Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 371 (2007) 148–154
Theoretical Computer Science
www.elsevier.com/locate/tcs

Approximately dominating representatives☆

Vladlen Koltun*, Christos H. Papadimitriou

Computer Science Division, University of California, Berkeley, CA 94720-1776, USA

Abstract

We propose and investigate from the algorithmic standpoint a novel form of fuzzy query called *approximately dominating representatives* or ADRs. The ADRs of a multidimensional point set consist of a few points guaranteed to contain an approximate optimum of *any* monotone Lipschitz continuous combining function of the dimensions. ADRs can be computed by appropriately post-processing Pareto, or "skyline", queries [Kian-Lee Tan, Pin-Kwang Eng, Beng Chin Ooi, Efficient progressive skyline computation, in: VLDB, 2001, pp. 301–310; Wolf-Tilo Balke, Ulrich Güntzer, Jason Xin Zheng, Efficient distributed skylining for web information systems, in: EDBT, 2004. [14]]. We show that the problem of minimizing the number of points returned, for a user-specified desired approximation, can be solved in polynomial time in two dimensions; for three and more it is NP-hard but has a polynomial-time logarithmic approximation. Finally, we present a polynomial-time, constant factor approximation algorithm for three dimensions.

1. Introduction

In recent years there has been much interest in "fuzzy" queries in databases [7], in which a user seeks an object that is not maximal with respect to one criterion, but is "good" in several respects simultaneously. In a multimedia database, for example, we may want to find a document in the corpus that best matches specifications of color (such as "orange") and shape (perhaps "round"). In querying a restaurant database we may seek an establishment that is near our current location, of high quality, inexpensive, etc. MapQuest may want to return to a user a path that minimizes some combination of distance, delay, number of turns, tolls, fuel consumption, etc. Notice that, in the above examples, we assume that we know the precise valuations of the objects (documents, restaurants, paths) in various dimensions (shape score, color score, restaurant quality, tolls, etc.); the fuzzy part is that we do not know the exact objective that we wish to optimize—presumably some function combining all these criteria.

There has been much work within database research both in defining and implementing such queries. Among the approaches taken and subproblems attacked are these:

- Exploring principled, axiomatic ways that lead to the proper definition of the combining function (see, e.g., [7, 9,10]). The main difficulty here (the interesting and elegant results in these papers notwithstanding) is that the

☆ Work on this paper was supported by an NSF ITR grant and a France–Berkeley Foundation grant.
* Corresponding author.
E-mail addresses: vladlen@cs.berkeley.edu (V. Koltun), christos@cs.berkeley.edu (C.H. Papadimitriou).

0304-3975/$ - see front matter © 2007 Published by Elsevier B.V.
doi:10.1016/j.tcs.2006.11.003

combining function depends in crucial and complex ways on the application and the intent of the query, and there are no universal rules to guide us here (except for some commonsense properties such as monotonicity).

- Assuming that we know a combining function, solving the algorithmic problem of retrieving *the k best items* with respect to this function [8]. The algorithms here are analyzed either in terms of an assumed distribution of the criteria values, or in a peculiar worst-case sense applicable to middleware algorithms. The k best goal is a reflection of the fact that, deep down, we are not sure about the combining function, and so we want enough results to cover as many possibilities as feasible.
- A more elaborate version of the latter method requests feedback from the user (such as singling out one of the k results as the "best", in a manner similar to Google's "more like this"), and uses this feedback to modify in a heuristic way the combining function and retrieve new, presumably subjectively better, results [5].
- A more radical and principled approach is to be completely agnostic about the combining function, and retrieve all objects that *could* be best under *some* monotonic combining function. This leads us directly to multiobjective optimization [12,13] and to *Pareto sets*, most often known in the database literature as "skyline queries". These return any object that is not dominated (in all criteria) by some other object; it can be shown (see Proposition 2.1) that these are precisely the objects that could be best under some monotonic combining function. (If we know that the combining function is linear, then we have a convex, smaller variant of the skyline query, see Proposition 2.2.) There are algorithms that compute these sets as quickly as it can be hoped (see, e.g., [1]). The downside here is that the skyline set of a large multidimensional point set can be (and is typically) huge: the skyline set of a random n-point subset of $[0, 1]^d$ contains $\Theta(\log^{d-1} n)$ in expectation [2]. Hence, skyline queries will typically flood the user with a large portion of the whole database.

