Universal Indexing of Arbitrary Similarity Models

Tomáš Bartoš
Supervised by Tomáš Skopal
Charles University in Prague, Faculty of Mathematics and Physics
SIRET Research Group
Malostranské nám. 25, 118 00 Prague 1, Czech Republic
bartos@ksi.mff.cuni.cz

ABSTRACT

The increasing amount of available unstructured content together with the growing number of large non-relational databases put more emphasis on the *content-based retrieval* and precisely on the area of similarity searching. Although there exist several indexing methods for efficient querying, not all of them are best-suited for arbitrary similarity models. Having a metric space, we can easily apply metric access methods but for nonmetric models which typically better describe similarities between generally unstructured objects the situation is a little bit more complicated.

To address this challenge, we introduce **SIMDEX**, the universal framework that is capable of finding alternative indexing methods that will serve for efficient yet effective similarity searching for any similarity model. Using trivial or more advanced methods for the incremental exploration of possible indexing techniques, we are able to find alternative methods to the widely used metric space model paradigm. Through experimental evaluations, we validate our approach and show how it outperforms the known indexing methods.

1. INTRODUCTION

Looking up the right information in large databases within an acceptable time frame is crucial in almost every area. For structured data such as relational databases and simple unstructured content types such as text documents, the efficient querying methods based on various indexing methods are known for decades. On the other hand, the currently popular multimedia objects, social network data, medical, biometric, or scientific databases, are more difficult to search or explore due to higher complexity of stored objects. Therefore, we use *content-based retrieval* [6, 16] which for querying purposes converts the objects from their native formats to more appropriate forms, i.e. *object descriptors*.

If we further apply appropriate similarity model, we can search for most similar objects to the given query object based on the relevancy between any pair of object descriptors which will describe the area of *similarity search* [23].

Proceedings of the VLDB Endowment, Vol. 6, No. 12

The choice of the best similarity model depends heavily on the type of the dataset and the application we deal with. The widely known *metric space model* [23] has been considered as the best choice for many applications because of its simplicity. The fixed properties of *identity*, *positivity*, *symmetry*, and especially the *triangle inequality* create a basis for *metric access methods* [5, 23, 17] (also known as metric indexes) used for indexing databases.

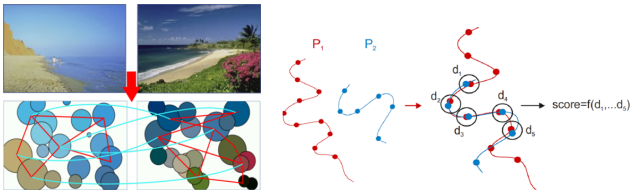
However, for various domains the metric indexing is not convenient because it does not necessary reflect the complexity of objects. So, there emerge similarity models that are more robust, better fit to a particular problem, or more precisely describe objects' relationships. They typically involve a similarity function that violates one or more metric postulates, which makes it difficult or even impossible to apply metric indexing principles. In Figure 1, we depict samples of such *nonmetric* similarity models [22] that use local features for images [14] and alignments of protein structures [9].

Although database researchers try to solve the indexing issues by employing new and usually more complex methods, they do not investigate the applicability of their techniques to specific domains. The much larger *domain expert* communities of various kinds associate people who use specialized similarity search applications and who expect the simplicity of the models, so they can apply them immediately. They focus mainly on the enhancements and customizations of the similarity model to better fit to their problem and are reluctant to use up-to-date more efficient database indexing methods, as they are either complex or difficult to employ.

Therefore the final solution for most applications usually degrades to simple but slow sequential scanning instead of various more sophisticated concepts. Though this is acceptable only to small sized databases, as for larger datasets the efficiency will eventually become a critical factor.

Our research aims to connect these two distinct worlds by enabling the domain experts to use advanced indexing techniques in a simple way. Driven by the efficiency, we propose the concept of SIMDEX – a universal framework that is capable of finding alternatives to metric indexing for any given database using just the pair-wise similarities (or distances) between objects. We focus on the *efficiency* (speeding up the queries), *effectiveness* (precision of the query results), and *simplicity* (the final results are easy-to-use).

