A Pareto-Dominant Clustering Approach for Pareto-Frontiers

Johannes Kastner
Institute for Computer Science
University of Augsburg
86135 Augsburg, Germany
johannes.kastner@informatik.uni-augsburg.de

Markus Endres
Institute for Computer Science
University of Augsburg
86135 Augsburg, Germany
markus.endres@informatik.uni-augsburg.de

Werner Kießling
Institute for Computer Science
University of Augsburg
86135 Augsburg, Germany
werner.kiessling@informatik.uni-augsburg.de

ABSTRACT

Managing large and confusing sets of increasing data is a well-known problem in Data Mining. Since compromises in many use cases like Recommender Systems or preference-based applications are becoming more and more usual, it is very useful to cluster sets of promising results in order to get an overview and present them properly. In this paper we present the Pareto-dominance as a very suitable and promising approach to cluster objects over better than relationships. In order to meet someones desires, one can tip the balance of the final results to the more favored dimension if no decision for allocating objects is possible.

CCS Concepts

•**Information systems** → **Clustering;**

Keywords

Clustering, Pareto-Dominance, k-means

1. INTRODUCTION

Suggestions in Recommender Systems sometimes are confusing, because there are often too many results presented to the user. Clustering these results is a very promising opportunity to present less but representative results to the user. For example, in a Recommender System which addresses people with a similar taste of music, the regional distance to users which represents possible matches, similar to other applications, e.g., Tinder[1], is considered almost as important as the music-matching score.

Assuming user Bob is searching for users, who have a high music-matching score and live in the neighborhood. There are users which are better than other users regarding one dimension, e.g., the music-matching score, but are worse at the same time than other users regarding the other dimension, e.g., the distance to Bob. In Figure 1 user P_1 has a very close distance to Bob, and dominates due to that all other users. But P_1 is dominated by all other users w.r.t. the very low music-matching score to Bob at the same time. This exposition, where only non-dominated users are shown is called Pareto-frontier or Skyline.

Now, the aim is to organize a large set of objects in a Pareto-frontier clearly. One method is to compress this set and express it with a smaller appropriate set of representatives. Another approach is to mask out undesirable results, e.g., users like P_9 and P_{10} who have a higher music-matching score, but unfortunately a very high distance. These two attempts can only be reached with some kind of clustering.

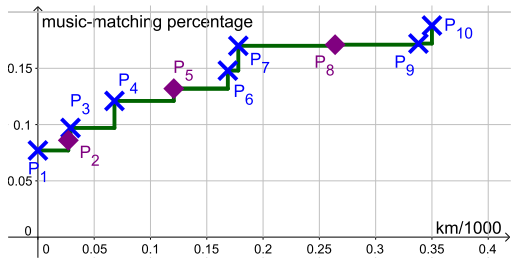

Figure 1: Pareto-frontier of users with preferably best music-matching and preferably closest distance.

However we have two dimensions with different domains, e.g., distance in kilometers and music-matching as a score value. Therefore it is hardly possible to achieve an useful outcome without great afford, because the dimensions should be set into relation to each other. In our example it is very circumstantial to set this wide range of distances in relation to only a small range of music-matching scores. Since every user has diverse requirements in such a Recommender System, it is by far not sufficient to use the basic k-means clustering algorithm along with the typical Euclidean distance.

The approach presented in this paper uses the Pareto-dominance in order to cluster over better than relationships unlike mapped Euclidean distances, which have to be adjusted inconveniently for each use case.

The rest of the paper is organized as follows: We present related work in Section 2 and explain the basic background

[1]http://www.gotinder.com/

2017, Copyright is with the authors. Published in the Workshop Proceedings of the EDBT/ICDT 2017 Joint Conference (March 21, 2017, Venice, Italy) on CEUR-WS.org (ISSN 1613-0073). Distribution of this paper is permitted under the terms of the Creative Commons license CC-by-nc-nd 4.0

knowledge in Section 3. Our Pareto-dominance clustering approach is described in Section 4. After that, we discuss experiments, where runtime, number of iterations and quality of the final clusters are considered and compared to the basic k-means clustering algorithm in Section 5. Finally Section 6 summarizes results and gives an outlook on future work.