In this paper we introduce *approximately dominating representatives*, or *ADRs*, a refinement of the skyline queries that remedies the output volume problem at a small (and controlled) loss of accuracy. For any $\varepsilon \geq 0$ specified by the user, an ε-ADR query will return a set of objects $\{a_1, \ldots, a_k\}$ that has the following property: for any other object a in the database, there is an $i \leq k$ such that the vector $a_i \cdot (1 + \varepsilon)$ (i.e., a_i boosted by ε in all dimensions) dominates a in all dimensions.

The ADR concept was inspired by the work in [12] (see also [13] for a database-related application) on approximate multiobjective optimization, and our basic results are variants of results there. However, [12] focused on combinatorial optimization problems, where instead of a database of objects we have an implicitly represented exponential set of feasible solutions (say, all spanning trees of n nodes), evaluated by multiple cost functions. [12] did not consider the problem of minimizing the size of the set of ADRs, the focus of our technical results.

There are several variants of ADRs that could be advantageous under some conditions and could be offered as options in an implementation: additive approximation instead of proportional; different approximations specifications for different criteria; the convex version alluded to in Proposition 2.2; and combinations of the above. In this paper we focus on the basic ADR (proportionate, uniform approximation of the whole skyline); our results can be extended painlessly to the other variants. The only exception is that, in a couple of our proofs we use the additive variant because it is simpler to explain (and is equivalent if one considers the logarithm of each criterion).

The skyline of a database (that is, the ε-ADR with $\varepsilon = 0$) is unique. For $\varepsilon > 0$, on the other hand, there may be many ADRs, varying significantly in size. However, an ADR can be considerably smaller than the exact skyline. We show that there always exists an ε-ADR of size $O((\frac{1}{\varepsilon} \log \frac{M}{m})^d)$–independent of n–where we assume that we have d criteria dimensions, and that the values of each criterion are between m and M (Proposition 2.4). This is a worst-case estimate: among the many ADRs of a database some will typically be much smaller, and our algorithms strive to find the smallest possible ADR for the given database and ε. For example, if the database has a "knee" or "sweet point"–that is, a point that approximately dominates all others–our ADR algorithms will find it and the query will return only this point. We show that an ADR is guaranteed to contain a close approximation of the optimum object under the (unknown) true combining function, as long as this function is monotonic and satisfies a Lipschitz continuity condition (Proposition 2.5).

Thus, the main problem that we attack in this paper is the following: *given a set of n points in d dimensions and* $\varepsilon > 0$*, find the smallest possible ε-ADR*. We show that the problem can be solved in linear time in two dimensions by a greedy algorithm (Theorem 3.2), and is NP-hard in higher dimensions (Theorem 4.1). However, it can be approximated within a factor of $\ln n$ by a trivial reduction to the SET COVER problem. It is an open problem whether

better approximation algorithms are possible; we conjecture that $\ln n$ is a lower bound for this problem (it is for the general set cover problem, see [11]).

We assume in general that we have computed the skyline query of the set, and our task is to post-process it to determine the smallest possible ADR (in the 2-dimensional case, our algorithm can be made optimal in the sense of [8], by running directly on the database).

Finally, we show that in three dimensions constant factor approximation is possible. Although the approximation factor is quite large, the point of our result is that alternatives to the greedy algorithm exist in three dimensions (even though the current state of the art can prove only very conservative bounds for them), and thus the $\ln n$ lower bound conjectured above for unbounded dimension does not hold in three dimensions. Unfortunately, we know of no techniques that can establish an inapproximability results for lower approximation factors.

A final note: very recently we became aware of independent work by Vassilvitskii and Yannakakis [15] which, even though on a different if related topic, reaches conclusions remarkably parallel to ours. They focus on multiobjective optimization problems, as opposed to static multi-dimensional skyline queries. Their goal is to obtain approximate Pareto curves with as few points as possible for the given ε. They approximate the optimum number of points within a factor of 3 in two dimensions; in three dimensions they show that no such approximation is possible, but they are able to obtain one if the ε is tripled or so. And in higher dimensions they show that even this is impossible.

