Proposal Summary

This Career Development Plan proposes an integrated collection of research and educational activities focused on algorithms for high-dimensional geometric problems. Geometric computing with high-dimensional data is of crucial importance to many areas of computer science, including machine learning, data mining, databases and information retrieval, computer vision and computational biology. Example problems in this area are: nearest neighbor search, many variants of data clustering, and discovering linear structure of the data (e.g. via Principal Component Analysis (PCA)). Unfortunately, the classical geometric algorithms for many of these problems do not scale well with the dimension. For example, the running times of the classical algorithms for the nearest neighbor search depend *exponentially* on the dimension, which makes them inefficient for dimension higher than, say, 20. This is unfortunate, since many applications involve number of dimensions anywhere from a few hundred to a few million.

In recent years, new, powerful techniques for solving these problems have been discovered, most notably dimensionality reduction and random sampling in geometric spaces. The algorithms obtained using those techniques enjoy very low (at most linear) dependence on the dimension, at the cost of providing approximate answers. The problems amenable to these techniques include nearest neighbor search, clustering and PCA. However, the algorithmic solutions to these problems still possess (sometimes quite severe) limitations. In this proposal we identify the key directions for circumventing these limitations and making the algorithms fully satisfying, both in theory and in practice.

The techniques which led to the development of the aforementioned algorithms are new and still not widely known. They include many tools which have been developed during 70s and 80s in the field of mathematics called *functional analysis*, and in particular in the *local theory of Banach spaces*. However, computer scientists became aware of those methods only very recently, and many of the fundamental results are still not widely known or used. Therefore, it is important to make these results accessible to large computer science audience so that they can be successfully used, investigated and developed. In the Education Plan section we outline a plan on how to make them more accessible to students as well as computer scientists.

1 Introduction

Geometric computing with high-dimensional data is of crucial importance to many areas of computer science, including machine learning, data mining, databases and information retrieval, computer vision and computational biology. Unfortunately, the classical geometric algorithms for many of the key problems (e.g., nearest neighbor, clustering) do not scale well with the dimension. In recent years, new powerful techniques for solving these problems have been discovered, most notably dimensionality reduction and random sampling in geometric spaces. The algorithms obtained using those techniques enjoy very low (at most linear) dependence on the dimension, at the cost of providing approximate answers. In the Research Plan section we describe the above results and techniques in greater detail and identify key problems which we believe can be successfully attacked using this approach.

The techniques which led to the development of the aforementioned algorithms are new and still not widely known. In the Education Plan section we outline a plan how to make them more accessible to students as well as computer scientists.

2 Research Plan

2.1 The context

Computing with massive and high-dimensional data is pervasive in many of today's applications. Such data arise whenever the application deals with large amounts of complex objects (text documents, images, web pages, genetic sequences, etc.) In order to handle such objects, one needs to extract important *features* of the objects, usually of numerical nature, and use vectors of such features to represent the objects. For example, the fundamental paradigm of *information retrieval* is that, for the purpose of measuring similarity between documents, a text document can be represented by a long vector of "counters", where each "counter" contains the number of occurrences of a specific word. Similarly, in the field of *visual information retrieval*, images are represented by features describing color and texture characteristics. In *computational biology*, features can include presence or absence of certain properties of a genetic sequence, or even the sequence itself if only base substitutions are allowed. In the field of *data mining*, features can include characteristics or identities of customers who bought a given product.

Once the feature representation of the data set has been extracted, most of the application tasks can be reduced to algorithmic problems involving a set of high-dimensional points. In most situations, those problems can be classified into the following three categories:

- Classification: In this case, each point is labeled as "positive" or "negative", and the goal is to design a classifier which, given a new point, assigns the "correct" label to it. This is often done by finding a hyperplane (or some higher dimensional surface) which separates "most" positive points from "most" negative points, and classifying a new point depending on which side of the hyperplane it lies. Alternatively, one can classify

the point using the label of its nearest neighbor. Both approaches involve well-defined algorithmic problems in high-dimensions.

