Domination Mining and Querying

Apostolos N. Papadopoulos Apostolos Lyritsis Alexandros Nanopoulos
Yannis Manolopoulos

Department of Informatics, Aristotle University, Thessaloniki 54124, Greece
{papadopo,lyritsis,ananopou,manolopo}@csd.auth.gr

Abstract. Pareto dominance plays an important role in diverse application domains such as economics and e-commerce, and it is widely being used in multicriteria decision making. In these cases, objectives are usually contradictory and therefore it is not straightforward to provide a set of items that are the "best" according to the user's preferences. Skyline queries have been extensively used to recommend the most dominant items. However, in some cases skyline items are either too few, or too many, causing problems in selecting the prevailing ones. The number of skyline items depend heavily on both the data distribution, the data population and the dimensionality of the data set. In this work, we provide a dominance-based analysis and querying scheme that aims at alleviating the skyline cardinality problem, trying to introduce ranking on the items. The proposed scheme can be used either as a mining or as a querying tool, helping the user in selecting the mostly preferred items. Performance evaluation based on different distributions, populations and dimensionalities show the effectiveness of the proposed scheme.

1 Introduction

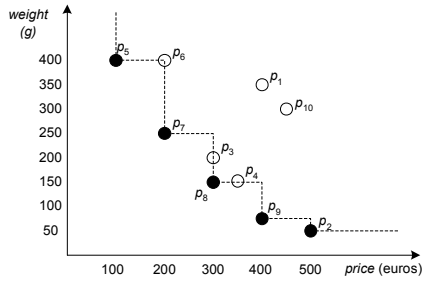
Preference queries are frequently used in multicriteria decision making applications, where a number of (usually) contradictory criteria participate towards selecting the most convenient answers to the user. Each item is represented as a multidimensional point.

Assume that someone is interested in purchasing a PDA device. Unfortunately, there are a lot of criteria that should be balanced before a wise choice is made. Assume further that the customer focuses on two important parameters of a PDA, the price, and the weight of the device. Therefore, the "best" PDAs are the ones that are cheap and light-weighted. Unfortunately, these two criteria are frequently contradictory and therefore, the number of candidates should be carefully selected.

In this example, we have two attributes, and the user is interested in items that have values in these attributes as minimum as possible. Depending on the semantics of each attribute, in other cases the user may ask for maximization of the attributes, or any combination (minimization in some attributes and maximization in the others). For example, if the user focuses on price and available memory, the "best" PDAs are the ones that are as cheap as possible and have

the largest available memory. Without loss of generality, in the sequel we focus on minimizing the attributes of interest.

A fundamental preference query is the skyline query. The skyline of a set of points $\mathcal{S}$ comprises all points that are not dominated. A point p_i dominates another point p_j, if p_i is as good as p_j in all dimensions and it is better than p_j in at least one of the dimensions. Let d be the total number of attributes (dimensions) and $p_i.a_m$ denote the m-th dimension value of point p_i. Since we have assumed that "smaller is better", p_i is better than p_j in dimension a_m if $p_i.a_m < p_k.a_m$.



(a) 2-dimensional representation

id	price	weight
p_1	400	350
p_2	500	50
p_3	300	200
p_4	350	150
p_5	100	400
p_6	200	400
p_7	200	250
p_8	300	150
p_9	400	75
p_{10}	450	300

(b) PDA records

Fig. 1. Skyline example.

The PDA example is depicted in Figure 1. In Figure 1(a) each PDA is represented by a two-dimensional point (each dimension corresponds to an attribute). PDA records are shown in Figure 1(b). Points connected by the dashed line comprise the skyline of the set of points (PDAs). Any point falling on the right and top of the dashed line is dominated by at least on skyline point.

The black dots of Figure 1(a) represent the skyline. These points are not dominated by any other point. On the other hand, points p_6, p_1, p_3, p_4 are dominated by at least one other point. For example, PDA p_6 is dominated by p_5 because $p_5.weight = p_6.weight$, but $p_5.price < p_6.price$. Therefore, none of these points can be part of the skyline. The skyline is therefore consists of points p_5, p_7, p_8, p_9 and p_2.

