Combinatorial Framework for Similarity Search

Yury Lifshits
Yahoo! Research
Santa Clara, CA, USA
http://yury.name
yury@yury.name

***Abstract*—We present an overview of the combinatorial framework for similarity search. An algorithm is combinatorial if only direct comparisons between two pairwise similarity values are allowed. Namely, the input dataset is represented by a comparison oracle that given any three points x, y, z answers whether y or z is closer to x. We assume that the similarity order of the dataset satisfies the four variations of the following *disorder inequality*: if x is the a'th most similar object to y and y is the b'th most similar object to z, then x is among the $D(a + b)$ most similar objects to z, where D is a relatively small disorder constant. Combinatorial algorithms for nearest neighbor search have two important advantages: (1) they do not map similarity values to artificial distance values and do not use triangle inequality for the latter, and (2) they work for arbitrarily complicated data representations and similarity functions.**

Ranwalk, the first known combinatorial solution for nearest neighbors, is randomized, exact, zero-error algorithm with query time that is logarithmic in number of objects. But Ranwalk preprocessing time is quadratic. Later on, another solution, called *combinatorial nets*, was discovered. It is deterministic and exact algorithm with near-linear time and space complexity of preprocessing, and near-logarithmic time complexity of search. Combinatorial nets also have a number of side applications. For near-duplicate detection they lead to the first known deterministic algorithm that requires just near-linear time + time proportional to the size of output. For any dataset with small disorder combinatorial nets can be used to construct a *visibility graph*: the one in which greedy routing deterministically converges to the nearest neighbor of a target in logarithmic number of steps. The later result is the first known work-around for Navarro's impossibility of generalizing Delaunay graphs.

***Keywords*-nearest neighbors, similarity search**

I. Introduction

Statements of many algorithmic problems including similarity search start with "Given n objects in metric space...". To formalize these problems we have to (1) define the data representation and (2) make some assumptions about the dataset. So far many algorithms were designed for specific data model, such as Hamming cube [51], or for so called *general metric spaces* [68] with assumptions that some intrinsic dimension [21], [44], [49] is small.

However, there is a large class of instances that requires principally new approach. We say that a dataset has the *separation effect* if the ratio between the largest and the smallest pairwise distances is below 2. Take for example the similarity measure "number of joint friends" for members of a social network. In a typical case there are at most 20 out 200 friends in common. This give us 0.9 in the so called Jaccard distance for two really similar people and 1.0 for completely unrelated. The similar story is observed with texts (similarity by words), webpages (similarity by referring sites), movies (similarity by reviewers and ratings) and so on. In all this cases all objects appeared to be "unique". Unfortunately, datasets with separation effect always have the doubling dimension close to the worst possible. Also the branch-and-bound techniques have to visit all objects, because the metric triangle inequality produces no new knowledge on yet uncomputed distances.

Addressing the challenge of separation effect, the *combinatorial framework* emerged in the papers by Goyal, Lifshits and Schütze [32] and by Lifshits and Zhang [55]. Instead of the availability of the distance between any two points in dataset S, suppose we are only given a *comparison oracle* O which can tells whether y is closer to x or z is closer to x. In other words, it discards all the metric information between points, with only the relative closeness comparisons remained. We define $\mathrm{rank}_x(y)$ to be an integer position of y in the list of all object in the dataset sorted by closeness to x. The central assumption of the combinatorial framework is that $R(x, y) = \max(\mathrm{rank}_x(y), \mathrm{rank}_y(x))$ is a quasimetric with some relatively small *disorder constant* D. More formally, we assume that the following *disorder inequality* holds: $R(x, z) \leq D(R(x, y) + R(y, z))$. This assumption is based on experimental observations on Reuters corpus of news articles [32].

In this paper we present a formal description of the combinatorial framework, list all known results and provide an extensive agenda for further research.

II. The Combinatorial Framework

The combinatorial framework is a model of computation for problems dealing with informal concept of "closeness". In literature both terms of similarity and distance are used: (1) when points are close to each other we say "distance is small" or "similarity is high", (2) for distance functions, the triangle inequality is usually assumed. The combinatorial framework is focused on the notion of similarity by not assuming the triangle inequality for distance values.

