 Measuring Quality of the Query Answer

As we have seen, the total answer can be divided into pieces regarding the peers that participated on the total answer to a query Q. To distinguish which peer answered better, the quality is measured. The quality is defined as a function $Qual(A_{P_i}(Q))$ which returns a quality object q_i as a result. This function returns metadata that represents the quality of the peer's answer. Since this object is not necessarily a number, we also define a function to compare quality objects:

$$compare_{qual}(q_1, q_2) = \begin{cases} -1 & q_1 \text{ is less than } q_2 \\ 0 & q_1 \text{ is same as } q_2 \\ 1 & q_1 \text{ is greater than } q_2 \end{cases}$$

The quality of the total answer is determined by applying the quality measuring function on $A(Q)$. An ordering $\preceq$ which we call the q-ordering is defined on the peers' answers $A_{P_1}(Q), \dots, A_{P_n}(Q)$ in $A(Q)$ according to their qualities. The quality objects are then $q_i = Qual(A_{P_i}(Q))$. The sequence of peers' answers is ordered according to the q-ordering when the following holds:

$$i_1 \dots i_n : q_{i_a} \preceq q_{i_b} \Leftrightarrow a < b \Leftrightarrow compare_{qual}(q_{i_a}, q_{i_b}) \neq 1$$

Intuitively, when the peers are ordered with respect to the position of their answers in the q-ordered set of partial answers they are ordered by their ability to answer the particular query Q.

2.3 Acquaintance and Friendship Relations

As we mentioned earlier, we distinguish two relationships in our social network. Firstly, the relationship of friendship represents the similarity of nodes – two nodes give a similar answer to same query. Secondly, the relationship of acquaintance denotes that the target of the relationship took part in the answer passed to the recipient. Formally, the friendship is defined.
A set of acquaintances for a given query Q is defined as a set of participating peers in the total answer $Acq(Q) = \{P | A_P(Q) \neq \emptyset\}$. Friends are identified in

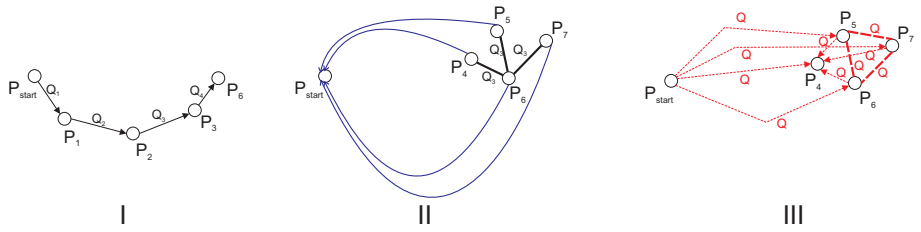

Fig. 1. Query answering process visualization.

the set of acquaintances as peers that at which the similarity of the data kept could be anticipated: $Fri(Q) = \{P||A_P(Q)| > c \cdot |A(Q)|\}$. The friendship relationship according to the particular query Q is assigned only to those peers that contributed to the answer with a significant partial answer, which is expressed by the positive-value constant c.
Then, according to a query Q, a peer P is an acquaintance $P^{Acq}(Q) \Leftrightarrow P \in Acq(Q)$. Similarly, with respect to a query Q, a peer P is a friend $P^{Fri}(Q) \Leftrightarrow P \in Fri(Q)$. Intuitively, the set of acquaintances and friends of each peer that took part in the query processing can be determined using following functions:

$$Acq_P(Q) = \begin{cases} Acq(Q) & P \in Fri(Q) \vee P = P_{start}(Q) \\ \emptyset & \text{otherwise} \end{cases}$$

$$Fri_P(Q) = \begin{cases} Fri(Q) & P \in Fri(Q) \\ \emptyset & \text{otherwise} \end{cases}$$