The number of skyline points depends heavily on the dimensionality of the data set (number of attributes) and the data distribution. Therefore, according to these factors, in some cases the number of skyline points are too few, too many. Moreover, in some cases the skyline points may not be available. In our PDA example, it is expected that PDAs belonging to the skyline are too popular, and therefore purchase may not be possible due to lack of stock. In these cases, a ranking should be enforced to the multidimensional points, and interesting items should be presented to the user. This is exactly the topic of this paper. More specifically, we provide a meaningful way of ranking multidimensional points according to their domination power, towards presenting the items to the user

in an incremental manner. The proposed techniques constitute a way to facilitate mining and querying for dominance relationships.

The rest of the work is organized as follows. Section 2 presents the related work in the area, and describes our contributions shortly. Our proposal is given in Section 3, whereas Section 4 gives some representative experimental results. Finally, Section 5 concludes the work and briefly describes future work in the area.

2 Related Work

Skyline queries have received considerable attention recently, due to their aid in selecting the most preferred items, especially when the selection criteria are contradictory. Although the problem has been attacked in the past by a number of researchers [2, 8], only recently it has been tackled under a database point of view [3].

The current literature is rich in algorithms and organization schemes to facilitate skyline query processing. In [11] an efficient skyline query processing scheme has been proposed based on branch-and-bound, which utilizes the R-tree spatial access method [5, 1]. This method shows significant improvement over previously proposed methods. This scheme however, assumes that the skyline is computed over the whole set of attributes, which in many cases may not be meaningful.

Recently, alternative solutions have been proposed towards helping the user even further in selecting the most promising items. The output of a skyline query may contain too few, or too many answers, posing difficulties in selecting the best items. Towards alleviating this problem, k-dominant skylines have been proposed in [4]. According to this method, the definition of dominance is relaxed, in order to enable some points to be dominated, reducing the cardinality of the skyline point-set.

Another approach has been followed by [6] where the proposed technique searches for *thick* skylines. The thick skyline is composed of some skyline points and some additional points which are close to skyline points but not necessarily contained in the skyline set. This way, only points in these dense areas are proposed to the user.

In the same lines, an algorithm is proposed in [10] for selecting skyline points according to their domination capabilities. More specifically, the algorithm selects a subset of the skyline points aiming at maximizing the number of dominated points. However, this method is NP-hard for high-dimensional spaces, and therefore approximation algorithms are required towards fast computation.

In [9] the authors study dominance relationships between different data sets (e.g., products and customers). The authors propose DADA cube, an organization scheme to support a number of significant query types towards analyzing dominance relationships. DADA cube has been designed in accordance to data cubes utilized in data warehouses.

Finally, [7] study algorithmic techniques to compute approximately dominating representatives (ADRs). ADRs can be computed by appropriately post-

processing the result of a skyline query. The authors show that The authors show that the problem of minimizing the number of points returned can be solved in polynomial time in two dimensions, whereas it is NP-hard in higher dimensionalities. However, there is a polynomial-time logarithmic approximation.

Our research focuses on helping the user quantify the significance of items based on domination power. Therefore, the proposed scheme is mostly related to research contributions [6, 10, 9] because: (i) as in [6] items other than skyline items may be proposed, (ii) as in [10] our scheme returns items according their domination abilities, and (iii) as in [9] we investigate dominance relationships, However, our work differs significantly from the previous contributions. A transformation is applied which maps the items from the original d-dimensional space to a 2-dimensional one. Mining and ranking of items is performed to the new space (target space), avoiding the excess skyline cardinality that appears in high dimensions. Additionally, our proposed scheme handles cases where the skyline points are too few (e.g., in correlated data sets).

3 Domination Mining and Querying

Some skyline points are more significant than others. The quantification of the significance can be performed in a number of ways. For example, significance can be measured by means of the number of points dominated by a skyline point. A skyline point p_i is more significant than another skyline point p_j if the number of points dominated by p_i is larger than that dominated by p_j. This is the main concept of the scheme proposed in [10] for selecting the most representative skyline points.

However, in several cases we have to consider non-skyline points as well, which may also be of interest to the user. This becomes more clear by investigating Figure 1. It is evident that p_2 is a skyline point. However, although the weight of this PDA is very low, its price is high in relation to the other devices (in fact it has the largest price of all). It would be better, if PDAs p_3 and p_4 are also proposed to the user, since they better compromise the weight/price tradeoff. This example shows that it would be more appropriate to extend the concept of significance for all points in the data set, instead of considering skyline points only as in [10].