Instead of providing exact values of similarity function σ we give an access to the *comparison oracle* which, for each query (x, y, z), tells whether $\sigma(x, y) < \sigma(x, z)$ or $\sigma(x, y) > \sigma(x, z)$. For simplicity, throughout this paper we assume that no tie exists.

Consider some dataset S of n points. For each point x, all other $n - 1$ points y can be sorted by $\sigma(x, y)$. We call all these n sorted lists together to be a *similarity order* for the dataset S. Indeed, this is the only information we can get from the comparison oracle. For any points $x, y \in S$, define $\text{rank}_{x,S}(y)$ to be the position of y when the elements of S are sorted according to similarity to x in the deceasing order. We omit S when it is clear from the context. For convenience let $\text{rank}_x(x) = 0$. We define a *combinatorial ball* as $B(x, r) = \{y : \text{rank}_x(y) < r\}$.

Once we waive the real values of similarities, we lost almost all the knowledge about the dataset. In order to make algorithmic problems tractable we introduce a consistency assumption for the similarity order. Namely, we assume that the rank function satisfies the following weak triangle inequalities:

$$\text{rank}_x(y) \leq D(\text{rank}_x(z) + \text{rank}_y(z)), \quad (1)$$
$$\text{rank}_x(y) \leq D(\text{rank}_x(z) + \text{rank}_z(y)), \quad (2)$$
$$\text{rank}_x(y) \leq D(\text{rank}_z(x) + \text{rank}_y(z)), \quad (3)$$
$$\text{rank}_x(y) \leq D(\text{rank}_z(x) + \text{rank}_z(y)). \quad (4)$$

The above inequalities are called the *disorder inequalities*, and the minimal constant D making them true is called the *disorder constant*. Some comments are in order. First, we list all four possible disorder inequalities simply because there seems not to be any strong reason that some are more reasonable than others. Second, by putting $z = y$ in the first inequality, we have that

$$\text{rank}_x(y) \leq D\text{rank}_y(x). \quad (5)$$

Thus any one of the above four disorder inequalities implies the other three with a constant $D' = D^2$. The disorder constant D captures the intuitive notion of intrinsic dimension. The following lemma illustrate this correspondence in the case of fixed dimensions:

Lemma ([32]). *For sufficiently large hypercube in the d-dimensional integer grid $\mathbb{Z}^d$ with Euclidean distance, its disorder constant is 2^{d-1} up to a multiplicative constant close to one.*

Let us compare the concept of disorder to distance-based notions of doubling dimension and expansion rate. We need the following definitions.

Definition 1. *(disorder dimension) Let (S, σ) be a set in similarity space and $D(S)$ be the disorder constant of S. Then we call $(1 + \log D)$ the* disorder dimension *of S.*

Definition 2. *The metric ball $MB(p, r)$ is the set of all possible objects within distance r from p. The expansion rate of a set S in a metric space is the minimal number c such that $\forall p \in S$, $\forall r$, it holds that $|MB(p, 2r) \cap S| \leq c|MB(p, r) \cap S|$.*

Lemma. *(Small expansion rates imply small disorder constants [55]) For any n-point set S with the expansion rate c in a metric space, all the four disorder inequalities are satisfied with $D = c^2$.*

Lemma. *(Doubling effect in the combinatorial framework [55]) Consider a set S of n points satisfying the disorder inequalities. Then for any point p and integer r the combinatorial ball $B(p, 2r)$ can be covered by at most $O(D^2)$ combinatorial balls of radius r.*

Indyk's examples. In email correspondence with the author Indyk showed a dataset with small doubling constant but large disorder constant, and another dataset with small disorder constant but large doubling constant and expansion rate.

Small doubling constant but large disorder: A union of (slightly perturbed) points on a circle, with its center z. Here, the doubling dimension is constant, while the disorder dimension is high. Specifically, the two nearest neighbors x and y of z could be a pair of antipodal points, but $\text{rank}_x(y)$ could be very large.

