Computer Science and Information Systems 19(2):887–912 https://doi.org/10.2298/CSIS211020015N

RG-SKY: A Fuzzy Group Skyline Relaxation for Combinatorial Decision Making

Sana Nadouri[1,2], Allel Hadjali[1] and Zaidi Sahnoun[2]

[1] LIAS/ISAE-ENSMA
Poitiers, France
sana.nadouri@{univ-constantine2.dz, ensma.fr}
allel.hadjali@ensma.fr
[2] LIRE/University of Constantine 2
Constantine, Algeria
zaidi.sahnoun@univ-constantine2.dz

Abstract. Skyline queries were recently expanded to group decision making to meet complex real-life needs encountered in many modern application domains that does not only require analyzing individual points but also groups of points. Group skyline aims at retrieving groups that are not dominated by any other group of the same size in the sense of a group-dominance relationship. It may often happens that this kind of dominance leads to only a small number of non-dominated groups which could be insufficient for the decision maker. In this paper, we propose to extend group skyline dominance by making it more demanding so that several groups leave incomparable. Then, the original group skyline will be enlarged by some interesting groups that are not much dominated by any other group. The key element of this relaxation is a particular fuzzy preference relation, named "much preferred", conveniently chosen. Furthermore, algorithms to compute the relaxed group skyline are proposed. Finally, a set of experiments are conducted on real, synthetic and generated data. Such experiments show that our proposal can really improve the decision process and satisfy user queries, insure reliability and decision quality.

Keywords: Data analysis, Group skyline queries, Relaxation, Fuzzy preferences, Decision making.

1. Introduction

Nowadays, multi-criteria analysis and decision making become more and more complex due to conflicting criteria and query complexity. Skyline operator [3], known as Maxima in computational geometry or Pareto in the business management field, manages this complexity using Pareto dominance. It extracts interesting objects from a dataset by respecting user preferences. It is particularly very successful in the database field, it has undergone an exponential interest due to the benefit that can be derived from it, even in real contexts. Skyline queries return then the most interesting points based on Pareto dominance relationship defined as follows: let a and b be two points (or database tuples) with the same number of attributes (also called dimensions), a dominates b, noted: ($a \prec b$), if a is as good as b in all dimensions and better than b in at least one dimension. If neither $a \prec b$ and nor $b \prec a$, then a and b are incomparable.

Many propositions and research works were published to study the skyline semantics and optimize its computation. Efficient algorithms were proposed to retrieve objects that present the optimal combination of the dataset characteristics [3,5,10,16,24,26,30,31]. Recently, the skyline definition turns out to be poor to deal with new complex real-world decision applications to answer different queries that require choosing a group of objects rather than individual objects of a dataset. Let us consider an example where a user wants to get the best Volleyball teams from a set of players, the traditional skyline returns the best players but not the best combinations to create a team that can not be dominated by any other existing team generated from the set of players. Another similar example consists of choosing a group of experts to review and evaluate papers based on the experts collective strength on multiple desired skills. This leads to the concept of Group Skyline [3] [14,38], noted G-SKY, which is very important and useful in many other domains like: groups recommendation, investments selection, detection of fire/crime (most dangerous places), etc. This novel concept has created new issues to the skyline community, for instance, generating groups and returning the appropriate skyline groups became the most challenging problems. Some solutions to these issues have been proposed in the literature [32,33]. However, querying a d-dimensional dataset using group skyline queries may lead to two particular scenarios: (i) a large number of skyline groups returned, which could be less informative for decision makers, (ii) a small number of skyline groups returned, which could be insufficient for decision makers. To solve the first problem (i), various approaches [15,29,40,43] are proposed to refine the group skyline, therefore reducing its size, but none of the existing work has addressed the problem (ii) to relax the group skyline in order to increase the number of group skyline results and thus satisfy better the decision makers needs.

