Fairest Neighbors
Tradeoffs Between Metric Queries

Magnus Lie Hetland[0000−0003−4204−2017] and Halvard Hummel[0000−0001−5691−8177]

Norwegian University of Science and Technology, Trondheim, Norway
{mlh,halvard.hummel}@ntnu.no

Abstract. Many methods exist for indexing metric spaces, in order to resolve similarity queries as efficiently as possible: A sample object is provided, and similar objects from the indexed data set are found. In general, the only ways to adjust the query are to replace the query object, or to vary the search radius or nearest neighbor count. We explore a generalization, where the desired result is a tradeoff between multiple query objects. This builds on previous results on complex queries, such as linear combinations. Here, we instead use measures of inequality, such as ordered weighted averages, and use existing index structures to find objects that minimize these. We compare our method empirically to linear scan and a post hoc combination of individual queries, and demonstrate a considerable speedup.

1 Introduction

From the early days, indexing metric spaces has mainly been in service of straightforward similarity search: Given some query object q, find other objects o for which the distance $d(q, o)$ is low—either all points within some search radius, or a certain number of the nearest neighbors. Alternative forms of search have been explored, certainly, such as *reverse* nearest neighbor queries or similarity joins. Some even combine multiple queries, imposing multiple requirements on the results simultaneously. For example, one may impose limits on both on the distance to q and on the number of nearest neighbors returned.

Of particular interest to us, is using multiple query objects q_i, without restricting the indexing methods used. That is, we wish to take any existing metric index, already constructed, and execute a combination query on it. Such a query may be specified directly by the user, or it may be used as a form of interactive refinement. A user first performs a query using a single object. Then she indicates which of the returned objects are most relevant (possibly to varying degrees), and these are then used as a second, combined query. The result should ideally be a tradeoff between the query objects. In particular, we wish to ensure that the query result is a *fair* tradeoff between the query objects, borrowing measures of fairness from the field of multicriteria decision making.

In this short paper, we introduce the idea of *fairest neighbors* (kFN), i.e., items that are close to multiple query objects at once, as measured by some kind of

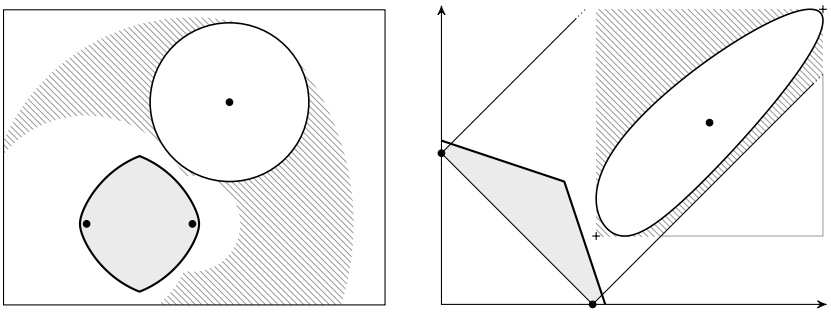


Fig. 1. A complex query with two query objects (gray) and a ball region. The right-hand subfigure shows the query and region in pivot space, where the two axes correspond to distancs from the two query objects.

fairness measure. We formulate such queries in the context of the complex queries of Ciaccia et al. [2], but extend the formalism applying linear ambit overlap [5] to ordered weighted averages (OWA) and weighted OWA, for improved bounds. The resulting queries may be resolved using existing metric index structures without modification. We perform some preliminary experimental feasibility tests, showing that such combined queries outperform linear scan and an alternative strategy involving multiple kNN queries.

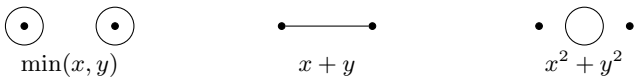
2 Complex Queries as Multicriteria Decisions

In 1998, Ciaccia et al. introduced a formalism for dealing with what they called *complex queries* for metric indexes—queries involving multiple query objects, along with some domain-specific query *language*, specifying which objects were relevant and which weren't [2]. Part of their formalism involved mapping distances to similarity measures, which were then constrained by some query predicate, though the core ideas apply equally well to distances directly. A central insight is that *monotone* predicates may be used not just to detect whether an object is relevant, but also whether certain *regions* might contain relevant objects.