Small disorder but large doubling constant: A set of n points $p_1, ..., p_n$, such that, if $i \neq j, d(p_i, p_j) = 10n + |i - j|$. The rank for ties are broken arbitrarily. It is not hard to see that $i, j,\ |i - j| \leq \text{rank}_{p_i}(p_j) \leq 2|i - j|$, therefore, the disorder constant is at most 2. However, note that the points are almost equidistant, in which case the doubling constant and the expansion rate is close to n.

We think that the combinatorial framework is worth studying for a number of reasons:

- **New solvable classes.** Problems like nearest neighbors are intractable in general. Thus, researchers are looking for additional assumptions and subclasses of the input data that make the problem easier. The combinatorial framework provides such a new subclass: all datasets with sublinear disorder constant. In particular, we can solve nearest neighbors even for some datasets not satisfying the metric triangle inequality.
- **Addressing heterogeneous data models.** In many modern applications object representations are far from simple abstractions like Euclidean space. For instance, consider a description of a blog: language, geographic location (dictionary parameters), age, number of posts (numerical parameters), referring links and reader list (graph parameters), text of profile and posts (text parameters) and posting timestamps (time parameters). Such a heterogeneous description needs a quite complicated similarity function. E.g. it can include manually

defined logical rules and threshold functions. The combinatorial framework can work with arbitrarily complicated similarity functions without any customization. This is an important advantage comparing to well-established locality-sensitive hashing approach [43] that requires designing new hash functions for every data model.

- **Using only relative order.** In many applications, designing a similarity (distance) function is challenge by itself. With the combinatorial framework this task becomes easier. Say, when similarity itself is a subject to learn [8], we need only comparative training information.
- **Non-reducibility to studied models.** The combinatorial framework draw some inspiration from previously studied models, most notably doubling dimension [34], [50], [49]. Despite some known results on repairing quasimetrics [57], we do not see a way to reduce this new model to the old ones. If you try to define a metric for some similarity order satisfying the disorder inequality you can achieve the metric triangle inequality *or* a small doubling dimension, but not both. Also, in the combinatorial framework, the full similarity order is not given as a part of the input. Thus, introducing any new rank-based distance can be computationally very expensive.

III. Combinatorial Algorithms

Randomized algorithms. Two new randomized algorithms for exact nearest neighbor search, Ranwalk and Arwalk were presented in the seminal paper on the combinatorial framework [32]. They are the first ones known to be *purely combinatorial* in the sense defined above. They bear some resemblance to the greedy search algorithms for small world networks (see, e.g., [45]).

Ranwalk performs a random walk in the search phase. It requires $\mathcal{O}(n^2)$ preprocessing space, $\mathcal{O}(n^2 \log n)$ preprocessing time and uses expected $\mathcal{O}(D \log n \log\log n + D^3)$ time for answering a query. It always produces the correct answer.

The Arwalk algorithm (walk via navigation array) requires $\mathcal{O}(nD \log n \log\log n)$ preprocessing space, $\mathcal{O}(n^2 \log n)$ preprocessing time and uses $\mathcal{O}(D \log n (\log\log n + \log 1/\delta) + D^3)$ time for answering a query. For every query it produces the correct answer with probability at least $1 - \delta$. The underlying data structure, called *navigation array*, is a $n \times D' \times \log n$ table of pointers to points in the database S. Informally, for every point $x \in S$ and every $k \leq \log_2 n$ we keep pointers to $D' = D \log\log n$ random points in the $n/2^k$ neighborhood of x.

The analysis of Arwalk shows that similarity search is tractable (near-linear preprocessing space, near-logarithmic query time) when the disorder dimension $\log D + 1$ is at most $\log\log n$.

Deterministic and exact algorithm for nearest neighbor search. Combinatorial nets [55] have a deterministic preprocessing algorithm of $D^7 n \log^2 n$ time complexity and a search algorithm of $D^4 \log n$ time complexity. Note that the subquadratic time complexity requires a computation even less than obtaining the comparison information between all pairs.