2. Preliminaries

We assume that the database has been reduced to the values of n objects with respect to d criteria. Thus, a database A is a finite subset of $[m, M]^d$, with $|A| = n$, where $0 < m < M$ are assumed to be fixed. We say that $a \in A$ dominates $a' \in A$, written $a \succeq a'$, if for all coordinates $i = 1, \ldots, d$ we have $a_i \geq a'_i$.

We further assume that all criteria are to be maximized. (Otherwise, a reversal of axis would do the trick.) The *skyline* (or *Pareto set*) of A is the set $\text{SKY}[A] = \{a \in A : \text{for all } a' \in A \setminus \{a\}, a' \not\succeq a\}$.

We are not certain that the following justification of the skyline has been brought to the attention of the database community. Call a function $f : \Re^d \mapsto \Re$ *monotonic* if $a \succeq a'$ implies $f(a) \geq f(a')$.

Proposition 2.1. *A point $a \in A$ is in* $\text{SKY}[A]$ *if and only if there is a monotonic function f such that $a = \arg\max_{a' \in A} f(a')$.*

Thus, $\text{SKY}[A]$ contains all possible optima of all possible monotonic combining functions. If we further know that f is linear, that is, of the form $f(a) = a \cdot c$ for some nonnegative vector c, then the possibilities are restricted in an interesting way. Define the *convex skyline of A* to be $\text{C-SKY}[A] = \{a \in A : \sum_{a' \in A \setminus \{a\}} \lambda_{a'} a' \not\succeq a$, where $\sum_{a' \in A \setminus \{a\}} \lambda_{a'} = 1$, $\min_{a' \in A \setminus \{a\}} \lambda_{a'} \geq 0\}$. The convex skyline omits points that are dominated by convex combinations of others.

Proposition 2.2. *A point $a \in A$ is in* $\text{C-SKY}[A]$ *if and only if there is a linear function f such that $a = \arg\max_{a' \in A} f(a')$.*

$\text{SKY}[A]$ is typically quite large:

Proposition 2.3 ([2]). *If A consists of n points drawn uniformly at random from $[m, M]^d$, the expected size of* $\text{SKY}[A]$ *is $\Theta(\log^{d-1} n)$.*

Fix a database A and $\varepsilon \geq 0$. A *set of ε-approximate dominating representatives*, or an *ε-ADR*, is a subset D of A that has the following property. For every $a \in A$ there is a $b \in D$ such that $(1 + \varepsilon) \cdot b \succeq a$. An ADR can be significantly smaller than $\text{SKY}[A]$:

Proposition 2.4. *There is always an ε-ADR of size $O((\frac{1}{\varepsilon} \log \frac{M}{m})^{d-1})$.*

Proof. Subdivide $[m, M]^d$ into axis-parallel orthogonal domains as follows. Consider the set of values

$$v_j = m(1 + \varepsilon)^j \quad \text{for } 0 \leq j \leq \left\lceil \log_{1+\varepsilon} \frac{M}{m} \right\rceil,$$

a geometric progression with step $1+\varepsilon$, minimal value m, maximal value above M and cardinality $N = O(\frac{1}{\varepsilon}\log\frac{M}{m})$. For any set of indices $1 \le j_i \le N-1$, for $1 \le i \le d$, consider the axis-parallel orthogonal domain

$$D_{j_1,\ldots,j_d}\left\{a \in [m,M]^d \;\middle|\; \forall 1 \le i \le d \,.\, V_{j_i} \le a_i \le V_{j_i+1}\right\}.$$

The number of such domains is $O((\frac{1}{\varepsilon}\log\frac{M}{m})^d)$.

For every $1 \le j_1, \ldots, j_{d-1} \le N-1$, let j_d be the largest index such that the domain $D_{j_1,\ldots,j_{d-1},j_d}$ contains at least one point of A (if no such index exists, do nothing). Then choose one point of A from each such domain, and call the resulting set of points D; it is clear that $|D| = O((\frac{1}{\varepsilon}\log\frac{M}{m})^{d-1})$.