- Data analysis: In this case, the data is unlabeled, but is conjectured to possess certain underlying structure which the user wants to discover. For example, the data could have a linear structure, i.e., be well-approximated by a low-dimensional hyperplane; in this case one would apply *Principal Component Analysis (PCA)* to discover such structure. Alternatively, the points could be *clustered*, i.e., partitioned into small number of groups, where points within the same groups are very close to each other. Again, both scenarios involve an algorithmic problem in high-dimensional space.

- Retrieval: Here, we are initially given a database of points/objects. The user can *query* the database, searching for a small number of interesting objects. In a very popular *similarity search* setting, the query itself is a point, and the database retrieves a small number of points closest to the query. As before, this can be reduced to an algorithmic problem (specifically, *nearest neighbor search*) in high dimensions.

Since the performance of the above applications depends heavily on the efficiency of its algorithmic components, improving the latter has been an important research goal in a variety of fields for over three decades. For example, some the algorithms for nearest neighbor search or clustering have been among of the first results obtained in the field of computational geometry. Unfortunately, the running times of those algorithms, both in theory and practice, depend *exponentially* on the dimension (e.g., see [AMN+94]), and so they are very inefficient in the above applications. This phenomenon, often referred to as "the curse of dimensionality", is widely conjectured to be inevitable, as long as one insists on answers which are always fully accurate. In particular, many variants of clustering problems are known to be NP-hard in high dimensions.

The running time of algorithms for Principal Component Analysis depends on the way the error of the linear approximation is measured. In the most common case of sum of squares of errors of individual points, the problem can be solved using Singular Value Decomposition (SVD) in $O(d^2n)$ time, where n denotes the number of points and d denotes the dimension. Although this running time is polynomial in d, its quadratic dependence on the dimension makes the algorithm inefficient when the number of dimensions is large[1] [FKV98].

The problem of finding optimal separating hyperplane is notoriously hard. Even when the error-free separation exists and therefore the solution can be found using linear programming, the algorithms have running times far from linear.

2.2 Approximate solutions: past work and open problems

In recent years, a significant progress in designing efficient algorithms for high-dimensional geometric problems has been made, by allowing the algorithms to return *approximate* answers.

[1] Alternatively, one can use Power or Lanczos method to solve SVD. In this case, the running time becomes $O(kdnI)$, where k is the dimension of the approximating hyperplane and I is the number of iterations of the algorithms. Again, the running time is far from linear.

The main tool used by these algorithms has been the *dimensionality reduction* technique. This approach allows one to reduce the dimension of the input space *exponentially* and still (approximately) preserve crucial properties of the input data. For example, the *Johnson-Lindenstrauss lemma* [JL84] guarantees that any set of n points in a high-dimensional Euclidean space can be quickly embedded into an $O(\log n/\epsilon^2)$-dimensional space, while preserving all distances up to a multiplicative factor of $(1+\epsilon)$, for any $\epsilon \in [0,1]$. The embedding can be in fact created by choosing a random linear mapping, and can be computed in $O(nd \log n/\epsilon^2)$ time. Thus, by applying dimensionality reduction first and then running an algorithm on the reduced space, one can solve problems deemed intractable otherwise. This approach is particularly useful for geometric algorithms which have running times or storage requirements exponential in the dimension. However, even if the dependence of the running time on the dimension is only polynomial (not exponential), substantial savings in the running time can be achieved if the dimensionality of the input space is large. Consequently, dimensionality reduction has been used as a building block in many theoretical results (discussed below), as well as in many applied scenarios, in particular by Ritter, Kaski and Kohonen [RK89, Kas98], who rediscovered a version of Johnson-Lindenstrauss lemma by themselves.

In case of nearest neighbor search and related problems, dimensionality reduction resulted in several approximate algorithms [Kle97, IM98, KOR98, Ind98, BOR99]. Those algorithms manage to achieve polynomial dependence of the running time on the dimension; see Section 2.3 for a more detailed description. The main idea behind those algorithms is to construct data structures which achieve at most exponential dependence on the dimension, and then reduce this dependence to polynomial via dimensionality reduction.

For the Singular Value Decomposition problem, several approximate algorithms are known [FKV98, DKFV99, AM01] (in addition to a huge set of exact or almost exact algorithms). The approximate algorithms run in at most linear (sometimes even sublinear) time and produce a k-dimensional approximation P' of the d-dimensional set of points P, such that the approximation error $|P - P'|$ exceeds the error achievable using the best k-dimensional approximation by an *additive* term depending on $|P|$. The error function $|\cdot|$ can be either the Frobenius or the Euclidean norm. The main tool used in those algorithms is a careful random sampling of the input points and/or the dimensions of the input space.

