Updatable Indices for Efficient, Generalised Top-k Queries [Extended Abstract]

Sean Chester
Supervised by Venkatesh Srinivasan, Alex Thomo, and Sue Whitesides
schester@uvic.ca, venkat@cs.uvic.ca, thomo@cs.uvic.ca, sue@uvic.ca

Department of Computer Science
University of Victoria, Canada, STN CSC V8W 2Y2

1 Top-k Queries and Modern-day Data

The way that we interact with databases has changed. The ubiquitous textbook example of finding employees with salaries over 60K and ages under 30 no longer reflects a typical use case in modern applications; instead, users are interested in books, movies, music, documents, services, and users that are similar to another object that they have in mind. This sort of similarity search is often implemented by means of a *top-k query*, wherein each tuple is assigned a score and the tuples with the k highest scores are reported back to the user.

The nature of the databases we query has changed as well, growing to Internet-scale, yet our expectation is for instantaneous results. Thus, this needle-in-a-haystack problem requires intelligent algorithms and data structures: naive solutions do not suffice. Furthermore, most repositories of data grow indefinitely as users add more content. This puts a strain on the design of the data structures, because if they do not handle insertions well, they quickly become obsolete. Although the literature on top-k queries is rather quite dense, there are still gaps in terms of this modern-day usage, especially in terms of designing indices with really efficient update performance.

A top-k query is, in fact, a question of algebra, in which dot product is used to model similarity. Under the assumption that the domain is numeric, every tuple is modelled as a vector, $t = \langle t_1, \ldots, t_d \rangle$ as is the query itself, $q = \langle q_1, \ldots, q_d \rangle$. The objective, then, is to retrieve from a set of tuples $\mathcal{D}$, the k tuples whose dot product with q is largest: $\{t \in \mathcal{D} : ||\{u \in \mathcal{D} : q \cdot u > q \cdot t\}|| \leq k\}$.

In terms of techniques, there are two prominent approaches. Start-from-scratch, single-pass, algorithmic approaches based on the Threshold Algorithm [5] perform quite well in practice, relying on the individual attributes each being available in sorted order. The second approach, the construction of indices inspired by Computational Geometry, considers that asymptotically better performance can be achieved by organising the data effectively with data structures. It is within the latter that this research fits. I seek to determine which algebraic and geometric transformations on the tuple vectors can create a conceptually simple

and efficiently updatable data structure to respond to top-k queries on indefinitely growing, Internet-scale databases and present here ongoing research that addresses related problems with a view to deciding the answer to this question.

2 Top-k Query Processing, From the Nineties 'Til Now

The literature on top-k query processing is dense. After all, the problem has been around since it was introduced by Matoušek and Schwarzkopf in 1992 [7]. In the interest of brevity, I focus here on describing only the most relevant papers to my particular research problem. For more comprehensiveness, Ilyas et al. [6] have a very nice survey on ranked query processing.

A landmark paper in the field is due to Fagin et al. [5], which introduces the Threshold Algorithm, a markedly effective middleware solution to answering top-k queries, especially over disparate data sources. The idea is to retrieve sorted access to each individual attribute and terminate the algorithm early once the sort order implies that the top k results have already been discovered. This (and the many papers based on it) is somewhat tangential to this research, however, in that I seek to design a *reusable* data structure that has asymptotically better long-term performance, rather than an efficient algorithm which is restarted from scratch at each execution.

Within the context of indexing approaches, there is a class of techniques all inspired by Computational Geometry and some of its iconic tools: duality transforms, arrangements, and doubly-connected edge lists. I mention here two especially salient works of this sort, the first due to Tsaparas et al. [9], and the more recent, Das et al. [3]. The idea is to represent the dataset as points and transform them into hyperplanes in the dual space where they form an *arrangement*. The position in the arrangement of a transformed point exactly determines its rank in the dataset. Such an approach can produce a very fast querying mechanism because for any given query, one need only to traverse the first k hyperplanes in an appropriate direction in dual space.

However, there are a few clear manners in which these techniques encounter difficulty. The most apparent of these is the curse of dimensionality. It is a result of Shläfli [8] that an arrangement of n hyperplanes in d dimensions will consist of $\mathcal{O}(n^d)$ cells. Consequently, while most of the research in this family is described in general terms, actual implementation and experimentation has always been done only in two dimensions. There is promise that this *Shläfli Phenomenon* can be addressed: Das et al., in fact, give a rather compelling technique to prune the arrangement to just that which is interesting, with efficacy dependent on the particular dataset.

The other clear challenge is that of updating the data structures. The zone theorem (cf. [4]) states that a new hyperplane intersects $\mathcal{O}(n^{d-1})$ cells of an arrangement of n hyperplanes in d dimensions, suggesting that the cost of updating an arrangement-based data structure is very high. Even the promising aforementioned paper of Das et al. relies on an expensive $\approx \mathcal{O}(n^{d/2})$ halfspace range search to conduct updates to their data structure.