2. RELATED WORK

A very early approach considering a Pareto-efficient clustering was published in [3] where more then one criterion for clustering was consulted. This paper presents on the one hand a modified relocation algorithm, and on the other hand a modified agglomerative algorithm. These approaches aim finding a Pareto-dominant clustering that dominates all other clusterings, while the approach presented in our paper uses Pareto-dominance in order to allocate an object to a specific cluster.

In [4] a k-means clustering-based technique was published, where a so-called SkyClustering method is working within a Skyline-computation in SQL on a relational database, in order to compress a large Pareto-optimal set of objects to explore the diversity of a Skyline.

In order to prevent a large set of Pareto-optimal objects in highdimensional space in [2] only a few dimensions k are considered for the Skyline computation. This approach attaches weight to less, but maybe more important dimensions on objects.

Another approach to handle with Skylines was presented in [7], where supervised alternative clusterings are introduced. The main focus of this paper is to find clusterings of good quality starting from given negative clusterings which should be as different as possible and at the same time a Pareto-optimal solution. If the solution is not satisfying, the Pareto-frontier will be reclustered with the tradition k-means clustering, unlike the Pareto-dominant clustering presented in our paper.

3. BACKGROUND

Before explaining the Pareto-dominance clustering framework, we describe some important concepts. One basic point is the preference framework in order to shape desire-based queries. The other basic point mentioned in this section is the k-means clustering algorithm.

3.1 Preferences

Preferences represent wishes which should be fulfilled. In database systems a preference $P = (A, <_P)$ is modeled as a strict partial order on the domain of A, $dom(A)$. The term $x <_P y$ can be described as "I like y more than x". The most important preference is the well-known Pareto preference which models equal importance.

DEFINITION 1 (PARETO PREFERENCE).
A Pareto preference $P := P_1 \otimes P_2 = (A_1 \times A_2, <_P)$ *with preferences* $P_i = (A_i, <_{P_i})$ *and tuples* $x = (x_1, x_2), y = (y_1, y_2) \in dom(A_1) \times dom(A_2)$ *is defined as follows:*

$$(x_1, x_2) <_P (y_1, y_2) \Leftrightarrow$$
$$(x_1 <_{P_1} y_1 \wedge (x_2 <_{P_2} y_2 \vee x_2 = y_2)) \vee$$
$$(x_2 <_{P_2} y_2 \wedge (x_1 <_{P_1} y_1 \vee x_1 = y_1))$$

In order to construct Pareto preferences there are several *preference constructors* published in [6], e.g., BETWEEN for values in an interval $[low, up]$ or LOWEST and HIGHEST for *minimum* and *maximum* . These preferences are used to build Pareto preferences in an intuitive way. Pareto preference queries coincide with the traditional Skyline queries [1], If we restrict the attention to LOWEST/HIGHEST as input preferences.

3.2 k-means Clustering

The k-means clustering algorithm is the point of origin for the clustering approach presented in our paper. Therefore the base k-means clustering is briefly described here, as it is presented in [5].

Given a set $X = \{x_i \mid i = 1, ..., n\}$ of d-dimensional points $x_i = (x_{i1}, ..., x_{id})$ of size n and a set of k Clusters $C = \{c_j \mid j = 1, ..., k\}$ the algorithm works as follows:

ALGORITHM 1 (K-MEANS-CLUSTERING).