Our contribution here is twofold. The domain experts receive a relatively simple tool for discovering alternative indexing methods for speeding up their similarity queries while database researchers will finally be able to apply the advanced concepts also outside of their expertise area.



(a) Image similarity (b) Protein similarity

Figure 1: Sample similarity models

In the following text, we describe the previous attempts to provide alternative ways of database indexing (Section 3), precisely describe how SIMDEX Framework differs from them (Section 4), depict two variants of framework's exploration phase (Sections 4.2 and 4.4), and validate their performance through experimental evaluations (Sections 4.3 and 4.5).

2. SIMILARITY SEARCHING

The principles of *similarity search* [23] are based on the extraction of important objects features (*object descriptors*) combined with the similarity measure used for querying. The most popular queries are the *range query* that for a query object q returns all database objects o_i that are similar to the query to some extent (defined by the radius r), and the kNN queries that return k most similar objects.

To properly rank the results, we need a similarity model that specifies the relevancy between any pair of objects. In this context, distances and similarities are interchangeable, as the increasing similarity (s) gives decreasing distance (δ) between any two objects from the database.

For many applications, the widely known *metric space model* [23] has been considered as the best choice for indexing, mainly because of its simplicity and fixed properties:

$$\begin{aligned} \delta(x,y) = 0 &\quad \Leftrightarrow x = y &\quad &\text{identity} \\ \delta(x,y) > 0 &\quad \Leftrightarrow x \neq y &\quad &\text{positivity} \\ \delta(x,y) = \delta(y,x) &&\quad &\text{symmetry} \\ \delta(x,y) + \delta(y,z) \geq \delta(x,z) &&\quad &\text{triangle inequality} \end{aligned}$$

The combination of these properties enables to index any database using *metric access methods*, also known as metric indexes [5, 23, 17]. Here, we put the strong focus on the query performance efficiency measured by the number of distance computations (DCs) used for ranking the database.

To eliminate as many DCs as possible, we leverage cheaper measures for distance value estimations – the *lowerbounds*. For any valid lowerbound expression LB it holds

$$LB(\delta(q,o)) \leq \delta(q,o) \tag{1}$$

where q is the query object, o is a database object, $\delta(q,o)$ is the distance between q and o, and LB is its estimation. The tighter/better the lowerbound is, the more objects it filters out without computing the generally expensive distance $\delta(q,o)$ and thus saves the evaluation time. To compute the lowerbound, we often use reference objects p_i called pivots which create the basis for various indexes, e.g. *LAESA* [15].

The well-known triangle lowerbound $LB_\triangle$ applies to the metric spaces and it uses a single pivot p:

$$LB_\triangle(\delta(q,o)) = |\delta(q,p) - \delta(p,o)| \leq \delta(q,o) \tag{2}$$

It is a cheap and effective lowerbound method however for nonmetric distances it usually leads to false dismissals [20].

3. RELATED WORK

To overcome the problems with indexing databases with nonmetric similarity models, we can either modify the model (by customizing the similarity measure), or completely change the indexing schema. These two main approaches have been recently studied in more details [19, 20, 10, 1, 14] and provide the foundation for indexing nonmetric models.

One of the very first methods dealing with the nonmetric models was TriGen algorithm [19, 20] which tries to convert nonmetric similarity spaces to semi-metric or metric ones. Based on a given database sample, the algorithm tunes the modified distance $g(\delta)$ with respect to the rate of non-triangular triplets (T-error). After a fixed number of iterations, we get the T-modifier for which the intrinsic dimensionality ρ [5] is minimized.

The modified distance $g(\delta)$ determined by TriGen can be immediately employed by the pivot table [23] for exact but slower (T-error is zero; ρ gets bigger) or approximate but fast (T-error is positive, ρ gets smaller) similarity search. As it was later discovered, the application of TriGen might lead to either large retrieval error or low indexability [22].

A similar concept based on the Lambda Tuning Algorithm has been studied for the nonmetric databases in which the triangle inequality does not hold [1]. Authors focused on tuning nine fuzzy T-norm operators with the connection to the pivot table indexing. As they state, these alternative methods provide increased efficiency but together with higher error rate. The main reasons are (1) the discrepancy between the λ parameter tuned on the sample database for the given error rate and the notable error that λ produced on similarity queries within the whole database, and also (2) required bidirectional distance-to-similarity conversions.