The edges of the social network are then defined by the two relations defined among the peers – vertices of the network for a given query Q:

$$P_1 \sim_Q^{Acq} P_2 \Leftrightarrow P_2 \in Acq_{P_1}(Q)$$

$$P_1 \sim_Q^{Fri} P_2 \Leftrightarrow P_1 \in Fri_{P_2}(Q), \quad P_1 \in Fri_{P_2}(Q) \Leftrightarrow P_2 \in Fri_{P_1}(Q)$$

Notice that the acquaintance relationship is not symmetric. Therefore this type of edges is directed in the network. The friendship relationship forms an equivalence relation and for such the direction of edges is unnecessary. The equivalence of friends implies the complete graph of relationship among friends regarding one query Q in the network. The nodes of this graph are then connected to the P_{start} through the acquaintance type of edge.
An example of the query processing is demonstrated in Fig. 1. Firstly, when the query Q is posed to the peer P_{start}, through the acquaintance type of edges is navigated to the best supposed acquaintance P_6. Secondly, the peer P_6 contacts through the friendship relationship according to the most similar query from its history to query Q its friends P_4, P_5, P_7 to send their part of the answer to P_{start}. Lastly, upon the $A(Q)$, the new sets $Acq(Q)$ and $Fri(Q)$ are identified and the new edges are stored in the history of the involved peers.

Algorithm 1 Query forwarding algorithm `forwardQuerySimple`

Input: P_{start}, contacted peer P, query Q, last peer's quality $lastQi$

```
1: get entry E = (Q', L_P^Acq(Q'), L_P^Fri(Q')) from history H with Q' most similar to Q
2: P' = best acquaintance from L_P^Acq(Q')
3: if compare_qual(Qual(A_P'(Q')), lastQi) > 0 then
4:    forwardQuery(P_start, P', Q, Qual(A_P'(Q')))
5: else
6:    for all F ∈ L_P^Fri(Q') do
7:       answerQuery(F, P_start, Q)
8:    end for
9:    answerQuery(P, P_start, Q)
10: end if
```

The edges in the history are stored as two lists of peers ordered according to the q-ordering of the partial answers in $A(Q)$ with respect to their qualities. The ordering respects the position of the peer's answer in the q-ordered $A(Q)$. Each of the lists comprises of pairs of a peer P_i and a quality object $q_i = Qual(A_{P_i}(Q))$:

- $L_P^{Fri}(Q)$ ... a list of friends of the peer P for a query Q sorted by *q–ordering*.
- $L_P^{Acq}(Q)$ ... a list of acquaintances of P for Q sorted by *q–ordering* .

2.4 Navigation

The query processing using the social network follows the common world concepts for searching. Basically, the best acquaintance regarding the particular subject is located and then his friends are contacted to return their part of the answer to the querist.

The acquaintances are contacted firstly because they represent the entities that have answered before. Initially, the P_{start} goes through its history of processed queries and finds the most similar query Q' to the query Q that is processing now. For query Q', also the lists of acquaintances and friends are retrieved from the history. The query Q is then forwarded to the best acquaintance regarding the query retrieved from the history. This concept is formalized in Algorithm 2.4. In general, the query can be passed to more then one acquaintance, it depends on the particular navigation algorithm implementation.

The process of the query forwarding can be repeated more times to find the peer that is most promising to hold the searched data. At each peer a different query can be retrieved from the history as the most similar to Q. The stop condition of the query forwarding is when the contacted peer's quality is better than any of his acquaintances to which it could pass the query.

Algorithm 2 Simple query answering procedure `answerQuery`

Input: current peer P, P_{start}, query Q

```
1: get all objects that satisfy Q
2: send retrieved objects to P_start
```

When the the best acquaintance regarding the particular query is found, it returns its part of the query answer to the querist. Then it looks up in the history for the most similar query and retrieves the set of friends associated with that query and forwards the query Q to them as described in Algorithm 2.4. The query is passed also to friends because it is supposed that they hold similar data. Therefore, it is anticipated that also the friends' partial answers will form substantial parts of the query answer $A(Q)$. After contacting, the peers pass their partial answers $A_{P_i}(Q)$ to P_{start}.