We claim that D is an ε-approximate ADR. In proof, consider a point $a \in A$; it must lie within some domain $D_{j_1,\ldots,j_d}$. Thus, D contains a point from this domain, or from one with higher d-th coordinate, call it b. It is immediate that $(1+\varepsilon)\cdot b \succeq a$. □

Call a function $f : \Re^d \mapsto \Re$ *log-Lipschitz continuous with constant C* if $f((1+\varepsilon)\cdot a) \le (1+C\varepsilon)\cdot f(a)$. Most common combining functions (such as linear combinations, max, etc.) are log-Lipschitz continuous with constant one. The following generalization of Proposition 2.1 is a justification of ε-ADR:

Proposition 2.5. *Any ε-ADR contains a point whose value is within a factor of $(1+C\varepsilon)$ of the optimal, for any monotone combining function that is log-Lipschitz continuous with constant C.*

Proof Sketch. Let $a \in A$ be the optimum. There is an a' in the ADR that satisfies $(1+\varepsilon)a' \succeq a$, and thus $(1+C\varepsilon)f(a') \ge f((1+\varepsilon)a') \ge f(a)$. □

Finally, we point out a straightforward connection of ADRs with the SET COVER problem. For each $a \in A$ define $S_a = \{a' \in A : (1+\varepsilon)\cdot a \succeq a'\}$. It is easy to see that, if D is an ε-ADR, then $\bigcup_{a\in D} S_a = A$. Hence, finding a good ε-ADR is tantamount to finding a small set cover. In fact, since all points in A are dominated by $\text{SKY}[A]$, it suffices to consider the intersections of S_a with $\text{SKY}[A]$. This latter observation often yields faster algorithms, since we can pre-process A to compute $\text{SKY}[A]$ [1] and select from this our ε-ADR.

3. Two dimensions

Assume that the points are sorted in decreasing first coordinate. We introduce a greedy algorithm that, for a given ε, works as follows. Consider the point a with the highest first coordinate, and let $B[a]$ be the set of all points b such that $a \in S_b$.

Lemma 3.1. *There is a $b^* \in B[a]$ such that $S_{b^*} = \bigcup_{b\in B[a]} S_b$.*

Proof. Let $C \subseteq A$ be the set of points b whose first coordinate b_1 satisfies $a_1/(1+\varepsilon) \le b_1 \le a_1$, and let b^* be the point in C that has largest second coordinate (C is nonempty because, by definition, it contains a). We shall first show that S_{b^*} contains a. That $b^* \in C$ implies that b^* covers a (within $(1+\varepsilon)$) in the first coordinate; and the maximality condition implies that the second coordinate is also covered; this is because C includes a itself, and so the second coordinate of b^* can only be larger. Now, since a is the point in A with the largest first coordinate, for each $b \in B[a]$, S_b must be contained in S_{b^*}. □

In other words, among all points $B[a]$ that could be selected in the ADR to cover a, there is one, b^*, whose S_{b^*} is maximal. Hence, since an ADR must contain one of these points, there is no reason not to pick b^* for inclusion in the ADR. This suggests the following greedy algorithm:

```
Input: Point set A ⊆ [m, M]^2, ε > 0
Output: Set of points D ⊆ A, the smallest ε-ADR of A
set b* = (M, 0) and D = ∅;
while there is a point a in A with a_1 ≤ b*_1 such that (1 + ε) · b* ⋡ a
do:
  select a to be the point with the highest a_1 among those;
  find the point b* with a_1/(1 + ε) ≤ b*_1 ≤ a_1 with highest b*_2;
  add b* to D;
```

Theorem 3.2. *The greedy algorithm computes in linear time the ε-ADR with the fewest points.*

Proof. It does compute an ε-ADR, because each chosen point b^* covers all points in A with first coordinate between the current a_1 (included) on the high end and the next a_1 (excluded) on the low side. Since a starts with the point with largest a_1 and in the end b^* dominates all points with first coordinate less than or equal to the current b_1^*, the collection D covers all of A. Optimality follows from Lemma 3.1, since b^* is always chosen to be the point whose S_{b^*} contains all other S_b's that cover the rightmost uncovered point in A. □

For efficiency, we can run this algorithm on the precomputed set $\text{SKY}[A]$, instead of A. Furthermore, it is not hard to see that a variant of this algorithm, alternating between the two coordinates, can be run directly on the set A and be optimized to stop at the earliest possible instant at which the ε-ADR has been found and validated, thus not examining large parts of A. This algorithm is optimal in the middleware sense of [8].