Completely different approach of dealing with nonmetric similarity models was introduced with the concept of Ptolemaic indexing [10, 14]. Instead of tuning the distance or data to comply with metric space properties, authors replace the problematic triangle lowerbound (Equation 2). To construct lowerbounds, they use *Ptolemy's inequality* which for any database objects x, y, u, and v defines the following relation

$$\delta(x,v) \cdot \delta(y,u) \leq \delta(x,y) \cdot \delta(u,v) + \delta(x,u) \cdot \delta(y,v) \tag{3}$$

If the distance function δ holds the properties of *identity*, *positivity*, *symmetry*, and satisfies *Ptolemy's inequality*, we can use *ptolemaic lowerbound* ($\mathrm{LB}_{\mathtt{ptol}}$). To do so, we first define the candidate bound δ_C using two pivots p and s:

$$\delta_\mathrm{C}(q,o,p,s) = \frac{|\delta(q,p) \cdot \delta(o,s) - \delta(q,s) \cdot \delta(o,p)|}{\delta(p,s)} \tag{4}$$

For simplicity, we let $\delta_\mathrm{C}(q,o,p,s) = 0$ if $\delta(p,s) = 0$. Considering a set of pivots $\mathbb{P}$, we maximize the candidate bound δ_C over all pairs of distinct pivots which results in:

$$\mathrm{LB}_{\mathtt{ptol}}(\delta(q,o)) = \max_{p,s \in \mathbb{P}} \delta_\mathrm{C}(q,o,p,s) \leq \delta(q,o) \tag{5}$$

Although this lowerbound is valid for *signature quadratic form distance* [14] applied to the effective matching of image signatures [4], it is in general difficult to determine which preconditions must be met, so the ptolemaic technique will perform better than any other approach such as the triangle lowerbound.

Algorithm 1 Iterative Exploration SIMDEX (S, δ)

Require: database sample S, distance function δ
1: $M_{\delta,S} \leftarrow$ new distance matrix (δ, S)
2: *candidates* $\leftarrow$ GenerateExpressions()
3: **for all** E_i in *candidates* **do**
4: **if** *validate*(E_i) equals **false then**
5: *candidates*.Remove(E_i) {validity check fails}
6: **end if**
7: **if** $ProbabilityTest(E_i, M_{\delta,S}) < 100\%$ **then**
8: *candidates*.Remove(E_i) {probability test fails}
9: **end if**
10: **end for**
11: **return** *candidates*

4. SIMDEX FRAMEWORK

The idea of ptolemaic indexing shows that finding new axioms suitable for indexing could be a solution to speeding up (exact) similarity search in other way than mapping the problem to the metric space model. To automate the whole process of axiom exploration, we introduce the concept of SIMDEX – a universal framework for finding alternative indexing methods for arbitrary similarity models [21, 3, 2].

4.1 Framework Overview

The input for SIMDEX is a distance matrix containing pair-wise distances between objects from a *database sample* (S) computed with a *black-box distance function* (δ). The resulting output is a set of lowerbounding expressions (we call them *axioms*) valid in the given similarity space that are used for indexing and effective similarity search. Moreover, the axioms are obtained in their lowerbounding forms, so they can be immediately used for filtering purposes in the same way as ptolemaic indexing was implemented [14].

Our approach does not use a single canonized form and a tuning parameter, but it generates candidate lowerbound expressions that are validated against the distance matrix. We evaluate two distinct approaches, namely the *iterative search* [21] and the *inequality symbolic regression* [21, 3, 2]. Both of them apply a similar concept and differ only in the way of creating candidate lowerbounds for testing

Note that, we do not study whether the input similarity model is valid or whether the query results make any sense.

4.2 Iterative Method

This technique allows us to iteratively construct and test the expressions, so we algorithmically explore axiom spaces specified in a syntactic way [21]. We define a grammar for generating candidate expressions; it includes a reasonable number of terminals such as *object descriptor variables* $(q, o, p_1, p_2, \ldots)$ and *constants* (c_i) together with the nonterminal *functions* $(+, *, -, /,$ with additional unary/binary math functions `min`, `max`, `sin`, `cos`, `pow`,$\ldots)$. All these items are mutually combined to build more complex, yet mathematically correct expressions.

