MoBIoS Index: Support Distance-Based Queries in Bioinformatics

Rui Mao, Weijia Xu, Willard S. Willard, Smriti R. Ramakrishnan and Daniel P. Miranker
(Department of Computer Sciences & Center for Computational Biology and Bioinformatics, the University of Texas at Austin, USA)

Abstract: Given a database, answering a distance-based query means retrieving all the data objects in the database that are in close proximity to a query object. Proximity is defined by any metric distance function. Close can mean within a certain distance, a range query, k-nearest neighbor or the two in combination. Answering distance-based queries is a fundamental component of many biological applications, including sequence homology, protein identification by spectral database look-up and biomedical image retrieval. Most systems for retrieving similar biological data objects are domain-specific. For each new model of similarity the retrieval problem must be revisited. (Metric) distance-based indexing only requires that the data can be abstracted into metric space, enabling the reuse of the same software package for many problems. The Molecular Biological Information System (MoBIoS) is a metric-space DBMS targeting bioinformatics applications. In this paper we describe the programmatic interface for MoBIoS index methods. In addition to built-in metrics, the interfaces enable users to integrate new metrics. The system supports four different retrieval methods. We characterize these methods and their applicability to different problems.

Keywords: Distance-based indexing, similarity query; bioinformatics; MoBIoS; Metric space; Retrieval method

1 Introduction

The similarity between two complex data objects may be modeled by a domain-specific distance function. By definition, the smaller the distance, the more similar the two data objects. Since an object is most similar to itself, the distance from an object to itself is zero. When using distance to define similarity, there are two basic types of queries, range and k-nearest neighbor (k-NN):

Definition 1: *Range query R(q,r)*[4, 5]: Given a query object q, find all data objects *x* in the database within distance r to q, i.e., $d(q, x) \leq r$. r is called the *radius* of the range query.

Examples of a range queries using Euclidean distance and Hamming distance are "find all the restaurants within 3 miles to my office", or "given a DNA 18-mer find all the 18-mers in the mice genome that differ by at most 10 mutations".

Definition 2: *k-nearest neighbor query kNN(q)* [4, 5]: Given a query object q, find the k closest data objects to q in the database.

Examples of a kNN query are "find the 3 closest restaurants to my office", or "find the 100 18-mers in the mice genome that are most similar to a particular DNA 18-mer". A kNN query can be systematically implemented by using successive range queries[4].

In the case that the distance function forms a metric, the triangle inequality can be exploited to create data structures where large data sets can be organized off-line to speed up on-line execution of these queries. In the case that the data structure is stored on disk, we call it *an index.*

Definition 3: A *metric space* [15] is a pair (M, d), where M is a nonempty set and $d: M \times M \rightarrow R^+ \cup \{0\}$, is a real-valued function, called a metric (distance oracle) on M, with the following properties:

(1) For all $x, y \in M$, $d(x, y) >= 0$ and $d(x, y) = 0$ if and only if $x = y$. (Positivity)

(2) For all $x, y \in M$, $d(x, y) = d(y, x)$. (Symmetry)

(3) For all $x, y, z \in M$, $d(x, y) + d(y, z) >= d(x, z)$. (Triangle Inequality)

The generality of the distance-based abstraction promises an opportunity where a single indexing method and its interface may support data retrieval for a large variety of unorthodox data types. In other work, this method has been applied to document retrieval and pattern recognition.

Similarity query plays a critical role in large biology databases where the data is complex and distance calculations are expensive. Many domain Specific solutions have been proposed, such as BLAST[1] for sequence homology and TurboSEQUEST[22] for protein identification using mass spectra data. However, each solution only works for its own domain.

Distance-based indexing is a general solution that can provide a uniform programming model for many types of biological data. It requires neither the domain information of the data, nor an interpretation of the data objects into a coordinate system. The primary requirement is a metric distance function which abstracts the data into a metric space. Distance-based indexing has been under tense study [5, 16].

The generality of distance-based indexing also results in its challenge. In particular, since there are no restrictions placed on the distance function, except that it is a metric, the distance function may encode a very high dimensional problem and the indexing method may suffer from the curse of dimensionality[2]. For example, the distance function may be the Euclidean distance between vectors in R^n. There are well known results that suggest it is not possible to create comparison-based index structures to accelerate distance-based queries on this data [17, 18].