Consider the following problem of finding the skyline l-groups (where l indicates the number of elements in each group from an n-tuple dataset): a decision maker (the trainer) wants to get the 5 best groups of 3 players by maximizing points and the number of blocks, when we run his/her query the system returns 2 groups of 3 Volleyball players each. Unfortunately, this answer does not satisfy the decision maker, he/she needs 5 teams of the best Beach-volleyball players but the traditional group skyline definition can return only 2 groups. It is the principal issue addressed in this paper. The solution advocated aims at enlarging the size of the group skyline by applying an appropriate relaxation process. This process consists of retrieving non-skyline groups by making more demanding the group-dominance relationship. To the best of our knowledge this is the first time the problem of group skyline relaxation is addressed.

Taking as starting point the study about the traditional skyline relaxation discussed in [2], we propose an extended group dominance relationship using a particular fuzzy preference relation, called *Much Preferred* (MP). The proposed approach allows increasing the group skyline with (non-skyline) groups that are only dominated to some extent by other groups in the sense of the extended group dominance introduced. The nature of the relation MP makes it more demanding the dominance between groups of the target dataset. In this context, a group still belong to the group skyline unless it is much dominated, in the spirit of the *MP relation*, by another skyline group. By this way, many groups are considered as incomparable and then as elements of the new relaxed group skyline (noted RG-SKY). Note that using the traditional group skyline definition such groups are pruned

[3] Named also combinatorial or compositional skyline

from the skyline groups. Furthermore, two algorithms (naive and optimized versions) with different cases to compute RG-SKY efficiently are provided. We also develop a set of experiments on real, synthetic and generated data to study and analyze the relevance and effectiveness of our proposal.

The remainder of the paper is organized as follows. In Section 2, we provide a necessary background and the problem description. In Section 3, we present a comparative study relying on a literature survey of related work. In Section 4, we define the RG-SKY concept and provide its properties w.r.t. both semantics and behavior. Algorithms for RG-SKY computation are also discussed. In Section 6, we evaluate the approach with different cases and discuss the results. Finally, Section 7 concludes by discussing the implications of this work in optimizing decision-making by increasing user satisfaction.

2. Background and Problem Description

This section presents a brief overview about the traditional skyline, group skyline and notions of fuzzy set theory used in this work. Then, it provides a description of the problem of interest. Table 2 provides the symbols with their meanings used in the rest of the paper.

Table 1. Symbols and their meanings

Symbol	Meaning
$\mathbb{D}$=(D_1, D_2, . . . , D_d)	*a set of d-dimensional data points*
d	*The number of dimensions of the set* $\mathbb{D}$
A_i	*The attribute* A_i
D_i	*The domain of* A_i
Q	*A point of* $\mathbb{D}$
g_i = (Q_1^i, ..., Q_l^i)	*A group of* l *points of* $\mathbb{D}$
$\mathbb{G}$ = (g_1, g_2, ..., g_l)	*The set of groups of size* l *of* $\mathbb{D}$
F	*An aggregation function*
MP	*Much preferred relation*
SKY	*The set of traditional skyline points*
G-SKY	*The group skyline (set of skyline groups)*
Rest	*The set of non skyline groups*
RG-SKY	*The relaxed group skyline*

2.1. Background

Traditional Skyline Queries Let $\mathbb{D}$ = (D_1, D_2, . . . , D_d) be a set of d-dimensional data points (that corresponds to a set of database tuples). We define a relation $R(A_1, A_2, ..., A_d)$ in $\mathbb{D}$ and we assume the existence of a total order relation on each domain D_i.

The traditional skyline query is based on Pareto dominance relationship defined as follows. Let a and b be two different points in $\mathbb{D}$, a dominates (in Pareto sense) b, denoted by a $\prec$ b, if for all i, a[i] $\leq$ b[i], and for at least one i, a[i] $<$ b[i], where a[i] (Resp. b[i]) is the value of the point a (Resp. b) for the attribute A_i and 1 $\leq$ i $\leq$ d. Formally,

$$a \prec b \Leftrightarrow \forall i \in \{1, ..., d\} : a[i] \leq b[i] \; and \; \exists j \in \{1, ..., d\} : a[j] < b[j] \qquad (1)$$