3 Experimental Results

In this section we present an experimental evaluation of the proposed distributed access structure for searching in metric data. The experiments have been conducted on two datasets represented by vectors having three and fortyfive dimensions respectively. The 45-dimensional vectors represent extracted color image features. The similarity function for comparing the vectors is a quadratic-form distance. The distribution of the dataset is quite uniform and such a high-dimensional data space is extremely sparse. The three-dimensional vectors were extracted from the high dimensional data using the first three dimensions. The number of vectors in each dataset was 100,000.
The peers in the network have been created using the distributed version of the M-tree indexing structure. The peers then correspond to the leaf nodes of the M-tree where data is present. Besides the data distribution among peers, the M-tree was also used to evaluate the precise total answer to queries used to measure various statistics of query processing by the social network.

3.1 Network Initialization

Firstly, the distributed M-tree is created on the provided dataset. Next, the peers are assigned their pieces of data. In both cases the data have been distributed among 47 peers. In case of the three-dimensional data the occupation ranged from 94 to 3,544 objects and the average occupation per peer was 2,127. The peer occupation for the other dataset ranged from 285 to 3,857 objects.
The edges in the network are then acquired by posing the learning queries to the created M-tree and processing its answer retrieved from the peers in the network. This process is called *learning*.
Intuitively, the longer the learning process is the better the social network query processing is afterwards. In the set of experiments regarding the recall and costs of the query processing using the social network the learning process comprised of 500 queries processed using the M-tree.

3.2 Recall and Costs

The algorithm used to demonstrate the properties of the social network query processing method is the basic variant described in Section 2.4. In this variant,

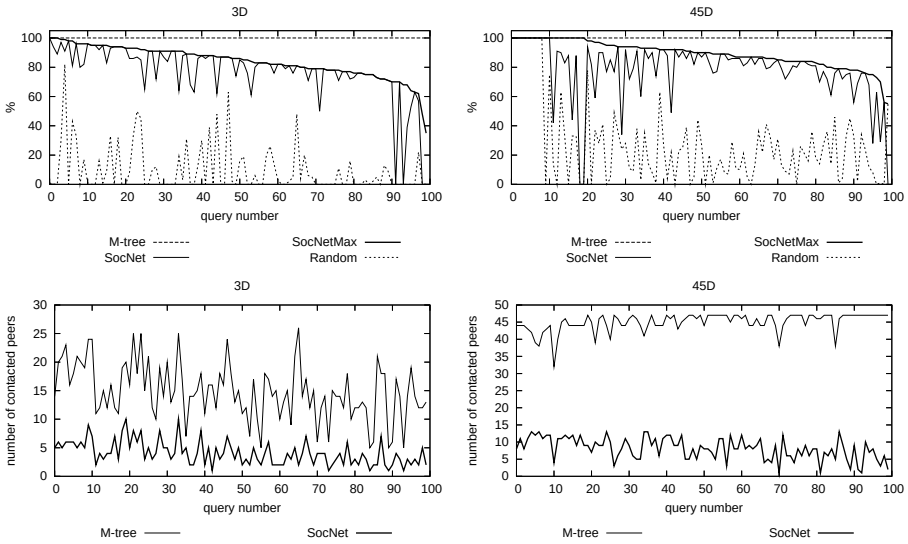

Fig. 2. Results of one hundred queries processed using the social network compared to the total answer of the M-tree: (top row) Recall and (bottom row) Costs.

the query is always forwarded to the best acquaintance and when the stop condition is met, i.e. no better acquaintance can be found, the query is forwarded to all friends of the best acquaintance regarding the closest query in the peer's history. A recall of one hundred queries processed using the built social network is presented in Fig. 2. The queries represent range queries with randomly picked objects with a fixed radius of 200 for the 3D data and 2,000 for the 45D data. These radii have been chosen because they returned on average 3-5% of data.