Nevertheless, all candidates are in the standardized form:

$$E_i \equiv \Big[\delta(q, o) \geq LB\Big] \tag{6}$$

where $\delta(q, o)$ is the real distance between the given query object q and a database object o, and the right-hand side LB is a nonterminal which is additionally expanded.

After we construct the set of candidate expressions, each expression E_i is validated – it cannot be computationally too expensive, the expanded LB form cannot contain the distance $\delta(q, o)$, while it should include combinations of $\delta(q, p_i)$ and $\delta(p_i, o)$, where p_i stands for a reference object (a pivot).

Only such candidate expressions that meet all the above requirements are further considered for testing. Algorithm 1 depicts the simplified version of the iterative method.

4.3 Iterative Method Evaluation

To validate the iterative exploration of lowerbound expressions, we built SIMDEX prototype and applied it to the real-world datasets. We focus on the nonmetric similarity models in which metric postulates used for indexing and querying produce notable errors.

We trace several parameters in order to find out the vitality of SIMDEX Framework. First, we study the *Average Precision* over all query evaluations using a specific lowerbounding method which gives us the quality of query results. More precisely it returns the number of items in the query result that are identical to the items in the query result from the sequential (SEQ) scan divided by the total number of items in the result. One might see this as the application of Jaccard distance [20] to the query result sets.

Next, we focus on the *Average DC ratio* that represents the query efficiency. This gives us the comparison between the average number of distance computations given by a query with a specific lowerbound and DCs produced by the SEQ scan. We select the average number of DCs as it is a relatively good measurement for query efficiency not dependent on the hardware used. We suppose that the SEQ scan always returns appropriate results (the ground truth).

In our experiments, we explore the axiom space given by the distance matrix (25×25 objects), and we test more than 50,000 candidate expressions. Next, we evaluate the indexing test based on 100 random kNN queries ($k = 10$) using the best candidate lowerbound expressions from the first step for pivot table (with 10 pivots) filtering as was recently proposed [14].

We evaluated various similarity models even though only a few gave us interesting results that we will describe in more details. For CoPhIR[1] dataset with nonmetric $L_{0.5}$ distance [20], the best discovered lowerbound expression is

$$\delta(q, o) \geq LB_{\triangle}^{1.85}(\delta(q, o)) = |\delta(q, p) - \delta(p, o)|^{1.85} \tag{7}$$

Although this candidate does not dominate in the number of distance computations (see Figure 2) it clearly produces no errors compared to other approaches (Figure 3) together with 55% of DCs given by the SEQ scan.

We also discuss color histograms from *Corel Image Features*[2] dataset using nonmetric Jeffrey Divergence [22] which gave us expressions such as

$$\#18690 \qquad \delta(q, o) \geq (\delta(q, p) - \delta(o, p))^2 = LB_{\triangle}^2(\delta(q, o)) \tag{8}$$

$$\#18906 \quad \delta(q, o) \geq (\delta(q, p_1) - \delta(o, p_1)) \cdot (\delta(q, p_2) - \delta(o, p_2)) \tag{9}$$

The squared triangle inequality (Equation 8) is only slightly more precise than $LB_{\triangle}$ (see Figure 5). However, we achieved a relative success with #18906 (Equation 9) – we got 99.8% precision with only 5% of DCs compared to SEQ scan. Even though $LB_{\triangle}$ still dominates in the number of DCs (Figure 4), it produces notable error rates.