1. *Choose an initial partition for the centroids of k clusters and repeat the following steps 2 and 3 until two succeeding clusterings are* stable. *Two clusterings are stable, if the points of the particular clusters are equal.*
2. *Allocate each point to the closest cluster based on the Euclidean norm to the cluster-centroids.*
$$d(x_1, x_2) = || x_1 - x_2 ||_2 = \sqrt{\sum_{i=1}^{d} (x_{1i} - x_{2i})^2}$$
3. *Recalculate the cluster-centroids of each cluster by averaging the contained points.*

EXAMPLE 1 (K-MEANS EUCLIDEAN NORM).
Consider the users in Figure 1, shown as points on a Pareto-frontier. Now we want to get three clusters. First, initialize the centroids of the three desired clusters randomly with P_2, P_5 *and* P_8 *for* C_1, C_2 *and* C_3.
Now for each user $P_1, ..., P_{10}$ *the Euclidean distance to all three centroids, determined in step 1, are calculated.* P_1, P_3 *and* P_4 *are allocated to* C_1, *while the users* P_6 *and* P_7 *are assigned to* C_2. *Finally* C_3 *receives the users* P_9 *and* P_{10}.
In the last step of each iteration, the new centroids W_1, W_2 *and* W_3 *of each cluster are averaged based on the points contained in the clusters. The result is shown in Figure 2.*

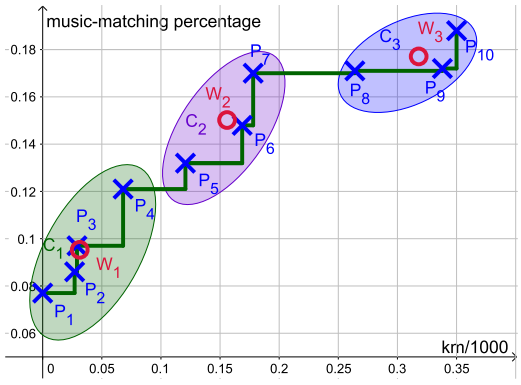

Figure 2: k-means clustering of users with the Euclidean norm after the first iteration.

4. PARETO-DOMINANCE FRAMEWORK

In this section we describe the Pareto-dominance framework in detail. While a Pareto preference is used to determine the importance of preferences, the *Pareto-dominance* in our approach is used to *allocate an object to the possibly best cluster*, which is not dominated by other clusters w.r.t. the distances of the individual objects, by using the Euclidean norm for one-dimensional distances. Moreover the Pareto-dominance can additionally be used to find the centroids closest point on the Pareto-frontier as new centroid in each cluster.

4.1 Cluster Allocation

Consider the Pareto-frontier in Figure 1, presenting users w.r.t. a possibly high music-matching score and a possibly close distance. We want to get three promising clusters C_1, C_2 and C_3. In this case assume the cluster-centroids are represented by the points P_2, P_5 and P_8 (Step 1 in Algorithm 1). After that, for each point $P_1, ..., P_{10}$ the particular distances of the x- and the y-dimension to the cluster centroids are calculated as it can be seen in Table 1. In our approach this replaces step 2 in Algorithm 1. Furthermore the y-dimension, which represents the music-matching score is chosen as more important than the x-dimension at the appearance of Pareto-optimal cluster-centroids. Thus a one-dimensional clustering is realized. The allocation of each user to a cluster can be seen in Figure 3.

	$d_{x_{C_1}}$	$d_{y_{C_1}}$	$d_{x_{C_2}}$	$d_{y_{C_2}}$	$d_{x_{C_3}}$	$d_{y_{C_3}}$
P_1	**27.44**	**0.01**	120.72	0.06	264.60	0.09
P_2	**0.00**	**0.00**	93.27	0.05	237.16	0.09
P_3	**1.66**	**0.01**	91.61	0.04	235.50	0.07
P_4	**41.27**	0.04	52.00	**0.01**	195.89	0.05
P_5	93.27	0.05	**0.00**	**0.00**	143.89	0.04
P_6	141.27	0.06	**48.00**	**0.016**	95.89	0.023
P_7	150.88	0.09	**57.61**	0.04	86.28	**0.00**
P_8	237.16	0.09	143.89	0.04	**0.00**	**0.00**
P_9	311.32	0.09	218.05	0.04	**74.16**	**0.00**
P_{10}	323.08	0.10	229.81	0.06	**85.92**	**0.02**