Another observation is that thick skylines that have been proposed in [6] do not solve our problem either, since a skyline point belonging to a dense area does not necessarily means that is more important than a skyline point lying in a sparse area. For example, consider a new PDA p with $p.price = 100$, and $p.weight = 50$. Evidently, although this point belongs to a sparse area, it is the most important skyline point since it has the smallest price and the smallest weight in comparison to all other PDAs. Moreover, we argue that sparse skyline points may be very important, since they are far from the competition.

3.1 Determining the Target Space

In the sequel we present three different transformation methods towards taking into account the *domination power* of each point. Points with large domination power should be proposed to the user before points with less domination power. According to our example, it is evident that the PDA represented by point p_8 has more domination power than that represented by p_5. In the sequel we will define domination power in a more formal way.

The first transformation (T_1), is the simplest to implement. Each multidimensional point p is transformed to a 2-dimensional point $T_1(p) = (da(p), 1 - DA(p)/a)$, where $da(p)$ is the area dominating p, whereas $DA(p)$ is the area dominated by p. Evidently, the values of $da(p)$ and $DA(p)$ are distribution independent, since they are computed based only on the location of p in the data space. This transformation may be used when only the domination *potential* of points is of interest.

$$T_1(p) = \left(\frac{da(p)}{a}, \quad 1 - \frac{DA(p)}{a} \right) \tag{1}$$

The second transformation (T_2), tries to capture the fact that the significance of a point p is increased when, the number of points dominating p decreases, and the number of points dominated by p increases. Based on this observation, each point p is transformed to $T_2(p) = (dps(p), 1 - DSP(p)/n)$, where $dps(p)$ is the number of points dominating p, and $DPS(p)$ is the number of points dominated by p. In contrast to the previous transformation, this one is distribution dependent, since only domination information is being taken into account, whereas the absolute location of p in the data space is completely ignored. This transformation should be used when the domination relationships among points needs to be investigated.

$$T_2(p) = \left(\frac{dps(p)}{n}, \quad 1 - \frac{DPS(p)}{n} \right) \tag{2}$$

The next transformation (T_3) tries to combine the previous two, by proposing a hybrid scheme. Each point p is transformed to $T_3(p) = (da(p)/a, DPS(p)/n)$, where again $da(p)$ is the area that dominates p and $DPS(p)$ is the number of points dominated by p.

$$T_3(p) = \left(\frac{da(p)}{a}, \quad 1 - \frac{DPS(p)}{n} \right) \tag{3}$$

Note that, all methods transform the d-dimensional points of P to 2-d, in order to reduce the cardinality of skyline sets in the target space, and provide a more convenient space for mining and querying dominance relationships.

Next we define two important measures, the *domination power* of a point and the *domination distance* between two points. The first one aims to quantify the significance of each point, whereas the second one quantifies the difference between two points in terms of their distance to the target space. Note that,

since the target space is defined in the unit square, the measures assume values in the interval [0,2].

Definition 1 (*domination power*)
The domination power $P(p)$ of a point p is the cityblock distance of its image p' point to the origin of the target space. More formally:

$$P(p) = p'.x + p'.y \tag{4}$$

Definition 2 (*domination distance*)
The domination distance $D(p, q)$ between two points p and q, is the cityblock distance of their images p' and q' in the target space. More formally:

$$D(p, q) = |p'.x - q'.x| + |p'.y - q'.y| \tag{5}$$

3.2 Creating and Maintaining the Target Space

There are two important issues to be resolved regarding the generation of the target space: (i) target space generation and (ii) target space maintenance. Among the three different transformations described previously, the simplest to generate and maintain is the first one (T_1), since the image p' of a point p depends only on the location of the point itself in the original space. On the other hand, the most expensive transformation is the second (T_2), because the image of a point depends on the locations of the other points. Moreover, the insertion/deletion of a point may cause changes in the images of other points. In the sequel, we describe the generation of the target space according to T_2, since the process for T_1 and T_3 are similar and less demanding regarding computations.