[1] http://cophir.isti.cnr.it/

[2] http://kdd.ics.uci.edu/databases/CorelFeatures/CorelFeatures

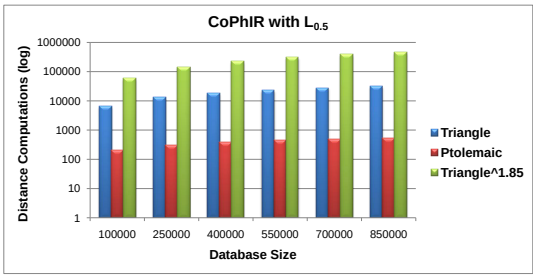


Figure 2: CoPhIR - Distance computations (log scale)

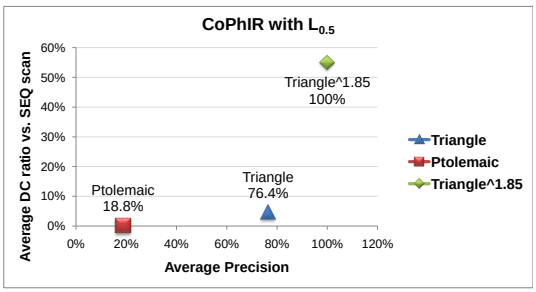


Figure 3: CoPhIR - Average DC ratio vs. average precision

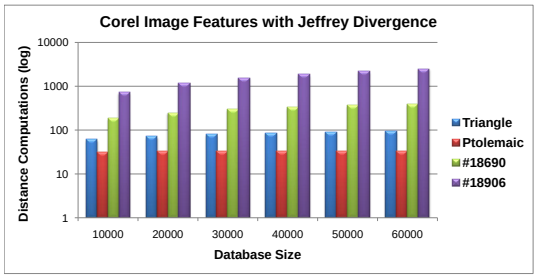


Figure 4: Corel - Distance computations (log scale)

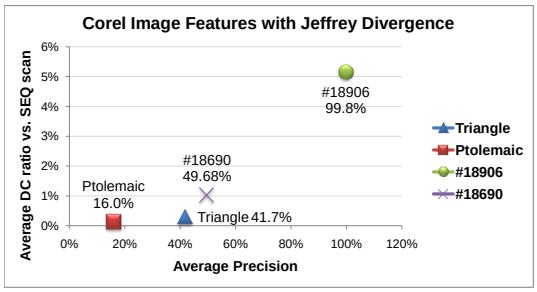


Figure 5: Corel - Average DC ratio vs. average precision

These results verify the viability of the straightforward approach. They also reveal challenges we need to address to improve the performance such as (1) the discovery of a complex axiom takes some time when constructing all expressions iteratively; (2) despite using various enhancements [21], we struggle with testing only unique expressions as there exist infinite forms of a single math expression; and (3) there is always a trade-off between testing larger samples (for more precise results) or testing more candidates.

As the iterative exploration did not give us the expected results, we investigated other possibilities to further enhance the performance of SIMDEX Framework. For this purpose, we explore the interesting area of genetic algorithms [12, 13].

4.4 Using Inequality Symbolic Regression

Even if we limit the grammar-based generation of the candidate expressions, we are still handling an exponential problem. We address this issue and our objective is to guide the exploration to the most promising candidates first. To achieve this, we develop the exploration phase based on the genetic programming (GP) principles [12] which were introduced to solve computationally intensive problems for which simpler algorithms or random optimizations do not work.

After building the initial population of random solutions for the given problem (lowerbounds in our case),we apply genetic operations [13] to build consequent generations which will lead to the perfect solution. These operations include:

- *reproduction* – the expression remain unchanged and is just copied to the next generation
- *mutation* – the expression is mutated at the random point and a new expression is derived
- *crossover* – two expressions are recombined at random points (two new offspring expressions appear)
- *alter architecture* – the new expression is created by changing a single unary or binary operation

After a fixed number of iterations or when a stop condition is satisfied, the algorithm returns the best-so-far individual which corresponds to the final resulting expression.

Inspired by the success of the software tool Eureqa [18], we focus on the *symbolic regression* [12, 11] which searches for the mathematical expression that fits to the given sample of real-valued numeric data. It looks for the perfect model that provides a linear combination of independent variables with corresponding numeric coefficients and that minimizes the measured error rate.