The queries in Fig. 2 are ordered in a descending manner with respect to the SocNetMax recall values. Both figures demonstrate the percentage of the total answer retrieved using the social network. The recall of SocNetMax value denotes a maximal part of the total answer that could have been retrieved contacting the same amount of peers that the social network did, i.e. if the social network has contacted eight peers, the SocNetMax represents the percentage of objects from the total answer represented by best eight peers. Finally, the random line represents the percentage of the total answer retrieved from the randomly picked peers which amount is the same as the social network has contacted.

The costs of a query processed presented in Fig. 2 are defined as the amount of peers that are contacted in order to answer the query – contacted by the *answerQuery* procedure. We can see that the amount of the peers contacted per query by the social network (SocNet) access structure is substantially smaller than the numbers of the M-tree in the case of the three dimensional data. The gap between those two curves is even greater in the case of the 45D data. This is

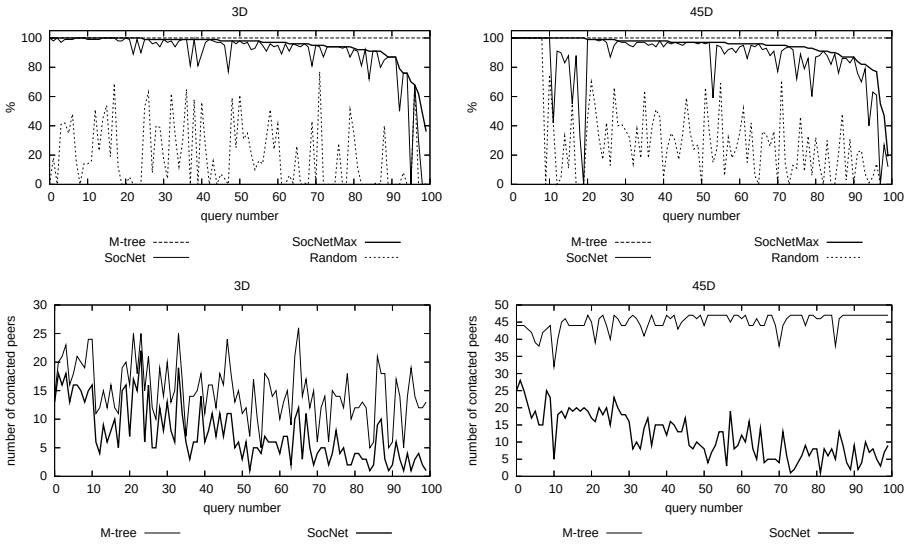

Fig. 3. Results of queries processed by the social network using the *forwardQueryFOAF* navigation algorithm: (top row) Recall and (bottom row) Costs.

caused by the worse clusterability of the latter dataset, since distance between any pair of the objects is more or less the same in this high dimensional data. The fluctuations in both figures are strictly related, i.e. recall is decreasing with the decreasing number of peers used to answer queries. Such a behavior of SocNet is caused by insufficient social information available to some peers. For the 45-dimensional data the recall exhibits higher fluctuations than for the three-dimensional data, which can be attributed to the fact that the complete result is spread over more nodes in the 45-dimensional data. This is also proportional to the ratio between the number of contacted peers by the M-tree and by the SocNet. An improved strategy for routing the query processing tries to eliminate such instability and is presented in the following section.

3.3 Friend of a Friend Query Forwarding

The great differences between the SocNet and SocNetMax in Fig. 2 present a fair amount of instability of the SocNet access structure. We have enhanced the navigation algorithm towards the greater stability of the gained results yet with the emphasis on sustaining the low query processing costs.The enhanced query answering procedure is described in Algorithm 3.3. Besides calling the enhanced answering routine, the query forwarding procedure remains the same. We will refer to this enhanced navigation algorithm as the *forwardQueryFOAF*.
Using the *forwardQueryFOAF* navigation algorithm, the set of peers that participate in the final answer grows because the contacted community of the similar