Let $x = [d(q_i, o)]_{i=1}^{m}$ be a vector of distances between query objects q_i and some potentially relevant object o. The relevance is then defined by some predicate $\mathrm{P}(x)$. This predicate is *monotone* if for all $x \leqslant y$, we have that $\mathrm{P}(y)$ implies $\mathrm{P}(x)$. That is, if we start with the distance vector of a relevant object, and we reduce one or more of the distances, the resulting distance vector should *also* be judged as relevant. In this case, using *lower bounds* for the individual distances is safe (i.e., it will not cause false negatives). So, for example, if we know that o is in a ball with center c and radius r, we can safely replace $[d(q_i, o)]_i$ with $[d(q_i, c) - r]_i$ and apply P to determine potential overlap. One can then find the k best objects by maintaining a steadily shrinking search radius encompassing the k

best candidates found so far, just as one would do for kNN. The idea is illustrated in Fig. 1: The vector $x - r$ of lower bounds corresponds to the lower left corner of the square enveloping the region in pivot space (the right-hand subfigure). A monotone query and a ball region may overlap only if this lower left point is inside the query definition in this space. Similarly, we may conclude that the region is *entirely inside* the query (and thus return all its objects without further examination) if the upper right corner $(x + r)$ satisfies the query predicate.

Two of the query types discussed explicitly by Ciaccia et al. are based on fuzzy logic, and one uses a weighted sum. These permit indicating degrees of relevance for the various query objects q_i, but may have many equally good solutions, with vastly different properties. What can be done if we wish to enforce some form of tradeoff? Consider a query predicate of the form $f(x) \leqslant r$. Different monotone functions f may yield very different query regions:



Minimum (corresponding to maximum, or standard fuzzy disjunctions, in the similarity formalism of Ciaccia et al.) produces results that are close to one or the other of the two query points, but not both. A sum gives us points that can lie anywhere between the two (in general within an ellipsoid). A sum of positive powers, however, produces items that are *between* the query points—ideally in the middle (i.e., in their metric midset). This is the kind of query we want.

Using sums of powers to characterize tradeoffs is a common approach in cardinal welfarism, and it is one of a broader class of aggregation functions used in multicriteria decision making [4].* These are all generally monotonically increasing, with the optimum found for some fair tradeoff between their parameters. Applied to individual query distances $d(q_i, o)$, our measure will of course need to be *minimized*, and so must be an *unfairness* measure, rather than a *fairness* measure. In the following, we will focus on *ordered weighted average* (OWA), and its generalization, *weighted* OWA. The OWA of some vector x is based on a weighting of the elements of x, just like a weighted average, except that the weights are applied based on the rank of each element x_i. That is, given a weight vector $w \geqslant 0$, whose weights sum to 1, the OWA of x is $wx^{\uparrow}$. Here, $x^{\uparrow}$ is a sorted version of x. As discussed in Section 3 (in a more general setting), by ensuring that w is also sorted, we get an unfairness measure. Our overlap check with an r-ball, using the complex query formalism, becomes:

$$f(x - r) \leqslant s \iff wx^{\uparrow} - r \leqslant s\,. \tag{1}$$

For some structures, such as VP-trees [8], LC [1,6] and HC [3], we also need to determine whether the query is *entirely inside* a ball region—or, equivalently, whether it intersects with the complement of the ball. Our lower bound on each

* Though Ciaccia et al. do not directly address fairness or tradeoffs, their *standard* and *algebraic fuzzy conjunctions*, correspond to the *maximin* and *Nash welfare* fairness measures, respectively, if applied, in isolation, to similarities [2].

distance between the query and the outside is $r - x_i$, and using monotonicity again, we get the criterion $r - wx^{\uparrow} < s$. If, however, we do not treat the query as a black-box monotone function, we can, as described in the following section, get the stronger criterion $r - wx^{\downarrow} < s$, where $x^{\downarrow}$ is x sorted in descending order. The difference between these two bounds can be *arbitrarily large*, even for just two query objects. The complemented ball is also just a particularly simple example of a linear ambit with negative coefficients [5]; the situation is similar for other such regions.