Without loss of generality, we assume the existence of an R-tree access method [5, 1] to organize the points in the original space. Other hierarchical access methods can be utilized equally well. The main issue regarding T_2 is that for every point p we need to calculate the values for $dps(p)$ and $DPS(p)$. A naive approach to follow is for each point to apply two range queries in the R-tree, one for the region $da(p)$ and one for the region $DA(p)$. Evidently, $da(p)$ contains all points that dominate p, whereas $DA(p)$ contains all points dominated by p. By using an arbitrary examination order, it is expected that significant I/O overhead will take place. Instead, we use an approach which better exploits locality of references. This approach is based on the Hilbert space filling curve. In the first phase the value $dps(p)$ is calculated for every point p. In the second phase the $DPS(p)$ value is calculated. Both phases respect the Hilbert order in visiting the points. Internal nodes are prioritized based on the Hilbert value of their MBR centroid.

A significant improvement can be applied to both algorithms towards reducing the number of I/O requests. This involves the utilization of batched processing. When a leaf node is reached, instead of executing multiple range queries for each point in the leaf, we execute only one in each phase. Our intuition regarding the efficiency of this variation is verified by the computational and I/O

Algorithm CreateTargetSpace (*Rtree*)
Rtree: the R-tree organizing the original space
minheap: the priority queue of tree entries prioritized by increasing Hilbert values
maxheap: the priority queue of tree entries prioritized by decreasing Hilbert values

```
1.  treeNode = root of the Rtree
2.  place entries of treeNode into minHeap
3.  while (minHeap not empty)
4.      heapEntry = top of minHeap
5.      if (heapEntry points to a leaf node)
6.          leafNode = the leaf pointed by heapentry
7.          execute range query using upper-right corner of leaf MBR
8.          update dominance information
9.      else
10.         internalNode = node pointed by heapEntry
11.         calculate Hilbert values and insert all entries of internalNode into minHeap
12.     end if
13. end while
14. treeNode = root of the Rtree
15. place entries of treeNode into maxHeap
16. while (maxHeap not empty)
17.     heapEntry = top of maxHeap
18.     if (heapEntry points to a leaf node)
19.         leafNode = the leaf pointed by heapentry
20.         execute range query using lower-left corner of leaf MBR
21.         update dominance information
22.     else
23.         internalNode = node pointed by heapEntry
24.         calculate Hilbert values and insert all entries of internalNode into maxHeap
25.     end if
26. end while
```

Fig. 2. Outline of CreateTargetSpace algorithm.

time required to build the target space as shown in the experimental results. The improved algorithm can be used both when all data are memory-resident or disk-based. The outline of the final algorithm is illustrated in Figure 2.

Since the suggested methodology is focused on analysis and mining, keeping the target space up-to-date upon every insertion and deletion is not our primary concern. However, if such a synchronization action is required this is easily achieved for T_1. More specifically, if a new point p is inserted in the original space, its image is directly calculated by using Equation 1. For T_2, two range queries must be executed in the R-tree, one for $da(p)$ and one for $DA(p)$ to produce the image of p. Moreover, a complete synchronization requires additional updates because: (i) the total number of points has been changed, (ii) for each

point p_x that dominated p the value $DPS(p_x)$ must be updated, and (iii) for each point p_x dominated by p the value $dps(p_x)$ must be updated. The case of T_3 is slightly more efficient, since only one range query is required, but several updates of images may be required. To avoid increased maintenance costs when T_2 or T_3 is being used, updates in the target space may be performed periodically. T_3 is the best compromise between maintenance efficiency and perception of domination information.

3.3 Utilizing the Target Space

Domination power and domination distance allow for a number of important querying and mining operations to be supported. The first important operation is the determination of the "best" points, by executing a skyline computation algorithm in the target space, such as the one proposed in [11]. If the user is not satisfied by the answers, the next layer of skylines can be computed, by ignoring the previously returned points. This process may continue according to the user's request. Note that, since the dimensionality of the target space is 2 and the distribution of the images show a high degree of correlation, the number of skyline points is not expected to be large. By supporting multiple skyline layers we alleviate the problem of the small skyline cardinality.

Another important operation is the top-k query, which given an integer k, returns the k best points, according to domination power. Since the target space is already computed, such a top-k query is immediately supported by applying a branch-and-bound search algorithm with a priority queue.

The third operation is related to nearest-neighbors in the target space. Given a point q, a k-nearest-neighbor (k-NN) query returns the k points closest to q', according to the domination distance measure. This way, one can determine which points "threaten" q with respect to domination distance. A similar query can be defined by given the query point q and a domination distance threshold e. In this case, we are interested in determining which point images p' are within distance e from q' (range query). Evidently, if two images are close in the target space, it is not necessary that the original points will be close to each other in the original space, and in fact this is one of the most important properties of the target space.