In this paper we describe the structure and application of the programmatic interface to the MoBIoS distance-based index structure. It is integral to the storage manager for the Molecular Biological Information System (MoBIoS) [6, 11, 12]. Analogous to Geographic Information Systems (GIS) which integrate spatial indexes with relational DBMSs and extend SQL to support spatial queries, MoBIoS integrates the distance-based index with a relational DBMS and extends SQL to support similarity queries in general metric

Author: **Rui Mao**(1975.05-), Ph.D. student, Department of Computer Sciences, the University of Texas at Austin, USA. Research area: distance-based indexing, database, datamining and bioinformatics. **Weijia Xu, Willard S. Willard, Smriti R. Ramakrishnan**: Ph.D. student, Department of Computer Sciences, the University of Texas at Austin, USA. **Daniel P. Miranker**: Professor, Ph.D. student, Department of Computer Sciences, the University of Texas at Austin, USA. Email: {rmao, xwj, willard, smriti, miranker}@cs.utexas.edu

spaces. MoBIoS is built upon Mckoi [9], an open source JAVA DBMS.

This package is available as open source. It is based on multi-vantage point trees (MVPT) [3]. This algorithm was chosen as the result of a study where the performance of an algorithm from each of the three major classes of distance-based index algorithm was compared using a suite of biological databases. The original MVPT is a main-memory data structure. We implemented the disk-based MVP index by fitting each index node into a disk page. Lastly, we designed new bulkload heuristics to improve query performance[7].

A special property of our implementation is that once an index for a dataset is constructed, a user may employ any of four different retrieval methods. In addition to range and k-nn queries, the package support *range-limited k-nearest neighbor,* and *approximate range-limited k-nearest neighbor* queries. If the application proves to be high-dimensional or otherwise troublesome, these retrieval methods may prove effective.

Definition 4: *Radius-limited k-nearest neighbor query RkNN(q, r)* [20]: Given a query object q, find the k closest, and within distance r, points to q in the database.

Definition 5: *Approximate radius-limited k nearest neighbor query AkNN(q,r)* [23]: This query is very similar to RkNN except that the search process is terminated by some greedy heuristics that remarkably accelerates the search and still produces exact solution when there are less than k results within the limiting radius r.

Another basic problem in bioinformatics is that the rapid growth of the amount of biological data has made main-memory solutions obsolete. To support the data intensive applications, disk-based database management systems (DBMS) are in great demand.

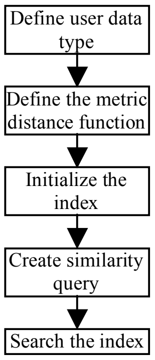
On top of the MoBIoS index, MoBIoS SQL (mSQL) is defined by extending standard SQL with support for similarity queries and biological functionalities[10]. Furthermore, several practical applications have been developed, with attractive empirical results.

The MoBIoS index package has recently been released. In section 2, we demonstrate the steps necessary to use the index by example. The properties of the similarity queries supported are discussed in Section 3 in the context of MoBIoS applications and empirical results.

2 Using the MoBIoS index

In this section, we show how to use the MoBIoS index for a similarity query of user defined data types. Only general steps are discussed here. Further details are available in the MoBIoS documentation. Some common data types, such as vector, DNA sequence, protein sequence and mass spectra, are already defined in MoBIoS. For these data types, there are command line tools to build the index and run the queries. See the MoBIoS website [12] for details.

Answering similarity queries in a divide-and-conquer distance-based indexing algorithm consists of an off-line phase and an on-line phase. First, the index is initialized off-line by materializing a hierarchical clustering of the data as a tree-based data structure. Associated with the root of every sub-tree is the bounding predicate satisfied by every data point in the cluster. Then, on-line data retrieval can exploit the triangle inequality to expedite the search procedure, amortizing the off-line construction cost and promising scalable performance. Given a query, if there is no data point that satisfies both the predicates of the index node and the query, that index node can be pruned to save distance calculations.



```
//Use the pre-defined Euclidean distance
1. Metric myMetric =
       LMetric.EuclideanDistanceMetric;
//Initialize a MVP index, data is a list of vectors
2. Index myIndex = new VPIndex(data, myMetric);
//create a range, kNN, RkNN, AkNN query
//respectively. v is a vector, the query object
3. Query q = new RangeQuery(v, 0.2);
4. Query q = new KNNQuery(v, 8);
5. Query q = new KNNQuery(v, 0.2, 8);
6. Query q = new KNNQuery(v, 0.2, 8, 2);
//search the index
7. Cursor c = myIndex.search(q);
8. while(c.hasNext()) System.out.println( c.next() );
```

Figure 1 (a), General steps of apply the MoBIoS index to distance-based queries of user defined data types; (b), Example code: distance-based queries of vectors

Generally, there are five steps to apply the MoBIoS index to user defined data types (Figure 1 (a)). An example application to vector data is given in Figure 1 (b).