A similar set of techniques is based on the Onion Technique [1]. These index structures layer the dataset based on the relationship among the data points. In the case of the Onion Technique, the layering is done with respect to the convex hull (and subsequent convex layers). The most recent paper of this type is that of Xin et al. [12], which uses a dominance relationship to partially order the data points. The concern for these techniques is that the convex hull and the number of incomparable, undominated points is known to become prohibitive in high dimensions and when attributes are anticorrelated.

So recently as last year, Vlachou et al. [10] introduced a very related problem, the *reverse top-k* query. The question, rather than *which are the top-k tuples for a given query*, is to discover *the queries for which a given tuple is among the top-k*. The relevance of this question cannot be understated: it is, in different terms, the question of inserting a new tuple into an index for top-k querying, reinvigorating the importance of resolving the expense of top-k index updates. The idea to use Computational Geometry-inspired techniques to improve the performance of reverse top-k querying has been illustrated by Wang et al. [11], but their technique still considers a start-from-scratch, single-pass, algorithmic approach rather than one of indexing and reports an experimental order of magnitude improvement rather than better asymptotic guarantees.

Thus, we find today that this research area is as important still as it has been for the past two decades and that something new is needed to appropriately address the curse of dimensionality and particularly the challenge of efficiently updating index structures in the face of dynamic dataset growth.

3 Something New

A recent and a forthcoming result on related problems combine to deliver new insight into this problem of efficiently updating top-k indices:

3.1 Indexing for Vector Projections

The first related problem is that of indexing for threshold vector projection queries [2], which differs from the top-k problem in two ways: the dot product operation is replaced with algebraic projection and the requirement of being among the top k results is replaced with a threshold value above which each tuple must score. That is to say, the goal is instead to produce from a dataset of tuple vectors $\mathcal{D}$, given a threshold τ and query vector q, the subset $\{t \in \mathcal{D} : (t \cdot q)/||q|| \geq \tau\}$. This is a slightly more general problem than the top-k formulation I gave in the introduction in that the resultant structure can be used to resolve top-k queries, but the threshold problem allows for conjoining queries–something that is undefined for top-k. The paper proceeds by solving for the nullspace of each tuple vector, effectively transforming into dual space with a meaningful interpretation (rather than arbitrarily as in the case of other duality transforms). More importantly, it derives a static solution space for each vector which can then be spatially indexed. Appropriate spatial indices are surveyed.

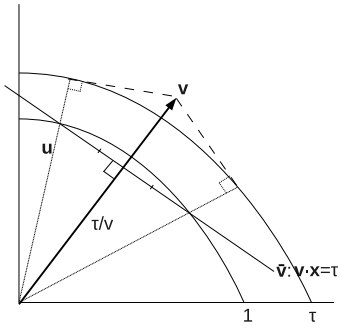

Fig. 1. An illustration, taken from Chester et al. [2], of a vector **v** and its translated nullspace (the line orthogonal to **v**) with which the index structure is built.

The algebraic transformation on one vector is illustrated in Figure 1. This is an interesting contribution in terms of the top-k problem because it provides a very different means of employing duality transform, the consequences of which are witnessed in the next problem.

3.2 Reverse Top-k Queries with De l'Origine Depth Contours

The purposeful duality transform demonstrated for the projection threshold queries can be utilized quite successfully for the reverse top-k query problem introduced by Vlachou et al. [10]. In work as yet unpublished but in the late stages of manuscript preparation, I use the transformed tuples to construct an arrangement (i.e., a set of faces, edges, and vertices induced in hyperspace by a set of intersecting hyperplanes) to design an efficient indexing data structure. The arrangement is important because the rank of each tuple for a particular query is precisely the DD (*de l'origine depth*, a concept we introduce) of the hyperplane into which it is transformed. By constructing DD contours from the arrangement and convexifying the resultant polygon, we achieve a reverse top-k index structure that can be queried in $\mathcal{O}(lg\ n)$ and built in $\mathcal{O}(nk)$ time. A secondary but also important contribution is the identification of new data independence, namely that the contours are independent of each other.

This transformation and the construction of DD contours is compared visually to the technique of Das et al. in Figure 2.

4 Toward a DD-based Updatable Top-k Index

The ideas developed for these last two problems offer substantial potential in terms of addressing the literature gap of efficiently handling insertions into top-k indices. This potential arises primarily out of two crucial characteristics of the DD contours. First, the contours are independent and the only communication needed among them may be in propagating changes. This is true, too, of other

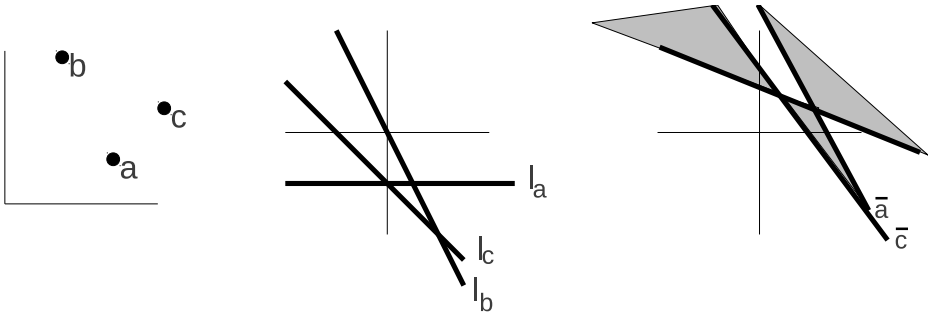

Fig. 2. A sample dataset of three points and the Das et al. and Chester et al. transforms. Note the contours induced by the lines, alternately filled grey.

duality-based techniques, but has not yet been recognised. Second, the convexification of each contour leads to the logarithmic query time, a technique unique to *de l'origine depth*.