Deterministic discovery of all near-duplicates. Detecting and eliminating replicated documents is recognized as one of the central problems for search engines [14], [9], [17], [47]. Notice that this is a typical situation of the separation effect: Almost all distances in the range $[\frac{1}{2}, 1]$, since 50% similarity is considered to be threshold for duplicates and thus all "original" documents have over one half distances between each other. Thus the metric triangle inequality is non-informative and doubling constant/expansion rate are close to the maximal possible. It was shown [55] that the combinatorial framework provides efficient solution to near-duplicates. Assume that the similarity oracle can also answer queries "Whether similarity between x and y is above the duplicate threshold or not?" Then all pairs of near-duplicates can be *deterministically* found in time $poly(D)(n \log^2 n + |Output|)$.

Small world design. Starting from the seminal paper of Kleinberg [45], navigating schemes for various small world models were intensively studied. Designing peer-to-peer network protocols such as Meridian [67] raises the following Small World Design problem: given n nodes in some metric space, construct a small number of out-going connections for every node such that from any given starting point greedy routing leads to the nearest neighbor of the target point. This problem was solved using the combinatorial framework [55]. Namely, for any dataset with disorder constant D, its *visibility graph* can be constructed in $poly(D) n \log^2 n$ time and with $\mathcal{O}(D^4 \log n)$ out-degrees. The greedy-style local search in visibility graph finds the exact nearest neighbor of the target destination by at most $\log n$ moves on the graph edges. And as a corollary, visibility graph creates $\mathcal{O}(D^4 \log n)$-to-check certificates of being the exact nearest neighbor. Such certificates are not known for previously studied models.

The Voronoi diagram and the closely related Delaunay graph have the following "local-implies-global" property: If some point in the dataset is closer to query q than the center of any adjacent Voronoi cell (i.e. any Delaunay neighbor), then this point is the nearest neighbor of q. Thus, greedy walk through Delaunay graph will eventually lead to the nearest neighbor. In 1999 Navarro [60] made an attempt to find a structure similar to Delaunay graph for a general metric space. Unfortunately, he came up with the following *negative* result:

Given the pairwise distances for a finite subset S of an *unknown metric space U, for each $a, b \in S$ there exists a choice for U where a and b are connected in the Delaunay graph of S.*

The combinatorial construction of visibility graph avoids a similar negative barrier by using the concept of *visible neighbors* instead of Delaunay neighbors.

IV. Related Work

Similarity search in small intrinsic dimension. Let us compare known combinatorial results to studies of growth restricting metrics. Combinatorial nets are deterministic, as opposed to the randomized data structure construction in Hildrum et al. [36] (based on Karger and Ruhl [44] and Plaxton et al. [64]). Comparing to cover-trees [11] combinatorial nets have similar preprocessing and search time complexity. The key advantage of combinatorial approach is that the "tractable" family of datasets (those with small disorder constant) is a superset of datasets with bounded growth rate.

Speaking about doubling dimension, combinatorial nets find the exact nearest neighbor, compared to the approximate ones by Krauthgamer and Lee [49], [50] and by Cole and Gottlieb [22]. The near-linear construction of combinatorial nets is deterministic while the data structure constructions by Clarkson [20] and Har-Peled and Mendel [34] are randomized.

Two more results from [11] and [22] are not yet matched within the combinatorial framework: linear space bound and handling insertions/delitions.

Near-duplicates. Compared to the results based on the minhashing [13], combinatorial solution has deterministic subquadratic guarantee for running time and works for arbitrary similarity functions.

Small world design. Early work on this problem includes Arya and Mount's discussion of a greedy "routing" scheme for approximate nearest neighbor search in R^d [7]. Recent algorithms by Fraigniaud, Lebhar and Lotker [30] and Slivkins [66] provide efficient solutions assuming doubling dimension is $\mathcal{O}(\log\log n)$. Moreover, [30] provides counterexamples for larger doubling dimensions. Combinatorial solution has a couple of advantages with respect to the work above. First, it has *deterministic guarantee* for convergence in logarithmic number of steps. Second, it has no dependence on the aspect ratio, i.e. ratio between maximal and minimal distances.