Table 1: Distances of each user to each cluster centroid. *$C_1 := P_2, C_2 := P_5, C_3 := P_8$,* ***x-dim. = distance, y-dim. = music matching score.***

Now we shortly explain the allocation of the given users to the clusters:

- The users P_1 and P_3 are assigned to cluster C_1, because the centroid of C_1 dominates the other two centroids regarding the distances in both dimensions.
- P_4 has *2 Pareto-optima*, because of the closer distance to C_1 regarding the x-dimension and to C_2 regarding the y-dimension. Hence C_1 and C_2 are Pareto-optimal w.r.t. to the x- and y-dimension. Now the one-dimensional clustering tips the balance to C_2.
- P_6 is allocated to Cluster C_2, because of the closer distances to the centroid of C_2 in both dimensions.
- P_7 is closer to C_2 concerning the x-dimension, but has a smaller distance to the centroid of C_3 regarding the y-dimension. This ensures that P_7 is allocated to C_3.
- P_9 and P_{10} are allocated to cluster C_3, because of the existence of only one Pareto-dominant cluster centroid namely C_3.

Figure 4 shows a snippet of the Pareto-frontier of Figure 2 respectively Figure 3. While P_4 is assigned to C_1 regarding

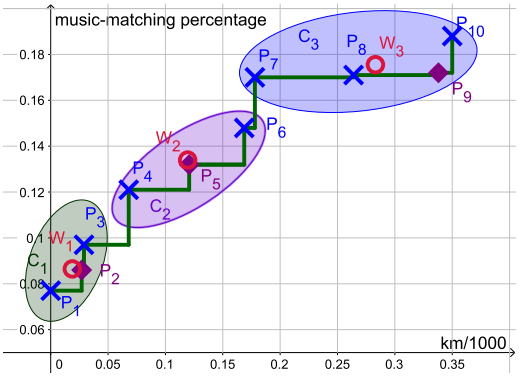

Figure 3: Clustering of users of Figure 1 with the Pareto-dominance after the first iteration.

the smaller Euclidean distance of 41.27 unlike a distance of 52.00 to C_2, the Pareto-dominance approach shows its versatility. The user Bob now can influence on the clustering by choosing one dimension as the more important at the appearance of Pareto-optima. If he chooses the x-dimension, as more important P_4 will be assigned to C_1. This ensures that each user will be allocated to one and only one cluster, to avoid overlapping and imprecise clusters. This example shows, that a Pareto-dominant clustering combined with a one-dimensional clustering at the appearance of Pareto-optima tends to a hierarchical clustering. Users with similar scoring values w.r.t. the music matching score are clustered together, unlike in the basic k-means clustering. In particular cluster C_1 and C_2 contain users with very similar music-matching scores, where the range between the two boundary points is very small unlike the k-means clustering approach. So if P_7 is allocated to C_2 and P_4 to C_1 the users contained in the clusters are not as similar as in our approach.

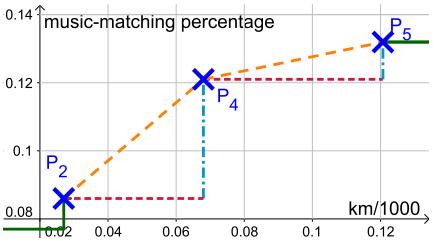

Figure 4: Camparison of Pareto-dominance and Euclidean Distance. *$C_1 := P_2, C_2 := P_5$.*

4.2 Cluster Centroids

After the allocation of each user to a cluster, the cluster-centroids are recalculated regarding the contained users. For each cluster all values for the x- and y-dimension are averaged, which can be seen in Figure 3 as W_1, W_2 and W_3.