In our scenario, we use *inequality symbolic regression* [2] which searches for inequalities $LB \leq \delta(q,o)$ with a specific error measure that minimizes the differences between the lowerbound and the real (computed) distance:

$$\texttt{AvgDifference}(LB) = \frac{1}{|T|}\sum_{t\in T}(\delta(q,o) - LB(t)) \qquad (10)$$

where t is the tuple of values for independent variables, $\delta(q,o)$ is the computed distance between the query (q) and object (o) variables and $LB(t)$ corresponds to the lowerbound computed with the values from the tuple. The lower the error measure, the better the candidate lowerbound estimates the real distance.

We also check for the *lowerbound correctness* which means that all results comply with the perception $LB \leq \delta(q,o)$. We also skip candidates for which $LB \leq 0$ (zero filtering power), and dismiss all other inappropriate candidates.

We apply a specific fitness function defined as:

$$fitness(LB) = \frac{\texttt{SuccessRatio}(LB)}{\texttt{AvgDifference}(LB)} \qquad (11)$$

where $\texttt{SuccessRatio}(LB)$ is the percentage of successful tuple evaluations, and $\texttt{AvgDifference}(\text{LB})$ corresponds to the average difference between the real distance and the tested lowerbound (see Equation 10).

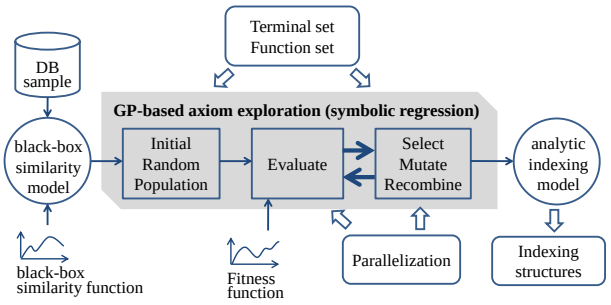


Figure 6: GP-based SIMDEX Framework

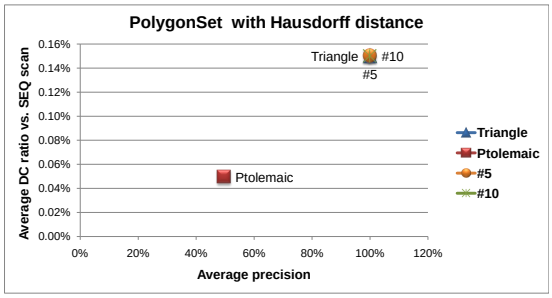


Figure 7: PolygonSet – Avg DC ratio vs. Avg precision

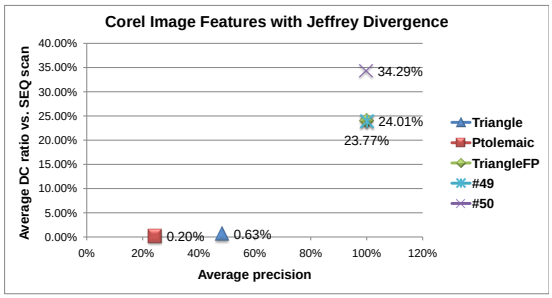


Figure 8: Corel – Avg DC ratio vs. Avg precision

Finally, we provide the high-level overview of the GP-based exploration. In Figure 6, we depict all components and how they interact with each other. For more details, we refer readers to the recent research [2, 3].

4.5 GP-based Experimental Evaluation

For experiments, we use the terminal set which includes distances between random pairs of six independent database objects (query q, object o, and pivots $p_1 \dots p_4$) and random numeric constants from the interval [-5, 5]. The function set consists of unary functions (`abs`, `sqr`) and the binary functions (`+`, `-`, `*`, `/`, `min`, `max`).

Regarding the specific settings for GP-based exploration, we use the population size of 800 individuals that evolve within 1,000 generations using the standard tournament selection with re-selection enabled. The genetic operations occur with probabilities: 5% mutation, 10% reproduction, 10% alter architecture, and 75% recombination.

As the evaluation function, we apply random sampling of n-tuples with a fixed number of 8,000 tuples for each n (the number of independent variables in the lowerbound). This enables us to use larger distance matrix consisting of pairwise distances between 1,000 randomly selected objects.

In all cases, we repeat the individual run of the genetic algorithm and provide results for the best run. In each run, we always save the top 10 individuals with best results to compare with previous/further results.