V. Directions for Further Work

The combinatorial framework is just the first implementation of *metric regularization idea*. Instead of working with numerical values of distances, we use ranks. Thus, we "redefine" or "regularize" initial metric preserving similarity order and could obtain nicer distribution properties at the same time. Our results indicate that this is, indeed, a very powerful tool for proximity problems.

Numerous questions remain to be answered. Here is a list of problems that one can further study.

- **Improving combinatorial framework.** What if the D in the disorder inequality is small *on average* instead of in the worst case? If this is too restricting, what is the largest fraction of bad triples we can handle? In some scenarios, not all pairs are comparable, so for each x, the other $n-1$ points only form a partial order. How should we change our assumption and algorithms? In general, it is important to find a combinatorial notion of dimension that is robust to exceptions and perturbations but still provide efficient algorithms.
- **Improving known algorithms.** Can the complexity of known algorithms be further decreased? When disorder constant is not given, can we compute (or at least obtain some probabilistic guarantees) its value in subquadratic time? How to handle insertions and deletions, (important for using visibility graph in network design)? Recall, insertions and deletions can be efficiently supported in growth-restricted metric [22]. Can we avoid reconstructing the whole data structure when parameters of similarity function has been changed? This is important since distance metric is sometimes not fixed but a subject of *learning* [31]. What is the I/O complexity of combinatorial method for datasets that does not fit to main memory? How it should be implemented on distributed architectures like MapReduce [25]?
- **Further experiments.** Compute disorder constant for classic public datasets. Is it empirically true that similarity order is more consistent for small ranks but has more perturbations on large values? Find the actual size of combinatorial nets needed to cover them. Test our algorithms in the context of MESSIF project [10]. Suggest heuristics (e.g. using Gonzalez sequence [33] for net construction) that can be useful. In practice we do not know disorder in advance. How does it change the implementation?
- **Utilizing combinatorial framework.** Can other problems dealing with distances be stated and efficiently solved in combinatorial framework? It seems that for some applications replacing distances by ranks can be meaningful. In particular, it is interesting to consider linear arrangement problem [18], closest pairs [24], distance labelling [66], shortest paths [3], detecting communities [61], and dimensionality reduction [15]. There is a variant of low-distortion embeddings focusing on local distortion guarantees [1]. How to construct embeddings with low rank-distortion? The kNN rule is just one of solutions for automatic classification problem. Can other approaches also be formalized within combinatorial framework? How known methods should be modified for bipartite problems (like users-

ads matching [54]), where similarities are defined only for bichromatic pairs?

Also, is it possible to "translate" classic techniques for similarity search like branch-and-bound [19], [38] hashing [5], [51] or random projections [28] to "combinatorial language"? What kind of guarantees can we prove for them using disorder inequality?

- **Disorder vs. doubling.** The combinatorial framework leads to new results that were not known in other models: subquadratic deterministic detection of near duplicates, short certificates for being nearest neighbor, generalized analog of Delaunay graph. Can similar results be proven using doubling dimension?

  In the other direction, it was shown recently, that a dataset can be embedded into its doubling dimension [2]. Does the similar statement applies for disorder?

  Finally, can combinatorial and distance-based methods be combined in some useful way? We post the following *unification challenge*:

  Is there a general framework for efficient solutions to similarity problems that contains both doubling dimension and the combinatorial approach as specific subcases?

- **Metric regularizations.** When is it possible to redefine metric on some dataset such that similarity order is preserved while doubling dimension is decreased?
- **Disorder of random sets.** Compute disorder values and more generally perform combinatorial analysis for some modelling examples. Study randomized datasets: (1) n random points on d-dimensional sphere, (2) n random strings of some fixed length in σ-size alphabet for Hamming/edit distance, (3) random texts generated by Zipf model [39], and (4) preferential attachment graphs with "number of joint friends" similarity.
- **Lower bounds.** Is it possible to prove lower bounds on preprocessing and query complexities in some "black-box" model of computation? Can we adapt techniques from other negative results [12], [50], [65]?