After that for each cluster-centroid W_1, W_2 and W_3 the closest Pareto-dominant user in each cluster can be optionally selected as the new cluster-centroid for the next iteration. In order to find these users, the particular distances regarding the two dimensions are calculated, which can be seen in Table 2. Figure 3 shows the calculated centroids before assigning and the new centroids aswell.

- P_2 is the new cluster centroid of C_1, because of the closest distance of each x- and y-dimension to W_1.
- For cluster C_2 P_5 is allocated as new centroid because of the closer distances in both dimensions, too.
- P_8 and P_9 both are Pareto-optima for the allocation of the cluster centroid of C_3, because P_9 is closer to W_3 regarding the y-dimension and P_8 regarding the x-dimension. The one-dimensional clustering determined by Bob tips the balance to P_9 as new cluster centroid.

	$d_{x_{W_1}}$	$d_{y_{W_1}}$	$d_{x_{W_2}}$	$d_{y_{W_2}}$	$d_{x_{W_3}}$	$d_{y_{W_3}}$
P_1	18.848	0.010	—	—	—	—
P_2	**8.595**	**0.001**	—	—	—	—
P_3	10.253	0.010	—	—	—	—
P_4	—	—	50.667	0.012	—	—
P_5	—	—	**1.331**	**0.002**	—	—
P_6	—	—	49.331	0.014	—	—
P_7	—	—	—	—	104.730	0.004
P_8	—	—	—	—	**18.451**	0.004
P_9	—	—	—	—	55.709	**0.003**
P_{10}	—	—	—	—	67.4726	0.0120

Table 2: Particular distances of recalculated cluster-centroids W_1, W_2, W_3 to the points in each cluster.

5. EXPERIMENTS

In this section we describe the implemented frameworks for Java and PG/PL-SQL briefly and present results of experiments regarding runtime, number of iterations and quality of the clustering approaches compared to the basic k-means approach.

5.1 Benchmark Settings

The first implementation of our Pareto-dominant clustering was realized as a Java program with a complexity of $\mathcal{O}(n \cdot c \cdot m)$ where n is the number of points that should be clustered in c clusters in m iterations. We also implemented a database internal approach based on PG/PL-SQL in PostgreSQL[2], because we want to show the benefits of clustering in relational databases.

For both approaches we varied the number of points and the number of desired clusters. To compare the runtimes, we created several synthetic anticorrelated sets of two-dimensional Pareto-optimal points. In order to gain averaged reliable data, clusterings were performed in test rows with varying numbers of repeats w.r.t. the number of points.

5.2 Benchmarks

The benchmarks of the Java implementation in Figure 5 show that the approach using the Pareto-dominance are mostly similar regarding the runtime compared to the basic k-means approach using the Euclidean distance. For constant numbers of clusters and growing number of points the runtime of the averaged clustering is growing for both approaches. Whereas for constant numbers of points and growing clusterings there are some aberations at $k = 7$ for 15000 for both approaches. Especially if points at the border of the cluster switch between two clusters, the runtime is growing. All in all the clustering approach using the Pareto-dominance is nearly efficient as using the Euclidean distance.

[2]https://www.postgresql.org/

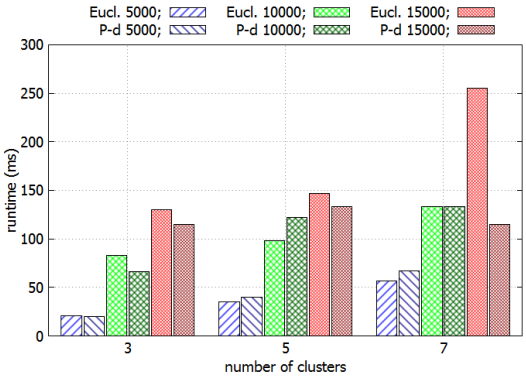

Figure 5: Java benchmarks of the basic k-means clustering algorithm with Euclidean distance(Eucl) and Pareto-dominance(P-d).