1) Define the user type

The user data type must implement the JAVA Serializable interface for disk I/O purpose. In the example shown in Figure 1(b), the data type is mobios.type.DoubleVector, pre-defined in MoBIoS.

2) Define the metric distance

The user defined metric distance should implement mobios.dist.Metric. There is only one method to be implemented by the user distance function:

double getDistance(Object first, Object second)

This method takes two data object and returns the distance between them. In Figure 1 (b), line 1 shows that the distance for vectors.

3) Initialize the index

To build an index, the user just needs to create an instance of VPIndex with the metric distance and a list of data. The instance can be searched immediately or serialized to disk for future use. The following is the constructor and comment of VPIndex.

public VPIndex(List data,Metric metric)

Parameters:

data - the List of data to build the index

metric – An instance of Metric class that specifies the Metric distance function to use when building the index.

4) Create a similarity query.

Both range query and kNN query are defined in MoBIoS. They can be easily instantiated by providing the query object and corresponding value of the radius or k. In Figure 1 (b), line 3 creates a range query looking for all points within Euclidean distance 0.2 to the query object, while line 4 creates a kNN query looking for the 8 closest points. Line 5 creates an RkNN query returning the 8 closest points within distance 0.2. Line 6 defines an AkNN query. As an approximate query, it aims at the 8 closest points within distance 0.2, but only the 2 closest points are guaranteed to be returned.

5) Search the index

To search an index built before, just invoke the search() method of it with the proper query. This method returns a Cursor, which is a subclass of JAVA Iterator. In Figure 1

(b) lines 7 and 8 show how to search the index and retrieve the search results.

The above are general steps for any data type. However, because of the inherent differences between data types, it is recommended that the user tunes the index on his data. The initial construction of the index is the most complex step and is critical to the query performance. There are many factors affecting the construction, and it is an open research topic to determine the best combination of the factors. To allow the user to tune the index for particular data, we have made a constructor of VPIndex with a few more parameters for adjustment:

```
public VPIndex(List data,Metric metric, int numPivot,
PivotSelectionMethod psm, int singlePivotFanout,
PartitionMethod pm, int maxLeafSize)
```

Parameters (only those not discussed before)**:**

numPivot - the number of pivots per node (default 2).

psm - the pivot selection method to use when building the index. There is one pivot selection methods pre-defined: *PivotSelectionMethods.FFT*, which applies the farthest-first-traversal algorithm [7, 8]. The user can define his own pivot selection method as long as it implements the *PivotSelectionMethod* interface.

singlePivotFanout - the number of partitions generated based on each pivot (default 3).

pm - the data partition method to use when building the index. Two partition methods are pre-defined, i.e. *BALANCED* and *ClusteringPartition*, implemented as *enum*s in *PartitionMethods*. User can define his own data partition method as long as it implements the *PartitionMethod* interface.

maxLeafSize - the maximum number of data points in a leaf index node, normally 100, depending on the size of the user defined data type.

For example, the following:

Index *myIndex* = new VPIndex*(data*, *myMetric*, 3, *PivotSelectionMethods.FFT, 3, PartitionMethods.BALANCED, 100*);

creates an instance of VPIndex with FFT as the pivot selection method and BALANCED as the data partition method using 3 pivots, 3 partitions each pivot and at most 100 data objects each leaf node.

3 Property and performance of similarity queries using MoBIoS index

To help the user decide which type of query fits his application best, we discuss the properties and performance of similarity queries in this section. Several applications of the MoBIoS index will be introduced to explain the context of which the performance of the queries will be illustrated.

3.1 Scalability of similarity query

In the study of scalability, a common methodology is to repeat the experiments on a suite of related datasets of different sizes. It is assumed that those datasets have identical data distribution. The performance measure, e.g. running time, is compared with the dataset size. If the performance measure increases at most sub-linearly as the dataset size increases, it is concluded that the system/algorithm scales well.

Typically the distance calculation of complex data type in applications is very costly. Therefore, in our study, the performance is measured by implementation-independent measures, i.e. the number of distance calculations and the number of I/O operations. Since both numbers lead to similar conclusions, we focus on the number of distance calculations.