4. NP-hardness

We show the following:

Theorem 4.1. *It is an NP-hard problem, given a point set $A \subseteq [0, 1]^3$ and an $\varepsilon > 0$, to find the ε-ADR with the fewest points.*

Proof Sketch. For simplicity we shall consider the ε-ADR problem under the *additive* definition of approximate dominance; that is, $a \in S_b$ if $a_i \leq b_i + \varepsilon$ for $i = 1, 2, 3$. The result for the ordinary definition of ADR follows trivially by mapping the set A of points constructed in the reduction to a set of points A' with coordinates $a'_i = \exp a_i$ for all $a \in A$ and $i = 1, 2, 3$.

The reduction is from 3SAT. We are given a Boolean formula with n clauses with 2 or 3 literals each. Set $\delta = 1/4n$ and $\varepsilon = \delta/n$. We shall create a set A of points, all lying just below the plane $x + y + z = 1$, such that the optimal ε-ADR of A reveals whether the formula is satisfiable.

The proof requires a number of gadgets. The *flip-flop* consists of the following points: $a = (-\delta, 0, 0)$, $b = (0, -\delta, 0)$, $t = (-2\varepsilon, -\varepsilon, -\varepsilon)$ and $f = (-\varepsilon, -2\varepsilon, -\varepsilon)$. The basic property of the flip-flop is this: $S_f = \{f, t, a\}$, $S_t = \{f, t, b\}$, and S_a, S_b are singletons. A good cover will contain exactly one of t (which will mean a literal will be true) and f (false). Such flip-flops can be combined in tandem with the b point of one coinciding with the a point of the next to form paths that will propagate the values of the literals. There are six variants of the flip-flop, by permuting dimensions, and six more in which b is instead $(0, -\delta/2, -\delta/2)$. (These variants are needed for "bending" the paths.)

The *clause* gadget has points $c_1 = (-\delta/2, -\delta/2, 0)$, $c_2 = (0, -\delta/2, -\delta/2)$, $c_3 = (-\delta/2, 0, -d/2)$ plus other points $d_1 = (-\varepsilon, -\varepsilon, -2\varepsilon)$, $d_2 = (-2\varepsilon, -\varepsilon, -\varepsilon)$, $d_3 = (-\varepsilon, -2\varepsilon, -\varepsilon)$, and the non-singleton approximately dominated sets are now $S_{d_1} = \{d_1, d_2, d_3, c_2, c_3\}$, $S_{d_2} = \{d_1, d_2, d_3, c_1, c_3\}$, $S_{d_3} = \{d_1, d_2, d_3, c_1, c_2\}$. Thus if the three c_i points of a clause gadget coincide with the "true" endpoints of three literal chains, there is a way to cover all six points with one representative iff the truth assignment suggested by the choices in the literal chains satisfies the clause.

To complete the construction, we embed the set of clauses on the $x + y + z = 1$ plane as follows. We choose a point for every variable, a point for every clause, and a curve connecting each variable with each clause where it appears (we assume that each variable has one positive and two negative appearances). The curves are such that they are well-separated by at least δ, except of course for their endpoints and their crossovers. Then we have a flip-flop at every variable point (if the coordinates of the variable point are (x, y, z) then we add this vector to the coordinates of the points a, b, t, f of the gadget), repeat for the clauses, and we replace each curve with flip-flops in tandem.