Acknowledgment

Author thanks Navin Goyal, Hinrich Schütze and Shengyu Zhang for their invaluable help in studies of the combinatorial framework.

References

[1] I. Abraham and Y. Bartal and O. Neiman. Local Embeddings of Metric Spaces. *STOC'07.*

[2] I. Abraham and Y. Bartal and O. Neiman. Embedding Metric Spaces in their Intrinsic Dimension. *SODA'08.*

[3] I. Abraham, C. Gavoille, A.V. Goldberg, and D. Malkhi. Routing in networks with low doubling dimension. *ICDCS'06.*

[4] A. Agarwal and S. Chakrabarti. Learning random walks to rank nodes in graphs. *ICML'07.*

[5] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *FOCS'06*, 2006.

[6] R. Angelova and G. Weikum. Graph-based text classification: Learn from your neighbors. *SIGIR'06.*

[7] S. Arya and D. M. Mount. Approximate nearest neighbor queries in fixed dimensions. *SODA'93*, pages 271–280, 1993.

[8] M.-F. Balcan, A. Blum, and S. Vempala. A discriminative framework for clustering via similarity functions. *STOC'08.*

[9] Z. Bar-Yossef, I. Keidar, and U. Schonfeld. Do not crawl in the DUST: different urls with similar text. *WWW'07.*

[10] M. Batko, D. Novak, and P. Zezula. MESSIF: Metric similarity search implementation framework. *DELOS'07.*

[11] A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. *ICML'06.*

[12] A. Borodin and R. Ostrovsky and Y. Rabani. Lower bounds for high dimensional nearest neighbor search and related problems. *STOC'99.*

[13] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. *STOC'98.*

[14] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the Web. *WWW'97.*

[15] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Transactions on Database Systems*, 2002.

[16] T.H.H. Chan, M. Dinitz, and A. Gupta. Spanners with slack. *ESA'06.*

[17] M. Charikar. Similarity estimation techniques from rounding algorithms. *STOC'02.*

[18] M. Charikar, K. Makarychev, and Y. Makarychev. A divide and conquer algorithm for d-dimensional linear arrangement. *SODA'07.*

[19] E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín. Searching in metric spaces. *ACM Computing Surveys*, 2001.

[20] K.L. Clarkson. Nearest neighbor queries in metric spaces. *Discrete and Computational Geometry*, 1999.

[21] K.L. Clarkson. Nearest-neighbor searching and metric space dimensions. In Nearest-Neighbor Methods for Learning and Vision: Theory and Practice, MIT Press, 2006.

[22] R. Cole and L.-A. Gottlieb. Searching dynamic point sets in spaces with bounded doubling dimension. *STOC'06.*

[23] R. Cole, L.-A. Gottlieb, and M. Lewenstein. Dictionary matching and indexing with errors and don't cares. *STOC'04.*

[24] A. Corral, Y. Manolopoulos, Y. Theodoridis, and M. Vassilakopoulos. Closest pair queries in spatial databases. *SIGMOD'00.*

[25] J. Dean, S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. *OSDI'04.*

[26] J. Dean and M. R. Henzinger. Finding related web pages in the world wide web. *WWW'99.*

[27] M. Dubinko, R. Kumar, J. Magnani, J. Novak, P. Raghavan, and A. Tomkins. Visualizing tags over time. *WWW'06.*

[28] R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. *SIGMOD'03.*

[29] D. Fetterly, M. Manasse, and M. Najork. Detecting phrase-level duplication on the world wide web. *SIGIR'05.*

[30] P. Fraigniaud, E. Lebhar, and Z. Lotker. A doubling dimension threshold $\theta(\log \log n)$ for augmented graph navigability. *ESA'06.*

[31] A. Frome and Y. Singer and F. Sha and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. *ICCV'07.*

[32] N. Goyal, Y. Lifshits, and H. Schütze. Disorder inequality: A combinatorial approach to nearest neighbor search. *WSDM'08.*

[33] T.F. Gonzalez. Clustering to minimize the maximum intercluster distance *Theoretical Computer Science*, 1985.