Although we evaluated various similarity models, we again pick two representational datasets to depict results of GP-based SIMDEX. We select

- *PolygonSet* which is a synthetic dataset of 250,000 polygons in 2D space, each consisting of 5 to 15 vertices measured by Hausdorff distance [23] using L_2 as the ground distance.
- Color histograms from *Corel Image Features* dataset using nonmetric Jeffrey Divergence

For indexing tests, we performed 100 randomly chosen kNN queries with $k = 10$ and averaged the results over three database sizes (100,000; 150,000; and 200,000) for *PolygonSet*, and over two database sizes (10,000 and 20,000) for *Corel Image Features*. We used pivot tables with 10 pivots.

We evaluate the effectiveness of selected lowerbounds by comparison with traditional triangle and ptolemaic lowerbounds and, if applicable, with TriGen algorithm [19, 20].

The *PolygonSet* dataset represents the metric similarity space because we select L_2 distance as the ground distance for the main Hausdorff distance [23]. We observe this fact in Figure 7 which shows how the precision relates to the average DC ratio. The triangle lowerbounding $LB_\triangle$ provides 100% precision together with the dramatic improvement of number of DCs. We admit that the Ptolemaic method was faster but it provided a critically low precision of 49%.

Then, we evaluated two newly discovered lowerbounds; the filtering power of the lowerbound #10

$$\delta(q,o) \geq |min\left(\delta(p_2,p_1)\cdot 3.2683, \delta(o,p_1) - \delta(p_1,q)\right)|$$

is the same as the $LB_\triangle$ while the lowerbound #5

$$\delta(q,o) \geq \left| min\left(2.3192, \frac{-4.9608}{\delta(o,p_1)^2}\right) - (\delta(o,p_2) - \delta(p_2,q))\right|$$

performs slightly better than the previous two techniques. Apparently, the improvements were hardly noticeable.

The second dataset *Corel Image Features* is nonmetric, therefore we apply also *TriangleFP* lowerbound which is the triangle lowerbounding using the modified distances obtained by the TriGen algorithm. For our data, we use 100,000 triplets in 24 iterations to find the best Fractional Power (FP) modifier [19, 20] with the weight $w = 0.802037$.

We present the overall results in Figure 8. Although there is a huge speed-up, the very low precision of the triangle lowerbounding (only 48%) verifies that the similarity model is nonmetric. The ptolemaic method is a bit faster but with even worse quality of 24% correct results.

It is remarkable to show the comparison of these standard methods with the newly discovered lowerbound #50

$$\delta(q,o) \geq min\left(\left|\frac{\delta(o,p_1) - \delta(p_1,q)}{max(4.3194, \delta(p_2,q))}\right|, \delta(q,p_1)\right)$$

It provides 99.8% precision rate using less than 35% of DCs from SEQ scan. We acknowledge that this is not a perfect fit, however it indicates a pretty good result.

More spectacular results gives the lowerbound #49

$$\delta(q,o) \geq min\left(\frac{\delta(p_1,q)\cdot(\delta(o,p_2)-\delta(p_2,q))^2}{2.0993}, \delta(o,p_3)\right)$$

as it is the perfect solution in this case. It delivers 100% precision together with the consumption of only 23.77% of DCs compared to SEQ scanning.

To be fair in the evaluation, we compare our results with TriangleFP lowerbounding. The result of TriGen algorithm clearly works here, as it holds 100% precision while using 24.01% of sequential DCs for the evaluation.

These experiments validates that if properly configured the GP-based SIMDEX is capable of discovering valid alternative indexing methods that are comparable or even better than the existing concepts. This also verifies our approach of searching for new axioms valid in arbitrary similarity spaces.

5. CONCLUSIONS

In this paper, we focus on the query performance and database indexing for content-based retrieval. In this area, the applied similarity models are generally nonmetric which raises an issue, as the classical metric indexes cannot be used and the query evaluations degrade to slow sequential scanning. We address this challenge by introducing SIMDEX Framework which is capable of discovering alternative indexing methods for arbitrary similarity models, not limited to metric ones. We outline the basic idea of iterative exploration of lowerbound candidates, followed by more advanced technique based on the genetic programming. Driven by the efficiency, we demonstrate the validity and applicability of SIMDEX and showed how it, in most cases, outperforms the existing indexing methods.