Many approximate algorithms for clustering in high-dimensional spaces have been discovered. In particular, polynomial-time algorithms for learning mixtures of Gaussians have been given in [Das99, AK01]. The latter algorithms use dimensionality reduction in order to remove the exponential dependence of the running time on the dimension. More general algorithms working in any *metric space* have been also designed, see e.g., [CGTS99, JV99, Ind99a, GMMO01] for the algorithms for the *k-median* problem.

The above results indicate that it is possible to achieve significantly improved running times for the algorithms in high-dimensional spaces provided we allow them to output approximate answers. However, much more remains to be done in order to leverage the full power of approximation in the context of high-dimensional problems. The following directions seem to be of particular interest:

1. Faster approximate nearest neighbor algorithms for l_2 norm. The currently known algorithms offer either very fast (polylogarithmic) query time using large storage, or sublinear (but polynomial) query time using fairly small storage (see Section 2.3 for more information). An ideal algorithm would have both fast query time and use small storage. Currently, there is no evidence that such an algorithm cannot exist. Having a fast algorithm for the *dynamic* approximate nearest neighbor would be of even greater value, since by the results of [GIV01] several other problems, such as closest pair, diameter and variants of clustering, can be reduced to the former problem with low overhead.

   The hope of achieving better algorithms for the nearest neighbor search is justifiable by the fact that, very recently, a new approximate algorithm for the *furthest neighbor search* has been obtained [Ind01]. This algorithm achieves better running time and storage requirements than the earlier algorithm of [GIV01], which was obtained by a reduction to the approximate nearest neighbor problem. It is plausible that the reduction can be *reversed*, thereby leading to a more efficient algorithm for nearest neighbor search.

2. Approximate nearest neighbor for general classes of spaces. All algorithms discovered so far are tailored to very specific spaces, namely l_p, $p \in [1 \ldots 2]$ [IM98, KOR98], and l_∞ [Ind98]. The algorithms for these two scenarios use very different techniques: the first one uses dimensionality reduction tools, while the second is based on divide and conquer approach. Is it possible to unify them somehow ? Ideally, one would like to obtain an algorithm which works for *all* normed spaces, or at least all l_p norms. Since such algorithms exist for low-dimensional spaces [AMN+94, AEIS99], it is plausible they could also exist in the high-dimensional case. In addition to the clear advantage of having one algorithm working for different settings, such a result would also yield the first algorithm for norms not covered by the existing algorithms. This is of large interest, since it has been observed (e.g., see [AAH01]) that for many data sets, metrics induced by l_p norms with $p < 1$ provide more meaningful estimation of the distances between the points than the standard l_1 or l_2 norms.

3. Faster algorithms for discovering linear structure of the data. The existing approximate SVD algorithms mentioned earlier have very efficient running times, but also incur large additive errors. In particular, they can report a solution with error arbitrarily larger than the error of the optimal approximation. Thus, it is of great interest to design fast algorithms which provide *multiplicative* approximation guarantees. Ideally, the algorithms should return solutions with cost at most $(1 + \epsilon)$ times the optimal cost.

   The scenario when the error is measured not by computing the sum of squares of the differences (but instead, say, by computing the maximum difference) is even less understood. Although several algorithms are known for the maximum difference case (e.g., see [HPV01] and the references therein), all of them have running times exponential in the dimension, which makes them inapplicable in the high-dimensional setting. Thus, obtaining efficient algorithms for these cases is an important open problem.

4. Algorithms for discovering non-linear structure of the data. In many settings, the structure of the data is significantly non-linear. Recently, several new algorithms which capture non-linear structure of the data have been discovered [RS00, TdSL00]. However, the approaches to this problem have so far been mostly heuristic, and almost no theoretical research in this area has been done (to the knowledge of the author). It is therefore of great importance to understand the intricacies of non-linear structure discovery, capture them in a common formal framework and design efficient algorithms for the formally posed problems.