Which brings us to the last gadget, the *crossover*. It consists of four points, $a = (-\delta/2, -\delta/2, 0)$, $b = (-\delta/2, 0, -\delta/2)$, $c = (-\varepsilon/2, -\varepsilon/3, -2\delta + \varepsilon)$, $d = (0, -\delta/2, -\delta/2)$, plus the points $tt = (-2\varepsilon, -\varepsilon, -\varepsilon)$, $tf = (-3\varepsilon/2, -\varepsilon, -2\varepsilon)$, $ff = (-\varepsilon, -3/2\varepsilon, -2\varepsilon)$, $ft = (-\varepsilon, -2\varepsilon, -\varepsilon)$. The coverage of these points is $S_{tt} = \{tt, ft, ff, tf, a, b\}$, $S_{tf} = \{tt, ft, ff, tf, b, c\}$, $S_{ff} = \{tt, ft, ff, tf, c, d\}$, $S_{ft} = \{tt, ft, ff, tf, d, a\}$, and thus represents a valid crossover between flip-flops (a, c) and (b, d). Placing this gadget where flip-flop paths cross completes the construction.

Let g be the total number of gadgets used in this construction. It is easy to see that the resulting set of points A has an ε-ADR with g points in it (that is, there is a way to choose a point from each gadget so that all of A is covered) iff the original formula was satisfiable. □

It is open whether the problem is MAXSNP-hard, that is, hard to approximate arbitrarily close. We conjecture that it is.

5. Approximation

As partial consolation for Theorem 4.1 we have:

Proposition 5.1. *There is a polynomial-time algorithm that approximates ADRs in any dimension within a factor of* $\ln n$ *of the optimum.*

Proof. Recall the sets $S_a = \{a' \in A : (1+\varepsilon)\cdot a \succeq a'\}$, one for each $a \in A$. It is easy to see that ADRs are precisely covers of A by these sets. And it is well-known [16] that the set cover problem can be approximated by the greedy algorithm ("add the set that contains the largest number of yet uncovered points"), which yields a cover that is within $\ln n$ of the optimum. □

Moreover, when $d = 3$, a *constant* approximation ratio is possible:

Proposition 5.2. *There is a polynomial-time algorithm that approximates ADRs in three dimensions within a constant factor of the optimum.*

The result is established using the SET COVER algorithm of [4]. Unfortunately, no good bounds have been calculated for the approximation ratio of this algorithm; it seems to be in the hundreds, making the result of limited value.

First some definitions. A *negative octant* $O^-(a)$ of a point $a \in \Re^3$ is the closed set $\{a' \in \Re^3 | a \succeq a'\}$ of all points dominated by a. Let $S = \text{SKY}[A]$ and let S^ε be the *ε-boosted* set $\{a\cdot(1+\varepsilon) | a \in S\}$. For a set X and a set $R \subseteq 2^X$ of subsets of X, the pair (X, R) is said to be a *set system*. Given a set system (X, R) and a parameter r, a subset $N \subseteq R$ is said to be a *$1/r$-net* for (X, R) if $x \in \nu$, for some $\nu \in N$, for all $x \in X$ that are contained in at least $|R|/r$ sets of R. In other words, a $1/r$-net covers all elements of X that are "heavily covered" by R.

Theorem 5.3 (*[4]*). *Suppose* (X, R) *admits a* $1/r$*-net of size at most* cr *for a constant* c *and for any* $1 \le r \le |R|$*, and such a* $1/r$*-net can be computed in polynomial time in* $|X|$*. Then a set cover for* (X, R) *of size* $4c \cdot OPT$ *can be computed in polynomial time, where OPT is the size of the smallest set cover for* (X, R).

We now establish that the set system $(S, \mathcal{O}^-)$ admits a $1/r$-net of size $O(r)$, where $\mathcal{O}^- = \{O^-(a) \cap S, a \in S^\varepsilon\}$. The recent result of Clarkson and Varadarajan [6] implies that a $1/r$-net of size $O(r)$ for a geometric set system (X, R) exists and can be found in polynomial time if the union $\bigcup_{\rho\in R'} \rho$ has complexity $O(|R'|)$, for any $R' \subseteq R$. (The complexity of the union of a set of geometric objects is said to be the number of geometric features on its boundary.) In our case this requires showing that the union of n negative octants in 3-space has complexity $O(n)$. This holds by a simple geometric charging argument or as a special case of the bound of Boissonnat et al. [3] on the union complexity of axis-parallel unit cubes. Clarkson and Varadarajan [6] additionally need a decomposition of the complement of the union into a linear number of sets of constant description complexity, which is easily seen to exist in our setting.