Apart from using the target space in a online fashion, data mining techniques can be applied to discover useful knowledge and support the online part in a number of ways. A useful operation is to detect the outliers in the target space. Points that are quite "isolated" from the competition are the candidates for being outliers. Regarding domination relationships, an outlier may be significant, and therefore it is important to give the chance to the user to check it. However, not all outliers need to be presented, but only the most significant ones with respect to domination power. It is up to the user to select all or the most significant ones for further inspection.

A data mining task related to outlier detection is clustering. In fact, many clustering algorithms support outlier detection (e.g., DBSCAN). The application of clustering in the target space will partition the images of points to a number

data set	**NAIVE** (IO/CPU)	**2P** (IO/CPU)	**2P-BATCH** (IO/CPU)
IND	1199297/117.73	973156/115.78	25719/4.13
ANTI	412939/32.80	137797/31.34	7873/1.53
CORR	1644380/165.68	1430990/162.70	33747/5.40
GAUSS	1253231/116.45	1030278/113.87	25849/4.00

Table 1. Cost for target space generation (10,000 points in 2-d).

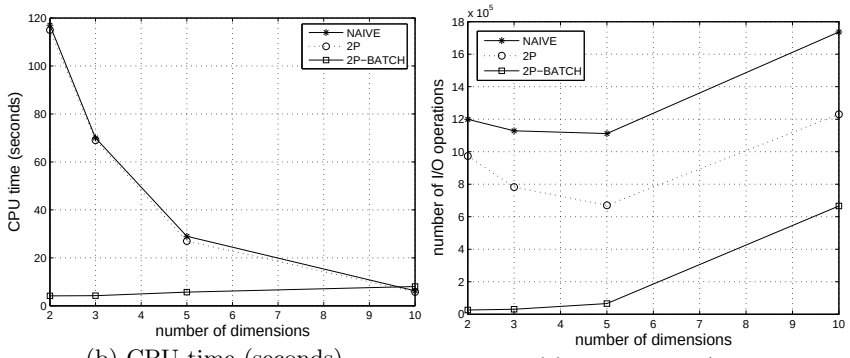
of groups. Each group is expected to contain "similar" points regarding their domination characteristics. Additionally, clustering can be utilized towards partitioning the target space and then use the partitions to suggest items to users. This way, instead of proposing single points, a whole cluster of points is suggested to the user. Cluster selection may be applied by using the domination power of the cluster centroid. Therefore, the cluster whose centroid is closer to the origin of the target space is proposed first.

4 Experiments and Results

In the sequel, we present some representative results. All experiments have been conducted on a Pentium IV machine at 3GHz, with 1GB RAM running Windows XP Prof. We have used four synthetically generated data sets with different distributions: (i) anticorrelated (ANTI), (ii) indepednent (IND), (iii) gaussian (GAUSS) and (iv) correlated (CORR). The data sets have been generated by using the generation process reported in [3].

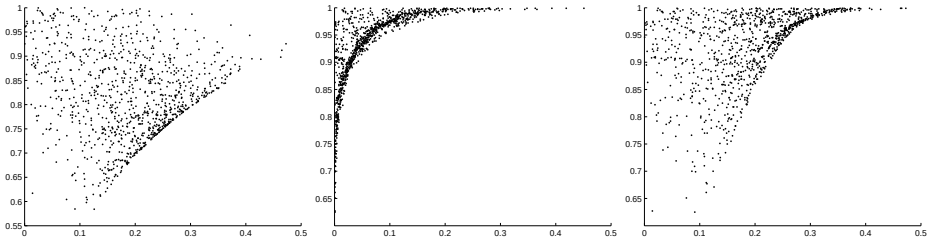
Initially, we present some representative results regarding the efficiency of target space generation, by comparing the naive method (algorithm NAIVE), with the two-phase Hilbert-based method (algorithm 2P) and the improved batch-enabled alternative (algorithm 2P-BATCH). Table 1 illustrates some representative performance comparison results for 2-dimensional data sets. CPU cost is measured in seconds, whereas the I/O cost corresponds to the number of disk accesses performed due to buffer misses. Figure 3 shows the scalability of the algorithms for different dimensionalities for the IND data set. An LRU-based buffer has been used which maintains the 25% of the total storage requirements. Each experiment has been executed on a data set containing 10,000 multidimensional points, whereas the page size of the R-tree has been set to 2 KBytes. It is evident, that 2P-BATCH has the best overall performance, both in terms of computational time and I/O cost.