The most important property of range query is that the number of query results increases linearly as the database size increases, owing to the identical data distribution of the databases. The only exception is when the query radius is 0 and there are no duplications in the database. Since the distances between the query object and each query result need to be calculated, the number of distance calculations increases at least linearly as the database size increases. In other words, it is impossible that a range query scales well using the common meaning.

To evaluate the scalability of range query, we consider the average number of distance calculations per query result. We expect this value to increase sub-linearly as database size increases.

However, since the running time is dominated by the number of distance calculations, it will not scale well for range queries. If an application needs strictly the range query, there is no way to make its running time scale well. Considering the speed difference between sequential disk access and random disk access, if on average more than 15% of the database is returned as query results, we recommend sequential scan of the database instead of using the index.

KNN query has the opposite property. That is, the number of query results is fixed. Further, duplicates are common in large databases. We witness scalable performance using radius-limited kNN query.

However, the query results of a kNN query from a small database are closer to the query object than those from a large database. In other words, the "covering radius" of the query results of a kNN query from a small database is larger than that from a large database. This issue is not acceptable for some applications.

For many practical applications, similarity query is just one step, and the distance measure cannot exactly reflect the need of the applications. That is, not all the results of similarity query are desirable, and instead, some data not returned by similarity query are desirable. This observation justifies the approximate similarity queries, i.e. approximation is allowed in answering the similarity queries. Approximate kNN queries are implemented in MoBIoS index [7]. Empirical results show that approximate queries give good performance while maintaining acceptable accuracy.

3.2 Empirical results from MoBIoS applications

In this section, we demonstrate the properties of similarity queries and study their scalability on empirical results with MoBIoS applications. Due to the space limit, only the number of distance calculations is considered. The running time and the trade-off between performance and accuracy are discussed in detail in corresponding publications.

Three applications are considered, i.e. homologous retrieval of peptide sequences [20], protein identification based on protein tandem mass spectra [14], and finding conserved primer pairs between rice and Arabidopsis genomes [19]

Peptide homologous retrieval follows a general framework first proposed and analyzed by Myers [13].

The database sequences are divided into overlapping 6-mers, on which the index is built. The distance between 6-mers is weighted Hamming distance parameterized by mPAM [20]. The query is also divided into overlapping 6-mers. Each query 6-mer is used as an approximate kNN query object to search the index. Finally, all the query results are chained together to generate the solutions to the full query [20].

The proteomics application of MoBIoS indexes protein tandem mass spectra. Existing mass spectrometry based protein identification tools first run a sequential scan of the database to identify candidates that are "similar" to a query spectrum. These candidate spectra are then input to a complex scoring/ranking scheme. We mapped tandem mass spectra to a high dimensional vector space model, and derived a semi-metric distance based on cosine distance [14]. Using a modified semi-metric search algorithm on the MoBIoS index, we mapped the spectra search problem to range and approximate kNN queries. The index acts as a fast scalable coarse filter, reducing both the number of distance calculations and the number of returned candidates when compared to linear scans [14].

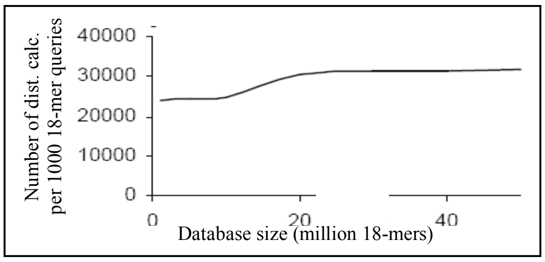


Figure 2 [19]. Finding conserved primer pairs: #dist. calc. vs. db size, range query scales well with radius 0

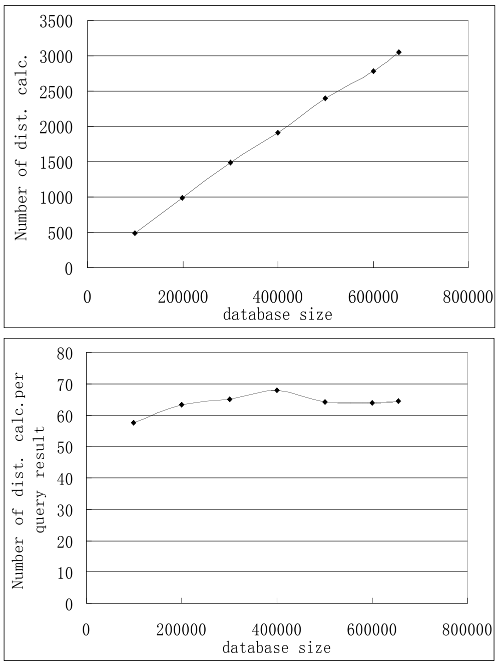


Figure 3. Protein identification: range query with radius 1.48. Up: #dist. calc. vs. db size, does not scale; Bottom: #dist. calc. per query result vs. db size, scales well.