Given this $1/r$-net construction, Theorem 5.3 implies that SET COVER for $(S, \mathcal{O}^-)$ can be approximated within a constant factor in polynomial time. This also serves as an approximation scheme for ADRs in three dimensions, and implies Proposition 5.2.

6. Discussion and open problems

We introduce ε-approximate ADR, a novel multi-dimensional query that refines the skyline query, and has the advantage that the answers contain considerably fewer points, without being much less informative. For 2-dimensional sets of points the size of this query can be minimized exactly, while for higher dimensions, approximately.

Many questions remain open. What is the precise approximation ratio? Does the constant approximation result extend to 4 or more dimensions? This requires finding linear-size $1/r$-nets for collections of negative octants in higher-dimensional spaces, which seems quite hard.

Are there limits to the approximation of ADRs? We suspect that a more careful reduction would produce a minuscule such limit for three dimensions; larger lower bounds would probably require specialized PCPs (of which, by the way, none is known to us for geometric problems).

Does the greedy algorithm for set cover perform better than its worst case estimate $\ln n$? We suspect that it does not. In fact, we conjecture that a $\ln n$ lower bound is possible if the dimension is unbounded (by reversing the reduction to SET COVER back to the result of [11]) for instance.

This work also suggests some interesting experiments. How do our algorithms (exact and approximate ones) behave in practice? And, perhaps most importantly, are the results of ADR queries satisfactory to users in typical situations?

References

[1] Wolf-Tilo Balke, Ulrich Güntzer, Jason Xin Zheng, Efficient distributed skylining for web information systems, in: EDBT, 2004.
[2] J.L. Bentley, H.T. Kung, M. Schkolnick, C.D. Thompson, On the average number of maxima in a set of vectors and applications, Journal of the ACM 25 (1978) 536–543.
[3] Jean-Daniel Boissonnat, Micha Sharir, Boaz Tagansky, Mariette Yvinec, Voronoi diagrams in higher dimensions under certain polyhedral distance functions, Discrete & Computational Geometry 19 (1998) 473–484.
[4] Hervé Brönnimann, Michael T. Goodrich, Almost optimal set covers in finite VC-dimension, Discrete & Computational Geometry 14 (4) (1995) 463–479.
[5] Kaushik Chakrabarti, Kriengkrai Porkaew, Sharad Mehrotra, Refining top-k selection queries based on user feedback, in: VLDB, 2000.
[6] Kenneth L. Clarkson, Kasturi Varadarajan, Improved approximation algorithms for geometric set cover, in: SoCG 2005 (in press).
[7] Ronald Fagin, Fuzzy queries in multimedia database systems, in: PODS, 1998, pp. 1–10, invited.
[8] Ronald Fagin, Amnon Lotem, Moni Naor, Optimal aggregation algorithms for middleware, in: PODS, 2001.
[9] Ronald Fagin, Combining fuzzy information from multiple systems, Journal of Computer and System Sciences 58 (1) (1999) 83–99.
[10] Ronald Fagin, Edward L. Wimmers, A formula for incorporating weights into scoring rules, Theoretical Computer Science 239 (2) (2000) 309–338.
[11] Carsten Lund, Mihalis Yannakakis, On the hardness of approximating minimization problems, Journal of the ACM 41 (5) (1994) 960–981.
[12] Christos H. Papadimitriou, Mihalis Yannakakis, On the approximability of trade-offs and optimal access of web sources, in: FOCS, 2000, pp. 86–92.
[13] Christos H. Papadimitriou, Mihalis Yannakakis, Multiobjective query optimization, in: PODS, 2001.
[14] Kian-Lee Tan, Pin-Kwang Eng, Beng Chin Ooi, Efficient progressive skyline computation, in: VLDB, 2001, pp. 301–310.
[15] Sergei Vassilvitskii, Mihalis Yannakakis, Efficiently computing succinct trade-off curves, in: ICALP, 2004, pp. 1201–1213.
[16] Vijay V. Vazirani, Approximation Algorithms, Springer-Verlag, 2001.