3 Ordered Weighted Averages and Linear Ambits

It is possible to construct a weighted generalization of OWA, called *weighted* OWA (WOWA), where some individuals (i.e., query objects) get preferential treatment when determining a tradeoff [7]. In this section, we examine the use of WOWA for fair neighbor queries.

Definition 1 ([4,7]). *Let $p = [p_1, \ldots, p_m]$ and $w = [w_1, \ldots, w_m]$ be weighting vectors, such that $p_i, w_i \in [0, 1]$ and $\sum_{i=1}^{m} p_i = \sum_{i=1}^{m} w_i = 1$. Then, the* weighted ordered weighted average (WOWA) *of a vector $x \in \mathbb{R}^m$ with regards to p and w is defined by:*

$$\mathrm{WOWA}(x; p, w) = \sum_{i=1}^{m} \left[\varphi\left(\sum_{k=i}^{m} p_{\sigma(k)} \right) - \varphi\left(\sum_{k=i+1}^{m} p_{\sigma(k)} \right) \right] x_{\sigma(i)} \,, \tag{2}$$

where σ is any permutation of x in increasing order and $\varphi : [0, 1] \to [0, 1]$ is the function defined by linear interpolation between values $\varphi(i/m) = \sum_{k=1}^{i} w_{m-k+1}$ and $\varphi(0) = 0$.

Ordinarily, WOWA uses decreasing weights, and is then a fairness measure. This works well for similarities, but in the following, we wish to work with distances, and thus need an *un*fairness measure. One way of achieving this is to simply use an *increasing* weight vector. This makes intuitive sense, but for similarities $s(u, v) = 1 - d(u, v)$, as used by Ciaccia et al. [2], we can show that the least unfair distance tradeoff is exactly the fairest similarity tradeoff.*

Proposition 1. *Let p, w and w' be WOWA weighting vectors, with $w'_i = w_{m-i+1}$ for all $i \in \{1, \ldots, m\}$. For any $x \in [0, 1]^m$, we have that:*

$$\mathrm{WOWA}(x; p, w') = 1 - \mathrm{WOWA}(1 - x; p, w) \tag{3}$$

Proof. Let φ_w and $\varphi_{w'}$ be the function φ, as defined in Definition 1, for w and w', respectively. Also, let σ and σ' be permutations of, respectively, $1 - x$ and x in increasing order so that $\sigma'(i) = \sigma(m - i + 1)$. We have that:

$$\mathrm{WOWA}(1 - x; p, w) = 1 - \sum_{i=1}^{m} \left[\varphi_w\left(\sum_{k=i}^{m} p_{\sigma(k)} \right) - \varphi_w\left(\sum_{k=i+1}^{m} p_{\sigma(k)} \right) \right] x_{\sigma(i)} \tag{4}$$

* Note that, following Ciaccia et al., we assume $s(u, v) \in [0, 1]$, which requires a bounded metric, with $d(u, v) \in [0, 1]$.