5. More general dimensionality reduction techniques. As mentioned earlier, dimensionality reduction allows one to speed-up virtually *any* algorithm designed for high-dimensional spaces. Unfortunately, the aforementioned Johsnon-Lindenstrauss lemma holds only for the Euclidean space. For other norms, only fairly limited analogs of this lemma exist (cf. [Ind00b]). Extending dimensionality reduction techniques to non-Euclidean spaces would provide a very general tool for improving the efficiency of algorithms in those spaces.

2.3 Research accomplishments

The computational geometry problems in high-dimensional spaces and related problems have been focal points of my research ever since I was a graduate student at Stanford. I have numerous publications in leading conferences and journals of the areas. The publication [Ind00b] was awarded the Machtey Award, as the best student paper presented at the Symposium on Foundations of Computer Science FOCS'00. Some details of the primary technical results are given are given below.

In a paper written jointly with Rajeev Motwani [IM98], we gave the first known $(1+\epsilon)$-approximate algorithms for the nearest neighbor problem with *both* efficient query time and polynomial (in n) storage. The first algorithm (obtained independently in [KOR98] and presented at the same conference) had query time polynomial in $d + \log n$ and used storage polynomial in n. Thus we showed that one can obtain essentially optimal query time while avoiding the "curse of dimensionality". Also, from the technical perspective, the above was the first result achieving exponential improvement in time or space by using dimensionality reduction techniques. Unfortunately, the storage requirements were still too large for the algorithm to be practical. The second algorithm circumvents this problem by using only $n^{1+1/(1+\epsilon)}$ units of additional storage and still achieving a sublinear query time of $dn^{1/(1+\epsilon)}$. The latter algorithm is practical and is in many scenarios faster than previously known algorithms, sometimes by several orders of magnitude [GIM99]. Its variants have been since used by several researchers, for efficient comparison of genomic sequences [Buh01] and motif finding [BT01], clustering of web documents [HGI00] and data mining [CDF+00]. In a sequence of follow-up papers [Ind98, FCI99, Ind00a, GIV01] we addressed other algorithmic problems in high-dimensional geometric spaces. In particular, we focused on the class of *proximity problems* (e.g., closest pair, variants of clustering etc) which contains problems defined exclusively in terms of distances between the input points.

The above results gave rise to the following question: is geometric structure of the input necessary for obtaining fast algorithms for proximity problems ? In papers [Ind99a, Ind99b] I showed that the answer is *no*: several proximity problems, including k-median and other variants of clustering, can be solved efficiently if we only assume that the relevant distances satisfy *metric constraints.* The algorithms developed in the paper in fact have running time *sublinear* in the input size, meaning that they complete their task *without* full knowledge of their inputs. The results and techniques from this paper have been subsequently used and refined, e.g., in [GMMO01, Tho01].

My recent paper [Ind00b] addressed another problem of computation with sublinear resources, this time using sublinear *space.* In particular, it addresses the problem of maintaining an l_p norm of a d-dimensional vector x, under dynamic increments/decrements of its coordinates, using storage sublinear (in fact, polylogarithmic) in d. This problem is of large importance in several fields, in particular in databases, since it specializes to the problems of maintaining the number of distinct elements in a relation, or maintaining the size of its self-join. Owing to its importance, this problem was a topic of several research papers [FM85, AMS96, FKSV99, FS00]. In [Ind00b] I presented an algorithm which unifies (and in many cases improves) many of the earlier results. In addition, it provides the first known dimensionality reduction method for the l_1 norm. The main technical contribution of the paper is the use of *stable distributions* (also called *heavy-tailed* distributions or *power laws*), which recently attracted significant attention in many areas of computer science, but had not been used as an algorithmic tool before.

3 Education Plan

As a field using the language of mathematics, theoretical computer science benefits tremendously from discovering connections between the problems of its interest and other areas of mathematics. Examples of such discoveries, which became great success stories are: using discrete mathematics tools for the purpose of rigorous analysis of algorithms, basing modern cryptography on number-theoretic assumptions, or using methods from coding theory to characterize the computational power of interactive and probabilistically checkable proofs. It is of crucial importance that mathematical areas that provide useful algorithmic tools are identified, and its methods made accessible to computer science researchers.