The benchmarks for the database approach in Figure 6 shows different behavior. The Pareto-dominance approach is of factor 2 slower than the approach using the Euclidean distance, because of the use of expensive database operations like Joins or Group by operations, which are not needed using the Euclidean distance. Especially the tests with $k = 5, 7$ and sets with 15000 points show aberations for the Pareto-dominance. But all in all for growing number of points and growing number of clusters the runtime is growing, too. Finally our approach is slower than the basic k-means clustering. But the effort of normalizing the users w.r.t. the two dimensions should be considered as a time-consuming process in each use case.

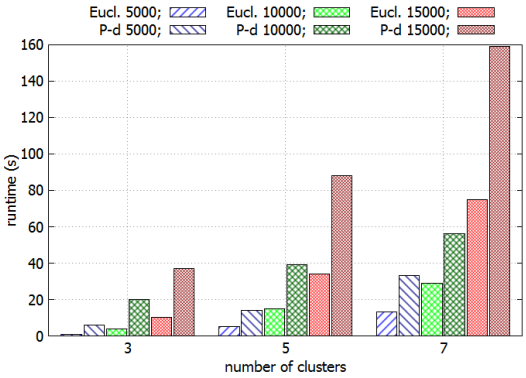

Figure 6: PostgreSQL benchmarks of the basic k-means clustering algorithm with Euclidean distance(Eucl) and Pareto-dominance(P-d).

The number of iterations w.r.t. the number of desired clusters and the number of points can be seen in Figure 7. For both frameworks, the number of iterations is similar. For growing number of desired clusters, the number of iterations is growing for both approaches aswell, except for the sets of $k = 5, 7$ with 15000 points using the Pareto-dominance. In contrary to our expectations for this experiment the number of points in the sets has no influence on the number of necessary iterations to achieve a stable clustering, i.e. two succeeding clustering-iterations are equal.

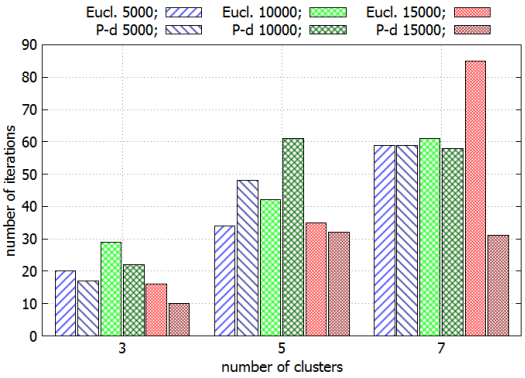

Figure 7: Comparison of Euclidean distance (Eucl.) and Pareto-dominance(P-d) regarding iterations.

5.3 Quality

In this subsection we want to compare the quality of clusters between the basic k-means clustering algorithm and our approach in order to show that the stable clusters are quite similar and not completely different and thus useful for clustering objects of Pareto-frontiers.

Considering the example from Figure 2 and 3, which show both the most occurent stable clusterings after the second iteration. We performed a test-series based on the k-means clustering, which were compared to a testrow with 100 clusterings of our approach. In order to compare these Clusters we use Precision and Recall. Precision represents all desired and delivered objects (correct alarms) in relation to all delivered objects (correct alarms & false alarms), whereas Recall represents the desired and delivered objects (correct alarms) in relation to all desired objects (correct alarms & false dismissals).

	C_1	C_2	C_3
false alarms (fa)	$\{\}$	$\{P_4\}$	$\{P_7\}$
false dismissals (fd)	$\{P_4\}$	$\{P_7\}$	$\{\}$
correct alarms (ca)	$\{P_1, P_2, P_3\}$	$\{P_5, P_6\}$	$\{P_8, P_9, P_{10}\}$
Precision: $\frac{ca}{ca+fa}$	1	0.66	0.75
Recall: $\frac{ca}{ca+fd}$	0.75	0.66	1

Table 3: Precision-Recall model in detail for the most occurent clustering in Figure 3.