Finding conserved primer pairs aims at identifying evolutionary reticulation events in flowering plants. A large number of paired, conserved DNA oligomers that may be used as primers to amplify orthologous DNA regions using the polymerase-chain reaction (PCR) are identified [19]. This first step is to develop an initial candidate set by comparing the Arabidopsis and Rice genomes, which is implemented by indexed nest-loop join using MoBIoS index [19]. The two genomes are divided into overlapping 18-mers. This similarity query is range query with radius 0, and the distance function is edit distance[21].

We first show the scalability of range query with radius 0. In finding primer pairs, the similarity query is range query with radius 0. Figure 2 shows the number of distance calculations scales well as database size increases [19].

Then, we show the scalability of range query with non-zero radius for protein identification in Figure 3. We can see that, with a large radius, 1.48, the number of distance calculations increases almost linearly as database size increases. However, the average number of distance calculations per query result does scale well.

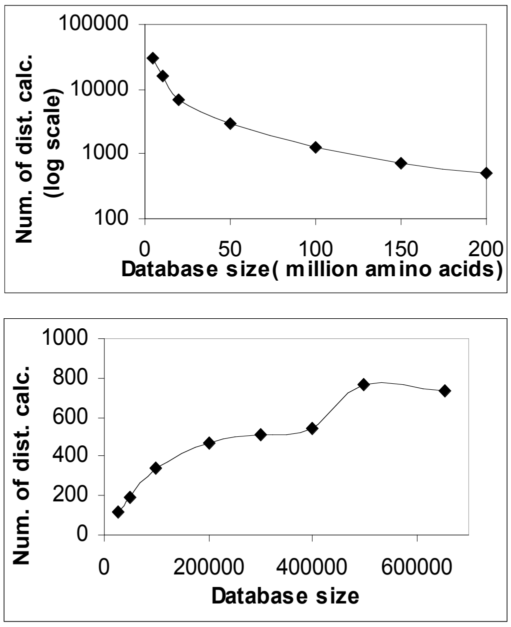


Figure 4 [7]. Scalability of approximate kNN query. Left: Peptide homologous retrieval, k = 300, scales "too well" because of duplicate. Right: protein identification, k= 100, scales well.

Next, we show the scalability of approximate kNN query. Figure 4 [7] shows the scalability of approximate kNN query of peptide homologous retrieval (left) and protein identification (right). Obviously, approximate kNN query scales well in both cases. Note that in peptide homologous retrieval, the number of distance calculations decreases as the database size increases. This is due to the large amount of duplicates in the data.

4 Conclusions and future work

In this paper, we demonstrate the usage of the MoBIoS index and the properties of similarity queries.

MoBIoS is a next generation DBMS for bioinformatics. Similarity query plays critical roles in many bioinformatics applications. The MoBIoS index, evolved from the multiple vantage point tree, provides a uniform programming model for similarity queries of any biological data, as long as a metric distance function can be defined. We showed that there are five essential steps to apply the MoBIoS index to any metric space biological data type.

We further investigate the properties and performance of similarity queries through real applications of MoBIoS. It is impossible for range query with non-zero radius to scale well because the number of range query results increases linearly, in which cases the average number of distance calculations per query result does scale well. Although kNN query does scale well, as the database size increases, the query results become closer and closer to the query object.

In summation, the scalability of similarity query is affected by the number of query results.

Approximate kNN query provides a trade-off between speed and accuracy. Empirical results show that approximate query scales well for real applications.

Reference

[1] Altschul, S.F., W. Gish, W. Miller, E.W. Myers, and D. Lipman, *Basic local alignment search tool.* Journal of Molecular Biology, 1990. **215**(3): p. 403--410.

[2] Bellman, R., *Adaptive Control Processes: A Guided Tour*. 1961: Princeton University Press.

[3] Bozkaya, T. and M. Ozsoyoglu, *Indexing large metric spaces for similarity search queries.* ACM Trans. Database Syst., 1999. **24**(3): p. 361-404.

[4] Chavez, E., G. Navarro, R. Baeza-Yates, and J. Marroqu, *Searching in metric spaces.* ACM Computing Surveys (CSUR), 2001. **33**(3): p. 273-321.