In recent years, great progress for many algorithmic problems has been achieved using the method of *approximate embeddings.* This approach allows one to obtain an approximate solution to a problem defined over a "difficult" geometric (or metric) space, by embedding this space into an "easier" one. The dimensionality reduction technique, described earlier, is a special case of this method, where the "easy" space has significantly lower dimension than the "difficult" one.

Many of the approximate embedding tools have been developed during 70s and 80s in the field of mathematics called *functional analysis*, and in particular in the *local theory of Banach spaces.* However, computer scientists became aware of those methods only very recently, and many of the fundamental results are still not widely known or used. Therefore, it is

important to make those results accessible to large computer science audience so that they can be successfully used, investigated and developed.

Thus, my main educational goal for the next few years is to identify and disseminate the key tools and methods of approximate embeddings. As the first step, I have designed and taught (during Fall 2000) an advanced topics course entitled "Algorithmic aspects of embeddings". Thanks to the course, many advanced graduate students became familiar with the topic and already some of them obtained novel algorithmic results using the methods presented in the course. However, the advance nature of the course made it difficult for non-theory students to fully benefit from it. Therefore, during Fall 2001, I am planning to co-teach an introductory course on "Computational Geometry", which will include basic material on geometric embeddings as a part of its curriculum. Since the course is aimed at beginning graduate students and advanced undergraduates, it is likely to reach a much wider audience than a highly specialized advance topics course can hope to.

In addition to course development, I am planning to prepare several tutorials and surveys articles devoted to approximate embeddings. At this moment, I am preparing a tutorial on this topic, to be presented at the Symposium on Foundations of Computer Science, 2001. In addition, I have been recently invited to write two chapters for the upcoming 2nd edition of the "CRC Handbook of Discrete Computational Geometry". One of the chapters, entitled "Nearest neighbor in high dimensions and related problems", will contain many of the algorithmic applications of geometric embeddings, while the second chapter (co-authored with Jiri Matousek) will be entirely devoted to geometric embeddings of finite metric spaces.

3.1 Education accomplishments

My formal teaching experience includes 1 semester of teaching assistantship at Warsaw University (Poland), 2 semesters of teaching assistantship at Stanford, one advanced topics course and one undergraduate level course at MIT. The advanced topic course, devoted to "Algorithmic aspects of embeddings", has been entirely designed by myself. Additionally, I have presented over 20 seminars in research labs and universities all over the world, including U.S.A., Germany and Poland. I have been invited to teach a mini-course on "Nearest Neighbor and Other Problems in High-dimensional Computational Geometry"(six lectures) at AT&T Shannon Lab in Florham Park, 1999, and to present its one hour version during DIMACS Summer School of Data Mining (Rutgers University, Piscataway, NJ, 2001). I have also been invited to give a two hour long tutorial on "Algorithmic Aspects of Geometric Embeddings" at the Symposium on Foundations of Computer Science, 2001.

References

[AAH01] C. Aggarwal and D. Keim A. Hinneburg. On the surprising behavior of distance metrics in high dimensional spaces. *Proceedings of the International Conference on Database Theory*, pages 420–434, 2001.

[AEIS99] A.Amir, A. Efrat, P. Indyk, and H. Samet. Efficient regular data structures and algorithms for location and proximity problems. *Proceedings of the Symposium on Foundations of Computer Science*, 1999.

[AK01] S. Arora and R. Kannan. Learning a mixture of gaussians. *Proceedings of the Symposium on Theory of Computing*, 2001.

[AM01] D. Achlioptas and F. McSherry. Fast computation of low-rank approximations. *Proceedings of the Symposium on Theory of Computing*, 2001.

[AMN+94] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching. *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 573–582, 1994.

[AMS96] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Proceedings of the Symposium on Theory of Computing*, pages 20–29, 1996.

[BOR99] A. Borodin, R. Ostrovsky, and Y. Rabani. Subquadratic approximation algorithms for clustering problems in high dimensional spaces. *Proceedings of the Symposium on Theory of Computing*, 1999.

[BT01] J. Buhler and M. Tompa. Finding motifs using random projections. *Proceedings of the Annual International Conference on Computational Molecular Biology (RECOMB01)*, 2001.

[Buh01] J. Buhler. Efficient large-scale sequence comparison by locality-sensitive hashing. *Bioinformatics*, 17:419–428, 2001.