Further, one can easily verify that $\varphi_{w'}(b) - \varphi_{w'}(a) = \varphi_w(1-a) - \varphi_w(1-b)$ for $a, b \in [0,1]$ and that $\sum_{k=i}^{m} p_{\sigma(k)} = 1 - \sum_{k=1}^{i-1} p_{\sigma(k)}$. Thus,

$$\begin{aligned}
\mathrm{WOWA}(1-x; p, w) &= 1 - \sum_{i=1}^{m} \left[\varphi_{w'}\left(\sum_{k=1}^{i} p_{\sigma(k)} \right) - \varphi_{w'}\left(\sum_{k=1}^{i-1} p_{\sigma(k)} \right) \right] x_{\sigma(i)} && (5)\\
&= 1 - \sum_{i=1}^{m} \left[\varphi_{w'}\left(\sum_{k=i}^{m} p_{\sigma'(k)} \right) - \varphi_{w'}\left(\sum_{k=i+1}^{m} p_{\sigma'(k)} \right) \right] x_{\sigma'(i)} && (6)\\
&= 1 - \mathrm{WOWA}(x; p, w') && (7)
\end{aligned}$$

Equation (3) can then easily be obtained from (7). □

For our overlap check, we wish to model a WOWA query as a *linear ambit* $\mathrm{B}[q, s; \mathrm{W}] = \{o : \mathrm{W}x_o \leqslant s\}$, where $x_o = [d(q_i, o)]_i$, as introduced by Hetland [5]. While WOWAs are not linear functions, we can emulate a query with m query objects as a linear ambit with $m!$ facets, one per possible permutation of x. Normally, the intersection check would require considering each facet in turn, which would quickly become computationally unfeasible with an increasing m, and could in theory lead to false positives.* However, when the weights for the WOWA representing our unfairness measure are ordered in increasing order (corresponding to a fairness measure on similarities, per Proposition 1), we can show that membership and overlap checks need only consider *one* of the facets, eliminating both of these problems.

Proposition 2. *Let w and p be weighting vectors, where $w_1 \leqslant w_2 \leqslant \cdots \leqslant w_m$. Let W be a matrix with $m!$ rows, one for each possible permutation, σ, of a m-dimensional vector. For a specific permutation σ, the value in column i of the corresponding row is:*

$$\varphi\left(\sum_{k=j}^{m} p_{\sigma(k)} \right) - \varphi\left(\sum_{k=j+1}^{m} p_{\sigma(k)} \right) , \tag{8}$$

where $\sigma(j) = i$ and φ is the function from Definition 1. For $x \in \mathbb{R}^m_{\geqslant 0}$ and $s \in \mathbb{R}$, let w_σ be the row in W corresponding to permutation σ. If σ orders x in increasing order, $\mathrm{W}x \leqslant s$ iff $w_\sigma x \leqslant s$. If σ orders x in decreasing order, $\mathrm{W}x > s$ iff $w_\sigma x > s$.

Proof. For any permutation σ, we can create a new permutation σ', with $\sigma'(i) = \sigma(i+1)$, $\sigma'(i+1) = \sigma(i)$ for some $i \in \{1, \ldots, m-1\}$ and $\sigma'(j) = \sigma(j)$ for all $j \notin \{i, i+1\}$. Since w is ordered in increasing order, we know that the growth of φ is monotonically decreasing over $[0,1]$. Combined with the fact that $\|w_\sigma\|_1 = \varphi(1) - \varphi(0) = \|w\|_1 = 1$ for all σ, we get that:

$$\begin{cases} w_\sigma x \geqslant w_{\sigma'} x & \text{if } x_{\sigma(i)} < x_{\sigma(i+1)} \\ w_\sigma x \leqslant w_{\sigma'} x & \text{if } x_{\sigma(i)} > x_{\sigma(i+1)} \\ w_\sigma x = w_{\sigma'} x & \text{otherwise} \end{cases} \tag{9}$$

* This is discussed by Hetland in Sect. 3.1 [5].