In the sequel, we investigate how clustering algorithms can aid towards exploration of the target space. Figure 4 illustrates the distribution of the point images in the target space for all available transformation methods (T_1, T_2, and T_3), for the ANTI data set containing 1,000 points. We have chosen this set because the number of skyline points in the original space is quite large, and therefore there are difficulties in proposing some representative points to the user.



(b) CPU time (seconds) (c) number of I/Os

Fig. 3. Comparison of methods for different dimensions (10,000 IND points).

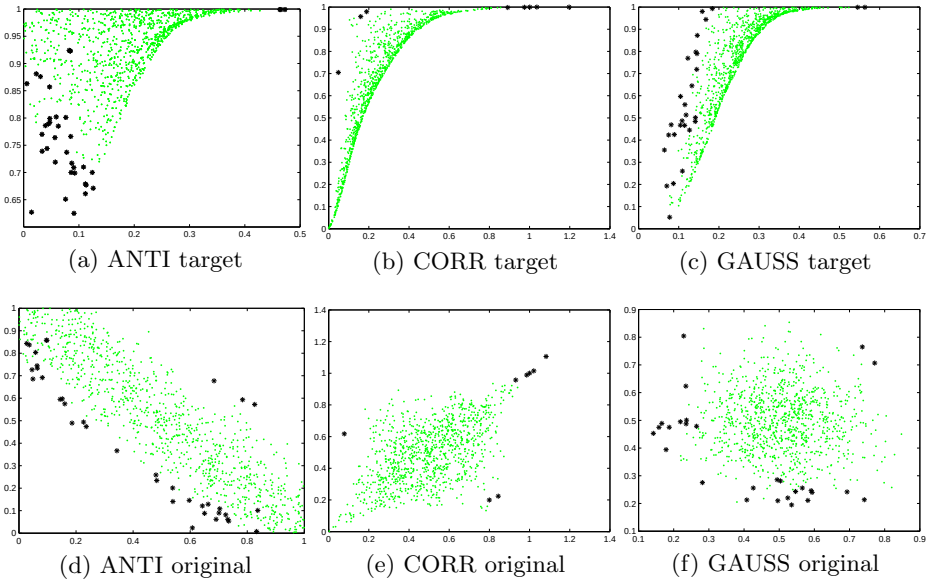


(a) T_1 (b) T_2 (c) T_3

Fig. 4. Target spaces for ANTI data set (1,000 points in 2 dimensions).

By inspecting Figure 4 we can see that the distribution of points in the target space is completely different than that in the original one. The transformation performed offers an opportunity to select the "best" points much more easily. This can be performed by other using the queries described in Section 3.3 or by performing a partitioning of the target space by means of clustering. To illustrate this issue, we have applied two fundamental clustering algorithms: (i) DBSCAN and (ii) K-means. By using DBSCAN we aim at discovering outliers whereas the use of K-means aims at space partitioning.

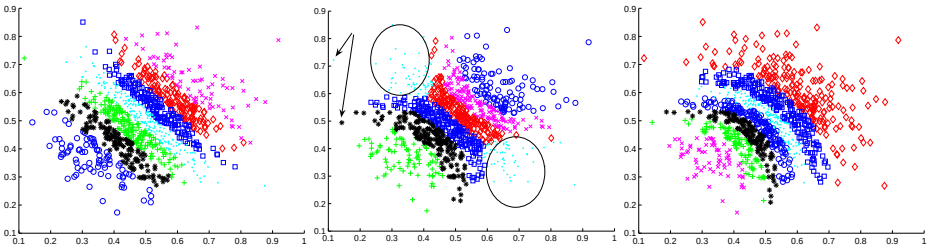
Figure 5 shows the outliers determined for ANTI, CORR and GAUSS by using the transformation T_3. In addition to the target space ((a), (b) and (c)), the position of the each outlier is also given in the original space ((d), (e) and (f)). As we have already mention, outliers may contain important information since they refer to points that are away from the competition, and therefore these points may require further investigation. In fact, as it is shown in Figures 5(a), (c), (e) and (f), some of the outliers are really important because their domination power is significant. It is up to the user to request the outliers towards further study.