Algorithm 3 Enhanced query answering procedure `answerQueryFOAF`

Input: contacted peer P, P_{start}, query Q

1: S = $\{E = (Q', L_P^{Acq}(Q'), L_P^{Fri}(Q')) \mid$ query object q' satisfies $Q\}$
2: **for all** $E \in$ S **do**
3: **for all** $F \in L_P^{Fri}(Q')$ **do**
4: answerQueryFOAF(F, P_{start}, Q)
5: **end for**
6: **end for**
7: get all objects that satisfy Q
8: send retrieved objects to P_{start}

peers grows larger. The trends of the results gained can be read in Fig. 3 for recall and Fig. 3 for costs and The queries on which we have measured the properties of the enhanced navigation algorithm are the same as those on which we measured the simple algorithm in the previous subsection.

3.4 Metric Social Network Learning Abilities

In the introduction to the architecture of the designed social network we have mentioned that the result of each query is stored in peers that took place in the query processing. For the structure's evaluation purposes this feature has been disabled during the experiments conducted in the previous section.

In this section we present results gained to demonstrate the ability of the designed social network to learn itself towards better query processing. The experimental results are presented using only the three dimensional dataset because the results gained on the other dataset were very similar to these.

To measure the ability of the social network of learning to process queries a fixed testing set of 20 randomly picked queries was used. The method is that after processing each 50^{th} query the ability to learn was turned off and the recall and costs of the social network on the testing set was measured and stored. An average recall and cost of the testing set queries at each point then represents the ability of the social network to learn. Two initial states of the network were used. Firstly, the state when the social network is well trained using the 250 queries processed by the M-tree. The second initial state represented a state of the network that is not trained yet. The edges in the network were created randomly and assigned to processed queries by the M-tree.

The nature of the social network using the *forwardQuerySimple* navigation algorithm is depicted in Fig. 4. This figure demonstrates the evaluation of the training abilities of the social network using the *forwardQuerySimple* algorithm and *forwardQueryFOAF* algorithm for navigation. In this figure, the curves *recall 1* and *costs 1* represent results initiated in the well trained social network and the *recall 2* and *costs 2* demonstrate the same abilities on the randomly initiated network. Also the costs of the M-tree are presented. The cost values are presented as a proportion of connected peers to the total amount of peers in the network.

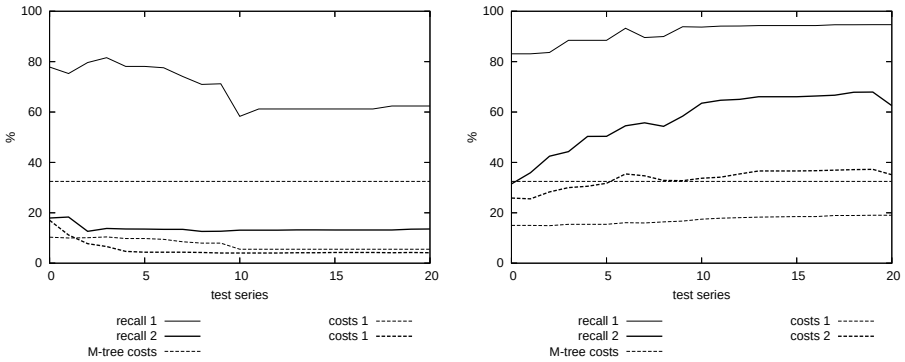

Fig. 4. Learning abilities of the social network access structure using the *forwardQuerySimple* (left) and *forwardQueryFOAF* (right).

The presented results demonstrate the inability of the social network equipped with the *forwardQuerySimple* to refine its social information towards more precise query answering. This is caused by the insufficient overall quality of the answer retrieved from the network. This inability is underscored by the poor results when deployed on the random initial state social network resulting.
On the other hand, using the *forwardQueryFOAF* algorithm to process the queries in the social network yield better results proved by the rising curve represented by the better recall values starting at both well trained and random initial states of the social network. This fact proves that the algorithm and the training abilities are communicating vessels since the more precise answers the better the learning ability is.