Table 3 shows the values for Precision and Recall for the most occurent Pareto-dominant clustering on the base of the k-means clusters. Precision and Recall both show for all clusters very high values with only two user, which switch between the clusters. Thus the quality of the Pareto-dominant clustering is in this case mostly als high as possible, but not equal to the k-means clustering with Euclidean distance.

Clustering	freq.	P_{C_1}	P_{C_2}	P_{C_3}	R_{C_1}	R_{C_2}	R_{C_3}
1	78	1.00	0.67	0.75	0.75	0.67	1.00
2	7	1.00	0.60	1.00	1.00	1.00	0.33
3	15	0.80	0.60	1.00	1.00	0.75	0.33
Averaged	100	0.97	0.65	0.81	0.81	0.70	0.85

Table 4: Averaged Precision-Recall model for 100 testrows and the three occurent clusterings of users in Figure 1.

In Table 4 the test series show the Precision and Recall scores for two more clusterings and the frequency in the test series of 100 clusterings, which get averaged. Finally in this case there is one very present clustering (1), which is a very satisfying clustering as mentioned in Section 4. Both other clusterings have very high scores for Precision. The scores for Recall especially for the third cluster for the clustering 2 and 3 are very low, which shows that only a few users based on the k-means approach were assigned to this cluster.

6. SUMMARY & CONCLUSION

We presented a novel Pareto-dominance based clustering framework on Pareto-frontiers. Our framework provides several reasons for using this to manage large confusing sets of tuples with explicitly different domains. First, one can influence the result of the clustering by attaching weight to a more important dimension in order to cluster at least over one dimension at the appearance of Pareto-optima. Second, tuples can now be clustered over better-than relationships in order to avoid adjustments for utilization in different uses cases. Third, our preliminary benchmarks show that a Pareto-dominant clustering can be realized in adequate time. The quality of our approach is satisfying, because the stable clusters distinguish from them of the basic k-means clustering, but are still as similar as possible, especially regarding the affiliation of similar points w.r.t. the one-dimensional clustering.

Future work includes the integration of multi-dimensional clusters using Borda count. For more comprehensive benchmarks, we want to investigate experiments on higher dimensions. Furthermore experiments with ground truth data sets and comparisons to other types of clusterings, e.g., agglomerative and density-based clusterings are scheduled aswell.

Acknowledgements

This work has been partially funded by the German Federal Ministry for Economic Affairs and Energy according to a decision by the German Bundestag, grant no. ZF4034402LF5.

7. REFERENCES

[1] S. Börzsönyi, D. Kossmann, and K. Stocker. The Skyline Operator. In *Proceedings of ICDE '01*, pages 421–430, Washington, DC, USA, 2001. IEEE.

[2] C.-Y. Chan, H. V. Jagadish, K.-L. Tan, A. K. H. Tung, and Z. Zhang. Finding K-dominant Skylines in High Dimensional Space. In *Proceedings of SIGMOD '06*, pages 503–514, New York, NY, USA, 2006. ACM.

[3] A. Ferligoj and V. Batagelj. Direct Multicriteria Clustering Algorithms. *Journal of Classification*, 9(1):43–61, 1992.

[4] Z. Huang, Y. Xiang, B. Zhang, and X. Liu. A Clustering Based Approach for Skyline Diversity. *Expert Syst. Appl.*, 38(7):7984–7993, July 2011.

[5] A. K. Jain. Data Clustering: 50 Years Beyond K-means. *Pattern Recogn. Lett.*, 31(8):651–666, June 2010.

[6] W. Kießling. Foundations of Preferences in Database Systems. In *Proceedings of VLDB '02*, pages 311–322, Hong Kong, China, 2002. VLDB.

[7] D. T. Truong and R. Battiti. A Flexible Cluster-Oriented Alternative Clustering Algorithm for Choosing from the Pareto Front of Solutions. *Machine Learning*, 98(1):57–91, 2015.