Without loss of generality, we consider in this definition the smallest value, the better.
The skyline of $\mathbb{D}$, denoted SKY($\mathbb{D}$), is a set of points that are not dominated (in Pareto

sense) by any other point from $\mathbb{D}$. Formally,

Hotels	Distance	Price	Is a Skyline ?
h1	1200	200	✗
h2	500	1400	✓
h3	800	2000	✗
h4	800	600	✓
h5	600	1400	✗
h6	100	2100	✓
h7	1000	100	✓

✓ Yes ✗ No

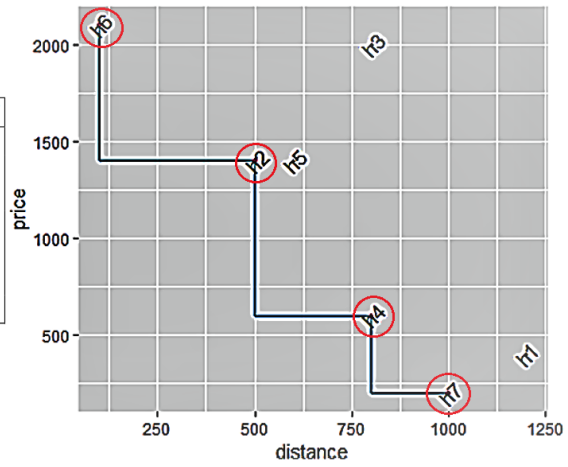

Fig. 1. The hotels conference Skyline points

The SQL[4] skyline query format is an SQL extension that incorporates a new clause **SKYLINE** where user preferences are specified [11]. The proposed SQL skyline syntax based on Borzsony [3] writes as follows:
"SELECT * FROM ... WHERE ... SKYLINE ... ORDER BY ..."

Example 1. Consider the example of a conference PhD student participant who wants to reserve a hotel room (see in Figure 1 the set of hotels). She/He wants the hotel to be close to the conference place and also to pay a reasonable price. The corresponding SQL skyline query writes:
"**SELECT** Hotels, Distance, Price **FROM** Hotels **SKYLINE** OF Price MIN, Distance MIN"
where MIN specifies that the two attributes should be minimized. Figure 1 shows the skyline points circled in red (i.e., $SKY = \{h_6, h_2, h_4, h_7\}$) that answer the user's query to get the best hotels that satisfy the students criteria. ■

Group Skyline Queries Despite the success of the traditional skyline, which focuses on top-1 solutions, it is inadequate when optimal groups are searched rather than individual points. For this reason, the skyline community proposed to extend the skyline definition to the combinatorial context to deal with group skyline instead of individual Skyline.

This new type of skyline queries extension relies on a dominance relationship between groups, also called combinatorial skyline queries. Group skyline returns groups that are not dominated by any other group in the Dataset. The dominance relationship between groups can be formulated in two different ways as follows:

– The first formulation relies on Pareto dominance and an aggregate function conveniently chosen. The dominance between groups is named *G-Skyline*.

[4] SQL stands for Structured Query Language· SQL allows us to access and manipulate databases

– The second semantics introduces a particular generalized dominance relation applied on permutations of groups to be compared. This type of dominance is named *G-Dominance*.

This leads thus to the two following formal definitions [43] (where g and g' are two groups with the same size):

Definition 1. ***(G-Skyline)****. Let F be an aggregate function and g = $\{Q_1, Q_2, ..., Q_l\}$ (resp. g' = $\{Q'_1, Q'_2, ..., Q'_l\}$) be a group represented by a point Q (resp. Q') with Q = F($Q_1, Q_2, ..., Q_l$) (resp. Q' = F($Q'_1, Q'_2, ..., Q'_l$)). For two distinct groups g and g', g dominates g' (denoted g $\prec_{gs}$ g') if Q dominates Q' (Q $\prec$ Q') in Pareto sense.*

Definition 2. ***(G-Dominance)****. Let us consider the two previous groups g and g', g G-dominates g' (denoted g $\prec_{gd}$ g'), if two permutations of l points can be found for g and g', g = $\{Q_{u1}, Q_{u2}, ..., Q_{ul}\}$ and g' = $\{Q'_{v1}, Q'_{v2}, ..., Q'_{vl}\}$, such that $Q_{ui} \preceq Q'_{vi}$ [5] for all i (1 $\leq$ i $\leq$ l) and $Q_{ui} \prec Q'_{vi}$ for at least one i.*

The group skyline of a dataset $\mathbb{D}$, denoted G-Sky($\mathbb{D}$), is a set of groups that are not dominated by any other group of $\mathbb{D}$ in the sense of definition 1 or 2. Formally, we write:

$$G - SKY(\mathbb{D}) = \{g \in \mathbb{G} \mid \nexists g' \in \mathbb{G} : g' \ltimes g\} \tag{3}$$

where $\ltimes$ stands for the relation $\prec_{gs}$ or $\prec_{gd}$.