If a permutation σ does not order x in increasing order, there exists an i such that $x_{\sigma(i)} > x_{\sigma(i+1)}$. Thus, there exists another permutation σ' with $w_{\sigma'}x \geqslant w_\sigma x$. Consequently, one of the permutations, σ, that maximizes $w_\sigma x$ must order x in increasing order. Note that by the third case in (9), if there are multiple permutations that order x in increasing order, the value of $w_\sigma x$ is equivalent for all of them. In a similar manner, any permutation, σ, that orders x in decreasing order minimizes $w_\sigma x$. □

Using the construct in Proposition 2, we can for a WOWA-based unfairness measure, defined by weighting vectors w and p, create a linear ambit $\mathrm{B}[q, s; \mathrm{W}]$. As long as the weighting vector w is in increasing order, i.e., $w = w^\uparrow$, the membership check of this ambit, $\mathrm{W}x \leqslant s$, is equivalent to checking that $\mathrm{WOWA}(x; p, w^\uparrow) \leqslant s$. That is, this ambit is equivalent to a range query with the WOWA-based unfairness measure. And when checking whether this query ambit overlaps with the inverted s-ball round c, we can in principle perform $m!$ individual checks like

$$r - w_\sigma x < s \iff w_\sigma x > r - s\,, \tag{10}$$

one per row σ.* Proposition 2 shows us that we need only consider the single row, corresponding to ordering x in decreasing order. In other words, the overlap check is strengthened from $s > r - \mathrm{WOWA}(x; p, w^\uparrow)$ to $s > r - \mathrm{WOWA}(x; p, w^\downarrow)$.

4 Experiments

To demonstrate the practical feasibility of the method, even without any fine-tuning or high-effort optimization, we have tested it empirically on synthetic and real-world data, using the basic index structure *list of clusters* (LC), as described by Chávez and Navarro [1]. Briefly, the LC partitions the data set into a sequence of ball regions, each defined by a center, a covering radius, and a set of member items. A search progresses by detecting overlap with each ball in turn, potentially scanning its members for relevance. A defining feature of LC is that the points in later buckets fall entirely outside previous balls, so that if the query falls entirely *inside* one of the balls, the search process may be halted.

More specifically, bucket centers were chosen to maximize distance to previous centers (heuristic $p5$ of Chávez and Navarro), with each ball constructed to contain the 20 closest points to the center, as well as any additional points that fall within the resulting radius.† The data sets used were:

- Synthetic: 100 000 uniformly random and clustered vectors from $[0, 1]^{\mathrm{D}}$, $\mathrm{D} = 2, 4, \ldots, 10$. The clustered vectors were constructed by first generating

* This follows from the linear ambit overlap check described in Theorem 3.1.2 of Hetland [5], as well as from the monotonicity result of Ciaccia et al. [2], inserting the lower bound $r - x$ into the ambit membership predicate.

† These choices were made based on the results of Chávez and Navarro [1], which indicated that $p5$ yielded the best results overall, and a bucket size of 20 was a good tradeoff between filtering power and scanning time for a wide range of data sets.

1000 cluster centers, uniformly at random, and then generating 100 vectors per cluster, by adding standard Gaussian noise.
– Real-world: The Colors, NASA and Listeria SISAP data sets.*

With the vectors, Euclidean distance was used, while with the strings (Listeria), Levenshtein distance was used. For the real-world data sets, the 101 first objects were taken as queries, while for the synthetic data sets, 101 queries were generated in addition. These were then used pairwise (1 and 2, 2 and 3, etc.) as the two query objects in an OWA query with weights 1 and 3 (as in the Gini coefficient). Fairest neighbor queries (kFN) were run for $k = 1, \ldots, 5$. Performance, measured by the number of distance computations needed to resolve the query, was averaged over the 100 query object pairs.

The results are shown in Table 1. As a baseline, the number of distance computations needed for a linear scan is listed, and the speedup for the combined kFN query is shown for each k. As a comparison, we also performed a *double* query, where a separate k was found for each of the two query objects, to ensure that the true kFN would be returned,† and then two separate kNN queries were performed, with the fairest neighbors found in their intersection.

The combined search used fewer distance computations than the alternatives for every data set and parameter setting. On average, the combined search used about half as many distance computations as the separate queries, and a quarter of a full linear scan.‡

5 Conclusions and Future Work

Ordered weighted averages (OWA) and weighted OWAs (WOWA) may be used as a query modality with any current metric index, when tradeoffs between multiple query objects are needed, to find their k fairest neighbors (kFN). They provide a large degree of customizability, both in terms of their fairness profile and the relative weights of different query objects, and are easy to implement. Other monotone (un)fairness measures may also be used, though possibly with weakened overlap checks in some cases.