(a) ANTI target (b) CORR target (c) GAUSS target

(d) ANTI original (e) CORR original (f) GAUSS original

Fig. 5. Outlier detection for ANTI, CORR and GAUSS data sets (1,000 points in 2 dimensions).



(a) T_1 (b) T_2 (c) T_3

Fig. 6. Data partitioning with K-means for $K = 7$ (1,000 points, GAUSS data set).

Let us see now how K-means clustering can help. By applying K-means on the target space, the points are partitioned into groups. We apply clustering based on the cityblock distance of the points. Figure 6 gives the clustering results for GAUSS data set, for all transformations. Only the original space partitioning is shown. Clustering has been performed for 7 clusters. However, other values can be used as well. By inspecting Figure 6 we observe that clusters in T_1 are more crisp, in the sense that points are grouped together with respect to the diagonal they belong. This was expected since T_1 is based on the coordinates of each point only, and the image of each point is generated without taking into account the other points locations. On the other hand, T_2 and T_3 better express the notion

of domination power. In both transformations, the shape of the clusters is quite different than that produced with T_1. For example, in T_2, the points indicated by arrows although they belong to the skyline, they receive a lower rank because the number of points they dominate is zero or very small. Moreover, notice that points in the same cluster does not necessarily share nearby positions in the original space. The points enclosed by the two ellipses belong to the same cluster. T_3 produces similar results to those of T_2, but the emphasis is given on the number of points that dominate a particular point and the area that this point dominates (not the number of points dominated).

5 Concluding Remarks

Although the literature is rich in studying efficient algorithms for domination computation, only recently the issue of analyzing domination relationships has received some attention. Towards this direction, we have focused on applying well-established data mining techniques and query operations aiming at a more convenient scheme for determining the "best" items and extracting useful knowledge from domination relationships. Future research in the area may focus on: (i) mining domination relationships among different types of items, (ii) measuring the domination power in a set of items instead of individual ones, (iii) the evaluation of different metrics for domination power and distance and (iv) enhanced visualization methods for dominance relationships.

References

1. N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. "The R*-tree: an Efficient and Robust Access Method for Points and Rectangles", *Proceedings of the ACM SIGMOD Conference*, pp. 322-331, Atlantic City, NJ, May 1990.
2. J.L. Bentley, K.L. Clarkson, and D.B. Levine. "Fast Linear Expected-Time Algorithms for Computing Maxima and Convex Hulls", *Symposium on Discrete Algorithms*, pp.179-187, 1990.
3. S. Borzsonyi, D. Kossmann, and K. Stocker. "The Skyline Operator", *Proceedings of the International Conference on Data Engineering*, pp.421-430, 2001.
4. C.-Y. Chan, H.V. Jagadish, K.-L. Tan, A.K.H. Tung, and Z. Zhang. "Finding k-Dominant Skylines in High Dimensional Space", *Proceedings of the ACM SIGMOD Conference*, pp.503-514, 2006.
5. A. Guttman. "R-trees: a dynamic index structure for spatial searching", *Proceedings of the ACM SIGMOD Conference*, pp.47-57, 1984.
6. W. Jin, J. Han, and M. Ester. "Mining Thick Skylines over Large Databases", *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2004.
7. V. Koltun, C.H. Papadimitriou. "Approximately Dominating Representatives", *Theoretical Computer Science*, Vol.371, No.3, pp.148-154, 2007.
8. H.T. Kung. "On Finding the Maxima of a Set of Vectors", *Journal of the ACM*, Vol.22, No.4, 1975.
9. C. Li, B.C. Ooi, A.K.H. Tung, S. Wang. "DADA: A Data Cube for Dominant Relationship Analysis", *Proceedings of the ACM SIGMOD Conference*, pp.659-670, 2006.
10. X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang. "Selecting Stars: The k Most Representative Skyline Operator", *Proceedings of the 23rd International Confernce on Data Engineering*, 2007.
11. D. Papadias, Y. Tao, G. Fu, and B. Seeger. "Progressive Skyline Computation in Database Systems", *ACM Transactions on Database Systems*, Vol.30, No.1, 2005.