Table 2. Players data

Players	points	rebounds
p_1	3	3
p_2	0	4
p_3	4	1
p_4	2	3
p_5	2	2
p_6	2	1

Example 2. Let us consider an example consisting of team players selection. Assume that we have six players p_1,...,p_6 shown in table 2 and we need team players formed by two players. In table 3, we generate all the possible groups of two players. The manager likes to extract the best teams based on the scored points and rebounds attributes of the players (i.e., **the greatest value, the better**).

– Based on the **Definition 1** and MAX aggregation function on each attribute, the values w.r.t. points and rebounds of each player of the generated teams are given in Column 3 of table 3. For instance, g_1 has two players p_1 and p_2, the values of $g_1 = (max(3, 0), max(3, 4)) = (3, 4)$. Now, applying the Pareto dominance on the generated teams, one can check that the group skyline contains only the team g_6.

[5] $Q_j \preceq Q'_j$ means that $Q_j \prec Q'_j$ or $Q_j \sim Q'_j$ where $\sim$ stands for the indifference relation (i.e., equally preferable). The indifference relation reduces to equality if each domain D_i is endowed with a total order.

Table 3. Skyline Teams (using definition 1)

Groups	Players	MAX (points,rebounds)	Group skyline
g_1	p_1(3,3) - p_2(0,4)	G_1(3,4)	✗
g_2	p_1(3,3) - p_3(4,1)	G_2(4,3)	✗
g_3	p_1(3,3) - p_4(2,3)	G_3(3,3)	✗
g_4	p_1(3,3) - p_5(2,2)	G_4(3,3)	✗
g_5	p_1(3,3) - p_6(2,1)	G_5(3,3)	✗
g_6	p_2(0,4) - p_3(4,1)	G_6(4,4)	✓
g_7	p_2(0,4) - p_4(2,3)	G_7(2,4)	✗
g_8	p_2(0,4) - p_5(2,2)	G_8(2,4)	✗
g_9	p_2(0,4) - p_6(2,1)	G_9(2,4)	✗
g_{10}	p_3(4,1) - p_4(2,3)	G_{10}(4,3)	✗
g_{11}	p_3(4,1) - p_5(2,2)	G_{11}(4,2)	✗
g_{12}	p_3(4,1) - p_6(2,1)	G_{12}(4,1)	✗
g_{13}	p_4(2,3) - p_5(2,2)	G_{13}(2,3)	✗
g_{14}	p_4(2,3) - p_6(2,1)	G_{14}(2,3)	✗
g_{15}	p_5(2,2) - p_6(2,1)	G_{15}(2,2)	✗

✓ Yes ✗ No

– Based on **Definition 2**, let us consider the two groups g_1 ={p_1(3,3), p_2(0,4)} and g_7 ={p_2(0,4), p_4(2,3)}. Since here the greatest value, the better, one can check that g_1 g-dominates g_7. Indeed, one can find a permutation for g_1 given by {p_2(0,4), p_1(3,3)} such that p_2(0,4) $\succeq$ p_2 (0,4) and p_1(3,3) $\succ$ p_4(2,3) (where $\succeq$ and $\succ$ are respectively *the preferred-or-equal relation* and *Pareto dominance relation* based on the order relations $\geq$ and $>$). Therefore, g_7 is not a skyline group. The same process leads us to a set of skyline group {g_1, g_2, g_3, g_6}, these are the only skyline groups as no other group with 2 points can g-dominate g_1, g_2, g_3 and g_6.