[5] Hjaltason, G.R. and H. Samet, *Index-driven similarity search in metric spaces.* ACM Transactions on Database Systems (TODS), 2003. **28**(4): p. 517-580.

[6] Mao, R., Q. Iqbal, W. Liu, and D.P. Miranker. *Case Study: Distance-Based Image Retrieval in the MoBIoS DBMS*. in *The 5th International Conference on Computer and Information Technology (CIT 2005)*. 2005. Shanghai, China.

[7] Mao, R., W. Xu, S. Ramakrishnan, G. Nuckolls, and D.P. Miranker. *On Optimizing Distance-Based Similarity Search for Biological Databases*. in *the 2005 IEEE Computational Systems Bioinformatics Conference (CSB 2005)*. 2005. Stanford University, California, USA.

[8] Mao, R., W. Xu, N. Singh, and D.P. Miranker. *An Assessment of a Metric Space Database Index to Support Sequence Homology*. in *3rd IEEE International Symposium on BioInformatics and BioEngineering (BIBE 2003)*. 2003. Bethesda, Maryland, USA: IEEE Computer Society.

[9] Mckoi, *Mckoi website: www.mckoi.com*.

[10] Miranker, D.P., W.J. Briggs, R. Mao, S. Ni, and W. Xu, *Biosequence Use Cases in MoBIoS SQL.* IEEE Bulletin of the Technical Committee on Data Engineering, 2004. **27**(3): p. 3-11.

[11] Miranker, D.P., W. Xu, and R. Mao. *MoBIoS: a Metric-Space DBMS to Support Biological Discovery*. in *15th International Conference on Scientific and Statistical Database Management (SSDBM 2003)*. 2003. Cambridge, Massachusetts, USA: IEEE Computer Society.

[12] MoBIoS, *MoBIoS website, http://www.cs.utexas.edu/~mobios*.

[13] Myers, E.W., *A Sublinear Algorithm for Approximate Keyword Searching.* Algorithmica, 1994. **12**(4/5): p. 345-374.

[14] Ramakrishnan, S.R., R. Mao, A.A. Nakorchevskiy, J.T. Prince, W.S. Willard, W. Xu, E.M. Marcotte, and D.P. Miranker, *A fast coarse filtering method for peptide identification by mass spectrometry.* Bioinformatics, 2006. **22**(12): p. 1524-31.

[15] Roman, S., *Advanced Linear Algebra*. Graduate Texts in Mathematics. Vol. 135. 1992: Springer-Verlag.

[16] Sahinalp, S.C., M. Tasan, J. Macker, and M.Z. Ozsoyoglu. *Distance Based Indexing for String Proximity Search*. in *19th International Conference on Data Engineering (ICDE 2003)*. 2003. Bangalore, India: IEEE Computer Society.

[17] Shaft, U. and R. Ramakrishnan. *When Is Nearest Neighbors Indexable?* in *Tenth International Conference on Database Theory (ICDT 2005)*. 2005. Edinburgh, UK: Springer.

[18] Weber, R., H.J. Schek, and S. Blott. *A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces*. in *International Conference on Very Large Data Bases*. 1998. New York City, New York, USA.

[19] Xu, W., W.J. Briggs, J. Padolina, W. Liu, C.R. Linder, and D.P. Miranker. *Using MoBIoS' Scalable Genome Joins to Find Conserved Primer Pair Candidates Between Two Genomes*. in *12th International Conference on Intelligent system for Molecular Biology (ISMB 2004)*. 2004. Galsgow, UK: Oxford University Press.

[20] Xu, W., R. Mao, S. Wang, and D.P. Miranker. *On integrating peptide sequence analysis and relational distance-based indexing*. in *IEEE 6th Symposium on Bioinformatics and Bioengineering (BIBE06)*. 2006. Arlington, VA, USA.

[21] Xu, W. and D.P. Miranker, *A Metric Model of Amino Acid Substitution.* Bioinformatics, 2004. **20**(8): p. 1214-1221.

[22] Yates, J., III, J. Eng, A. McCormack, and D. Schieltz, *Method to correlate tandem mass spectral data of modified peptides to amino acid sequences in the protein database.* Anal. Chem., 1995. **67**(8): p. 1426-36.

[23] Yianilos, P.N. *Excluded middle vantage point forests for nearest neighbor search*. in *Technical report, NEC Research Institute. Presented at the Sixth DIMACS Implementation Challenge: Near Neighbor Searches workshop*. 1999.