4 Concluding Remarks and Future Work

Distributed processing of similarity queries currently attracts a lot of attention because of its inherent capability of solving the issue of data scalability. We have proposed an approach based on social networking which is able to answer any similarity query modelled using metric space paradigm. The principle exploited in this proposal models social relationships with regards to specific queries. As a result, a multigraph is created in which individual communities sharing similar data can be identified. The presented experiment trails confirm suitability and auspiciousness of such approach. Moreover, the network with enhanced navigation is able to evolve autonomously while improving quality of query results.
For future work, various aspects of navigation strategies will be deeply studied in order to design more sophisticated and possibly self-adapting policies. The peers in our social networks were assigned with data objects based on the M-tree clustering principle. Influence of such data partitioning will be verified. Also dynamicity in the sense of peers' joining and leaving the network will be investigated. This is also related to the dynamicity from the data point of view where

the data content of individual peer can change, which invalidates relationships established in the network so far.

References

1. R. Akavipat, L.-S. Wu, F. Menczer, and A.G. Maguitman. Emerging semantic communities in peer web search. In *P2PIR '06: Proceedings of the international workshop on Information retrieval in peer-to-peer networks*, pages 1–8, New York, NY, USA, 2006. ACM Press.
2. Farnoush Banaei-Kashani and Cyrus Shahabi. SWAM: A family of access methods for similarity-search in peer-to-peer data networks. In *CIKM '04: Proceedings of the Thirteenth ACM conference on Information and knowledge management*, pages 304–313. ACM Press, 2004.
3. Michal Batko, David Novak, Fabrizio Falchi, and Pavel Zezula. On scalability of the similarity search in the world of peers. In *Proceedings of First International Conference on Scalable Information Systems (INFOSCALE 2006), Hong Kong, May 30 - June 1*, pages 1–12. ACM Press, 2006.
4. Matthias Bender, Sebastian Michel, Peter Triantafillou, Gerhard Weikum, and Christian Zimmer. MINERVA: Collaborative P2P search. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 1263–1266. VLDB Endowment, 2005.
5. Paolo Ciaccia, Marco Patella, and Pavel Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pages 426–435, 1997.
6. Vlastislav Dohnal, Claudio Gennaro, Pasquale Savino, and Pavel Zezula. D-index: Distance searching index for metric data sets. *Multimedia Tools and Applications*, 21(1):9–33, 2003.
7. Prasanna Ganesan, Beverly Yang, and Hector Garcia-Molina. One torus to rule them all: Multi-dimensional queries in P2P systems. In *WebDB '04: Proceedings of the 7th International Workshop on the Web and Databases*, pages 19–24, New York, NY, USA, 2004. ACM Press.
8. Alessandro Linari and Gerhard Weikum. Efficient peer-to-peer semantic overlay networks based on statistical language models. In *P2PIR '06: Proceedings of the international workshop on Information retrieval in peer-to-peer networks*, pages 9–16, New York, NY, USA, 2006. ACM Press.
9. Peter R. Monge and Noshir S. Contractor. *Theories of Communication Networks*. Oxford University Press, April 2003.
10. Egemen Tanin, Deepa Nayar, and Hanan Samet. An efficient nearest neighbor algorithm for P2P settings. In *Proceedings of the 2005 national conference on Digital government research*, pages 21–28. Digital Government Research Center, 2005.
11. Stanley Wasserman, Katherine Faust, and Dawn Iacobucci. *Social Network Analysis : Methods and Applications (Structural Analysis in the Social Sciences)*. Cambridge University Press, November 1994.
12. D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.
13. Pavel Zezula, Giuseppe Amato, Vlastislav Dohnal, and Michal Batko. *Similarity Search: The Metric Space Approach*, volume 32 of *Advances in Database Systems*. Springer, 2005.