However, there are still some issues we need to address such as testing only unique expressions or better n-tuples sampling. Besides the currently used multi-core CPUs with multi-threading, we plan to apply also other parallelism techniques to explore greater spaces (e.g., boosting the GP-based concept with the *island-population model* [7, 8]).

6. ACKNOWLEDGMENTS

This research has been supported by Grant Agency of Charles University (GAUK) project 567312.

7. REFERENCES

[1] T. Bartoš, A. Eckhardt, and T. Skopal. Fuzzy Approach to Non-metric Similarity Indexing. In *SISAP 2011*, pages 115–116. ACM, 2011.

[2] T. Bartoš, T. Skopal, and J. Moško. Efficient Indexing of Similarity Models with Inequality Symbolic Regression. In *GECCO 2013*. ACM, 2013.

[3] T. Bartoš, T. Skopal, and J. Moško. Towards Efficient Indexing of Arbitrary Similarity. *SIGMOD Record*, 42(2):5–10, 2013.

[4] C. Beecks, M. S. Uysal, and T. Seidl. Signature quadratic form distance. In *Proc. ACM International Conference on Image and Video Retrieval*, pages 438–445, 2010.

[5] E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín. Searching in metric spaces. *ACM Comp. Surveys*, 33(3):273–321, 2001.

[6] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and Effective Querying by Image Content. *J. Intell. Inf. Syst.*, 3:231–262, July 1994.

[7] F. Fernandez, G. Spezzano, M. Tomassini, and L. Vanneschi. Parallel genetic programming. In *Parallel Metaheuristics*, Parallel and Distributed Computing, chapter 6, pages 127–153. USA, 2005.

[8] C. Gagné, M. Parizeau, and M. Dubreuil. A robust master-slave distribution architecture for evolutionary computations. In *Late Breaking Papers, GECCO 2003*, pages 80–87, July 12-16 2003.

[9] J. Galgonek, D. Hoksza, and T. Skopal. SProt: sphere-based protein structure similarity algorithm. *Proteome Science*, 9:1–12, 2011.

[10] M. L. Hetland. Ptolemaic indexing. `arXiv:0911.4384 [cs.DS]`, 2009.

[11] J.W.Davidson, D.A.Savic, and G.A.Walters. Symbolic and numerical regression: experiments and applications. *Information Sciences*, 150(1-2):95 – 117, 2003.

[12] J. R. Koza. *Genetic programming*. MIT Press, Cambridge, MA, USA, 1992.

[13] J. R. Koza and R. Poli. Genetic programming. In *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, chapter 5, pages 127–164. Springer, 2005.

[14] J. Lokoč, M. Hetland, T. Skopal, and C. Beecks. Ptolemaic indexing of the signature quadratic form distance. In *SISAP 2011*, pages 9–16. ACM, 2011.

[15] G. Navarro. Analyzing metric space indexes: What for? In *IEEE SISAP 2009*, pages 3–10, 2009.

[16] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. H. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The QBIC Project: Querying Images by Content, Using Color, Texture, and Shape. In *Storage and Retrieval for Image and Video Databases (SPIE)'93*, pages 173–187, 1993.

[17] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann Publishers Inc., USA, 2005.

[18] M. Schmidt and H. Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

[19] T. Skopal. On fast non-metric similarity search by metric access methods. In *EDBT'06*, LNCS 3896, pages 718–736. Springer, 2006.

[20] T. Skopal. Unified framework for fast exact and approximate search in dissimilarity spaces. *ACM Transactions on Database Systems*, 32(4):1–46, 2007.

[21] T. Skopal and T. Bartoš. Algorithmic Exploration of Axiom Spaces for Efficient Similarity Search at Large Scale. In *SISAP 2012*, LNCS, 7404, pages 40–53. Springer, 2012.

[22] T. Skopal and B. Bustos. On nonmetric similarity search problems in complex domains. *ACM Comp. Surv.*, 43:1–50, 2011.

[23] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach*. Advances in Database Systems. Springer-Verlag, USA, 2005.