Some lines of research might involve looking into how index structures may be adapted, perhaps by adjusting various construction heuristics, to such fair neighbor queries, and whether the requirements for efficiency in practice are different from, say, single-object ball queries. One might also wish to look into generalizations of fairness, where one permits negative weights for certain objects, which is straightforward for weighted sum, but whose implementation is less obvious for WOWA.

---

* Available at `https://sisap.org`.

† Of course, these individual k parameters would not be available when resolving a real query, but this gives an optimistic bound for the competition.

‡ More specifically, the average proportions, using the geometric mean, were 48.4 % and 25.7 %, respectively, corresponding to speedups of 2.1 and 3.9.

Table 1. Experimental results. For each of the double (two separate) and combined queries, the speedup from the number of distance computations needed for linear scan is listed for each $k = 1, \ldots, 5$.

			Double					Combined				
Data set	Dim.	Scan	1	2	3	4	5	1	2	3	4	5
Colors	112	225 162	3.29	3.09	2.98	2.91	2.84	5.55	5.22	5.04	4.91	4.80
NASA	20	80 098	1.57	1.46	1.42	1.38	1.36	2.64	2.48	2.39	2.33	2.28
Uniform	4	200 000	2.17	2.14	2.12	2.10	2.09	7.13	6.94	6.82	6.72	6.65
	6	200 000	2.16	2.07	2.02	1.99	1.96	5.73	5.47	5.32	5.20	5.10
	8	200 000	2.01	1.89	1.82	1.78	1.74	4.20	3.95	3.79	3.68	3.59
	10	200 000	1.87	1.73	1.65	1.60	1.56	3.20	2.96	2.82	2.73	2.66
Clustered	4	200 000	2.28	2.25	2.22	2.21	2.19	7.62	7.47	7.37	7.30	7.23
	6	200 000	2.39	2.28	2.22	2.17	2.14	6.17	5.91	5.75	5.62	5.53
	8	200 000	2.07	1.93	1.87	1.82	1.79	4.41	4.12	3.95	3.84	3.75
	10	200 000	1.84	1.72	1.64	1.60	1.57	3.20	2.96	2.83	2.73	2.66
Listeria	—	41 118	1.25	1.17	1.16	1.13	1.06	1.28	1.28	1.27	1.27	1.27

References

1. Chávez, E., Navarro, G.: A compact space decomposition for effective metric indexing. Pattern Recognition Letters **26**(9), 1363–1376 (2005)
2. Ciaccia, P., Patella, M., Zezula, P.: Processing complex similarity queries with distance-based access methods. In: International Conference on Extending Database Technology. pp. 9–23. Springer (1998)
3. Fredriksson, K.: Engineering efficient metric indexes. Pattern Recognition Letters **28**(1), 75–84 (2007)
4. Gonzales, C., Perny, P.: Multicriteria decision making. In: Marquis, P., Papini, O., Prade, H. (eds.) A Guided Tour of Artificial Intelligence Research: Volume I: Knowledge Representation, Reasoning and Learning, pp. 519–548. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-06164-7_16
5. Hetland, M.L.: Comparison-based indexing from first principles. arXiv preprint arXiv:1908.06318 (2019)
6. Tellez, E.S., Chávez, E.: The list of clusters revisited. In: Mexican Conference on Pattern Recognition. pp. 187–196. Springer (2012)
7. Torra, V.: The weighted OWA operator. International Journal of Intelligent Systems **12**(2), 153–166 (1997). https://doi.org/10.1002/(SICI)1098-111X(199702)12:2<153::AID-INT3>3.0.CO;2-P
8. Yianilos, P.N.: Data structures and algorithms for nearest neighbor search in general metric spaces. In: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms. pp. 311–321 (1993)