[CDF+00] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. Ullman, and C. Yang. Finding interesting associations without support prunning. *Proceedings of the 16th International Conference on Data Engineering (ICDE)*, 2000.

[CGTS99] M. Charikar, S. Guha, E. Tardos, and D. Shmoys. A constant factor approximation algorithm for the k-median problem. *Proceedings of the Symposium on Theory of Computing*, 1999.

[Das99] S. Dasgupta. Learning mixtures of gaussians. *Proceedings of the Symposium on Foundations of Computer Science*, pages 634–644, 1999.

[DKFV99] P. Drineas, R. Kannan, A. Frieze, and V. Vinay. Clustering in large graphs and matrices. *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 1999.

[FCI99] M. Farach-Colton and P. Indyk. Approximate nearest neighbor algorithms for hausdorff metrics via embeddings. *Proceedings of the Symposium on Foundations of Computer Science*, 1999.

[FKSV99] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. An approximate l1-difference algorithm for massive data streams. *Proceedings of the Symposium on Foundations of Computer Science*, 1999.

[FKV98] A. Frieze, R. Kannan, and S. Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *Proceedings of the Symposium on Foundations of Computer Science*, 1998.

[FM85] P. Flajolet and G. Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31:182–209, 1985.

[FS00] J. Fong and M. Strauss. An approximate lp-difference algorithm for massive data streams. *Proceedings of the Symposium on Theoretical Computer Science*, 2000.

[GIM99] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB)*, 1999.

[GIV01] A. Goel, P. Indyk, and K. Varadarajan. Reductions among high-dimensional geometric problems. *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2001.

[GMMO01] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams. *Proceedings of the Symposium on Theory of Computing*, 2001.

[HGI00] T. Haveliwala, A. Gionis, and P. Indyk. Scalable techniques for clustering the web. *WebDB Workshop*, 2000.

[HPV01] S. Har-Peled and K. Varadarajan. Approximate shape fitting via linearization. *Proceedings of the Symposium on Foundations of Computer Science*, 2001.

[IM98] P. Indyk and R. Motwani. Approximate nearest neighbor: towards removing the curse of dimensionality. *Proceedings of the Symposium on Theory of Computing*, 1998.

[Ind98] P. Indyk. On approximate nearest neighbors in l_∞ norm. *Journal of Computer and System Sciences, to appear. Preliminary version appeared in Proceedings of the Symposium on Foundations of Computer Science*, 1998.

[Ind99a] P. Indyk. Sublinear time algorithms for metric space problems. *Proceedings of the Symposium on Theory of Computing*, 1999.

[Ind99b] P. Indyk. A sublinear-time approximation scheme for clustering in metric spaces. *Proceedings of the Symposium on Foundations of Computer Science*, 1999.

[Ind00a] P. Indyk. Dimensionality reduction techniques for proximity problems. *Proceedings of the Ninth ACM-SIAM Symposium on Discrete Algorithms*, 2000.

[Ind00b] P. Indyk. Stable distributions, pseudorandom generators, embeddings and data stream computation. *Proceedings of the Symposium on Foundations of Computer Science*, 2000.

[Ind01] P. Indyk. Better algorithms for high-dimensional proximity problems via asymmetric embeddings. *Manuscript*, 2001.

[JL84] W.B. Johnson and J. Lindenstrauss. Extensions of lipshitz mapping into hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

[JV99] K. Jain and V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. *Proceedings of the Symposium on Foundations of Computer Science*, 1999.

[Kas98] S. Kaski. Dimensionality reduction by random mapping: Fast similarity computation for clustering. *Proceedings of IJCNN, International Joint Conference on Neural Networks*, 1998.

[Kle97] J. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, 1997.

[KOR98] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *Proceedings of the Thirtieth ACM Symposium on Theory of Computing*, pages 614–623, 1998.

[RK89] H. Ritter and T. Kohonen. Self-organizing semantic maps. *Biological Cybernetics*, 61:241–254, 1989.

[RS00] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[TdSL00] J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[Tho01] M. Thorup. Quick k-median, k-center, and facility location for sparse graphs. *Proceedings of the International Colloquium on Automata, Languages and Programming*, 2001.