Based on the experimental evaluation done in [41], it has been proved that it is more interesting to consider **Definition 1** because monotone function definition is a subset of the permutation definition and generally it returns less information (i.e., a small number of skyline groups) compared to **Definition 2**. In the rest of the paper, we make use of the **Definition 1** when computing the group skyline. ■

Fuzzy Set Theory: A refresher The first article in fuzzy set theory written by Zadeh in 1965 [36] shows the intention of the author to generalize the classical notion of a set and a proposition to accommodate fuzziness to represent classes or sets of objects with all-defined boundaries. These sets allow us to describe gradual transitions between total membership and absolute rejection. Typical examples of these fuzzy classes are those described using adjectives or adverbs of the natural language, such as not cheap, young and tall. Formally, a fuzzy set F on the universe X is described by a membership function μ_F: X $\rightarrow$ [0, 1], where μ_F(x) represents the degree of membership of x in F.

Using this definition, if μ_F(x)=0 then the element x $\notin$ F, if μ_F(x) = 1 then x $\in$ F, these elements represent the core of F denoted by CORE(F) = {x $\in$ F | μ_F(x) = 1}. When $0 < \mu_F(x) < 1$, it became a partial membership, these elements form the support of F denoted by SUPP(F) = {x $\in$ F | μ_F(x)$>$ 0}. The complement of F, denoted $\overline{F}$, is defined by $\mu_{\overline{F}}(x)$

= 1 - $\mu_F(x)$. More μ_F(x) is close to the value 1, more x belongs to F. Therefore, given x, y ∈ F, we say that x is preferred to y iff μ_F(x) > μ_F(y). If μ_F(x) = μ_F(y), then x and y have the same preference.

In practice, F can be represented by a trapezoid membership function (t.m.f) $(\alpha, \beta, \varphi, \psi)$, where $[\beta, \varphi]$ is the core and $]\alpha, \psi[$ is its support (see Figure 2). This kind of membership functions in addition to its simple representation (only a quadruplet of values is need), leads to uncomplicated computational operations as well.

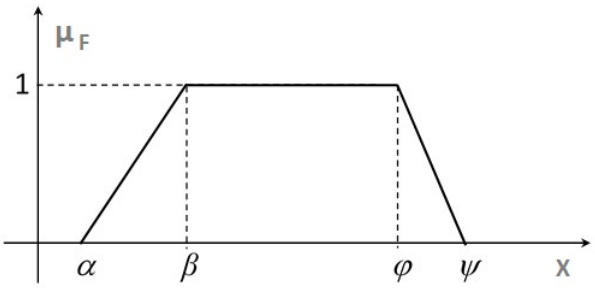

Fig. 2. Trapezoidal membership function

2.2. Problem description

Let $\mathbb{Q}$ be a user skyline query and $\mathbb{D}$ be the target dataset. Assume the user wants to find K groups of l elements for decision purposes. Let G-SKY($\mathbb{D}$) be the group skyline computed and | G-SKY($\mathbb{D}$) | be its size.

One can easily observe that if | G-SKY($\mathbb{D}$) | < K, the user is not then able to make the desired decision due the insufficient skyline groups returned. The problem of interest is then how to enlarge the size of G-SKY($\mathbb{D}$) in order to obtain a relaxed variant, called RG-SKY($\mathbb{D}$), with more groups (i.e., G-SKY($\mathbb{D}$) ⊆ RG-SKY($\mathbb{D}$)). We call this problem the group skyline relaxation problem.

3. Related work

We review here the main research works related to group skyline both from the computation and semantics point of view. We also provide a comprehensive comparison of those works w.r.t. to a set of criteria conveniently chosen.