[34] S. Har-Peled and M. Mendel. Fast construction of nets in low dimensional metrics, and their applications. *SoCG'05, SIAM Journal on Computing 2006.*

[35] M. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. Measuring index quality using random walks on the web. *WWW'99.*

[36] K. Hildrum, J. Kubiatowicz, S. Ma, and S. Rao. A note on the nearest neighbor in growth-restricted metrics. *SODA'04.*

[37] E. A. Hirsch and A. Kojevnikov. Unitwalk: A new SAT solver that uses local search guided by unit clause elimination. *Annals of Mathematics and Artificial Intelligence*, 2005.

[38] G. Hjaltason and H. Samet. Index-driven similarity search in metric spaces. *ACM Transactions on Database Systems*, 2003.

[39] B. Hoffmann, Y. Lifshits, and D. Nowotka. Maximal intersection queries in randomized graph models. *CSR'07.*

[40] S. Ilyinsky, M. Kuzmin, A. Melkov, and I. Segalovich. An efficient method to detect duplicates of web documents with the use of inverted index. *WWW'02.*

[41] P. Indyk. Nearest neighbors in high-dimensional spaces. Chapter 39 of Handbook of Discrete and Computational Geometry, 2004.

[42] P. Indyk. Private communication, 2007.

[43] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. *STOC'98.*

[44] D. R. Karger and M. Ruhl. Finding nearest neighbors in growth-restricted metrics. *STOC'02.*

[45] J. Kleinberg. The small-world phenomenon: an algorithmic perspective. *STOC '00.*

[46] J. M. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. *STOC'97.*

[47] A. Kolcz, A. Chowdhury, and J. Alspector. Improved robustness of signature-based near-replica detection via lexicon randomization. *KDD'04.*

[48] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. GroupLens: applying collaborative filtering to usenet news. *Commun. ACM*, 1997.

[49] R. Krauthgamer and J. R. Lee. Navigating nets: simple algorithms for proximity search. *SODA'04.*

[50] R. Krauthgamer and J. R. Lee. The black-box complexity of nearest-neighbor search. *Theoretical Computer Science*, 2005.

[51] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *STOC'98, SIAM J. Comput. 2000.*

[52] R. Lempel and S. Moran. Rank-stability and rank-similarity of link-basedweb ranking algorithms in authority-connected graphs. *Information Retrieval*, 2005.

[53] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 2004.

[54] Y. Lifshits and D. Nowotka. Estimation of the click volume by large scale regression analysis. *CSR'07.*

[55] Y. Lifshits and S. Zhang. Combinatorial Algorithms for Nearest Neighbors, Ner-Duplicates and Small World Design. *SODA'09.*

[56] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *Internet Computing*, 2003.

[57] R. Macíyas and C. Segovia. Lipschitz functions on spaces of homogeneous type. *Advances in Mathematics*, 1979.

[58] C. D. Manning, P. Raghavan, and H. Schütze. Introduction to Information Retrieval. *Cambridge University Press*, 2008.

[59] C. D. Manning and H. Schütze. Foundations of Statistical Natural Language Processing. *MIT Press*, 1999.

[60] G. Navarro. Searching in metric spaces by spatial approximation. *SPIRE'99, J. VLDB 2002.*

[61] M.E.J. Newman. Detecting community structure in networks. *The European Physical Journal B-Condensed Matter*, 2004.

[62] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. *WWW'06.*

[63] M. T. Orchard. A fast nearest-neighbor search algorithm. *ICASSP'91.*

[64] C.G. Plaxton, R. Rajaraman, and A.W. Richa. Accessing nearby copies of replicated objects in a distributed environment. *Theory Comput. Syst.*, 1999.

[65] C. Sahinalp and A. Utis. Hardness of string similarity search and other indexing problems. *ICALP'04*.

[66] A. Slivkins. Distance estimation and object location via rings of neighbors. *PODC'05, J. Distributed Computing 2007*.

[67] B. Wong, A. Slivkins, and E.G. Sirer. Meridian: A lightweight network location service without virtual coordinates. *SIGCOMM'05*, 2005.

[68] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search - The Metric Space Approach*. Springer, 2006.