My future work is in exploring this potential. The DD concept and associated ideas are optimised for reverse top-k research and need to be adapted for the creation of indexes for traditional top-k. Much of the existing implementation is reusable, but determining the precise algorithmic details will take modest effort. Additionally, quite how to synthesise this research with other promising contributions (such as the pruning technique of Das et al.) needs deep consideration.

The logarithmic query time arises out of the convexification of the contours and the subsequent recursive algorithm for intersecting a convex polygon with a line. Generalising this result to higher dimension (i.e., researching *disk-aware* techniques for intersecting a convex polytope with a hyperplane) would lead to an indexing technique effective in arbitrary dimension, something as yet unimplemented in any experimental study.

Given the success of this research–an efficiently updatable top-k index for arbitrary dimension–there is ample opportunity to exploit the independence among contours and among subdivisions of Euclidean space with hardware acceleration, such as GPUs. The potential here is exciting as a means to combat the curse of dimensionality and arises out of the data-flow-centric, sequential, memory-constrained design that was central in the conceptualisation of this research from the beginning. Especially because the implementation has been written so far in C, the translation into languages like CUDA will be (at least to what extent possible) relatively straight-forward.

5 Conclusion

In this paper I outlined the current state of index-based top-k query processing and described the shortcomings that exist at this point in the literature. Namely,

evaluation of the techniques is only done in two dimensions and updates, especially in higher dimensions, are not handled especially efficiently. These are both important considerations because the modern-day setting of databases is one of Internet scale and indefinite growth and because a two-dimensional relation does not offer the user the flexibility that ad-hoc top-k querying is meant to provide.

I shared the research that I have completed and have forthcoming on two problems that are of substantial connection to these shortcomings, that of indexing for vector projections and that of efficiently indexing for reverse top-k queries. Finally, I indicated the three research directions in which I intend to use this research and the existing literature in order to resolve the shortcomings on the top-k query problem.

References

1. Chang, Y.C., Bergman, L., Castelli, V., Li, C.S., Lo, M.L., Smith, J.R.: The onion technique: indexing for linear optimization queries. In: Proceedings of the 26th SIGMOD International Conference on Management of Data. ACM (2000)
2. Chester, S., Thomo, A., Venkatesh, S., Whitesides, S.: Indexing for vector projections. In: Proceedings of The Database Systems for Advanced Applications, 16th International Conference, Part II, LNCS 6588. pp. 367–376. Springer-Verlag, Berlin Heidelberg (April 2011)
3. Das, G., Gunopulos, D., Koudas, N., Sarkas, N.: Ad-hoc top-k query answering for data streams. In: Proc. International Conference on Very Large Databases (VLDB). pp. 183–194. ACM, New York, NY (2007)
4. Edelsbrunner, H., Seidel, R., Sharir, M.: On the zone theorem for hyperplane arrangements. In: Maurer, H. (ed.) New Results and New Trends in Computer Science, Lecture Notes in Computer Science, vol. 555, pp. 108–123. Springer Berlin / Heidelberg (1991)
5. Fagin, R., Lotem, A., Naor, M.: Optimal aggregation algorithms for middleware. Journal of Computer and System Sciences 66, 614–656 (2003)
6. Ilyas, I.F., Beskales, G., Soliman, M.A.: A survey of top-k query processing techniques in relational database systems. ACM Computing Surveys 40 (October 2008)
7. Matoušek, J., Schwarzkopf, O.: Linear optimization queries. In: Proceedings of the 8th Annual Symposium on Computational Geometry. ACM (June 1992)
8. Shläfli, L.: Theorie der vielfachen kontinuität. Denkschriften Der Schweizerischen Naturforschenden Gesellschaft 38, 1–237 (1901)
9. Tsaparas, P., Koudas, N., Palpanas, T.: Ranked join indices. In: Dayal, U., Ramamritham, K., Vijayaraman, T.M. (eds.) Proceedings International Conference on Data Engineering (ICDE). pp. 277–288. IEEE Computer Society (2003)
10. Vlachou, A., Doulkeridis, C., Kotidis, Y., Norvag, K.: Reverse top-k queries. In: Data Engineering (ICDE), 2010 IEEE 26th International Conference on. pp. 365–376. IEEE (March 2010)
11. Wang, B., Dai, Z., Li, C., Chen, H.: Efficient computation of monochromatic reverse top-k queries. In: Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on. vol. 4, pp. 1788–1792. IEEE (August 2010)
12. Xin, D., Chen, C., Han, J.: Towards robust indexing for ranked queries. In: Proc. International Conference on Very Large Databases (VLDB). pp. 235–246. ACM, New York, NY (2006)