The brute method to get G-SKY is to enumerate all the groups, and then run the query based on the group dominance relationship. The brute force method computes the aggregate tuple for each group, then uses any traditional skyline algorithm to find the group skyline. This method is significantly time consuming and the storage cost can be exponential due to the huge intermediate input for the traditional skyline tuple algorithm. In [38] some alternatives to this naive method are proposed. The existing group skyline propositions focus on the (i) problem using one of the definitions presented in Section 2 on stream or static data, e.g,. the papers [15,29,32,33,37,40,43] combine the advantages of group skyline and top-k queries. Another work [19] presents a structure that represents the points in a directed skyline graph and captures all the dominance relationship among the

points based on the notion of skyline layers. Some papers [8,12,17] focus on stream data to compute G-SKY continuously, where they invoke the problem of computing G-SKY when a new point p arrives dynamically. The underlying idea is to store dominance information that could be reused in another search space pruning. In [14,18], authors generate candidate groups in a progressive manner and update the resulting group skyline dynamically. Some other papers [6,7,13,34,35,39,41,42,44] focus on optimizing experimental performance of the group skyline algorithm by using Parallelization and other methods. Based on our previous survey papers [21,23], we summarize and compare the above existing works in Table 4 (where (i) **D.type (Data type):** stands for stream or static data, (ii) **Perf (Performance):** means that the work focuses on the execution time rather than the quality of the responses, (iii) **Def (Definition):** indicates which definition 1 or 2 used to compute the group skyline, (iv) **Ref (Refinement):** means that the work is endowed with a refining process to reduce the skyline groups set returned and (v) **Rel (Relaxation):** means that the work tries to get more skyline groups by relaxing the definition of the traditional skyline groups).

As it can be seen, none of the previous work deals with the silence problem (Namely, the set of answers is empty or insufficient to decision making) in the group skyline context compared to the traditional skyline where we can cite the two papers [2,20] that use respectively the kNN definition and a fuzzy preference relation to relax the skyline result.

Table 4. Group Skyline Related Work comparison

Related Work	D.type	Perf.	Def.	Ref.	Rel.
Efficient computation of combinatorial skyline queries [6]	Static	✓	1	✓	✗
An Efficient Algorithm to Compute Compositional Skyline[7]	Static	✓	None	✓	✗
Finding Group-Based Skyline over a Data Stream in the Sensor Network[8]	Stream	✓	2	✓	✗
Efficient processing of skyline group queries over a data stream[12]	Stream	✓	2	✓	✗
Combination skyline queries[13]	Static	✓	1	✓	✗
Group skyline computation[14]	Static	✓	2	✓	✗
Incremental evaluation of top-k combinatorial metric skyline query[15]	Static	✓	None	✓	✗
Progressive approaches for Pareto optimal groups computation[17]	Stream	✓	None	✓	✗
Discovering Group Skylines with Constraints by Early Candidate Pruning[18]	Static	✓	1	✓	✗
Finding pareto optimal groups: Group-based skyline[19]	Static	✓	1	✓	✗
Top-k combinatorial skyline queries[29]	Static	✓	1	✓	✗
Identifying Most Preferential Skyline Product Combinations[32]	Static	✓	None	✓	✗
Identifying most preferential skyline product combinations under price promotion[33]	Static	✓	None	✓	✗
Efficient Contour Computation of Group-based Skyline[34]	Static	✓	2	✓	✗
Fast algorithms for pareto optimal group-based skyline[35]	Static	✓	2	✓	✗
Finding k-Dominant G-Skyline Groups on High Dimensional Data[37]	Static	✓	None	✓	✗
On skyline groups[38]	Static	✓	2	✓	✗
Finding optimal skyline product combinations under price promotion [39]	Static	✓	None	✓	✗
Top-k Dominating Queries on Skyline Groups[40]	Static	✓	1,2	✓	✗
Computing skyline groups: an experimental evaluation[41]	Static	✓	1,2	✓	✗
Computing Skyline Groups: An Experimental Evaluation [42]	Static	✓	1,2	✓	✗
Top-k Skyline Groups Queries[43]	Static	✓	1	✓	✗
Parallelization of group-based skyline computation for multi-core processors[44]	Static	✓	2	✓	✗

✓ Yes ✗ No

4. Group skyline relaxation approach

We discuss here our fuzzy approach to relax the group skyline. The idea is to extend the group dominance (given in Definition 1) by making it more demanding. The relaxed group Skyline obtained, RG-SKY, is no longer a flat set but a discriminated set where each of its elements is associated with a degree.

The main idea consists of computing the extent to which a group, discarded by the G-Skyline dominance relationship (denoted $\prec_{gs}$, see Definition 1), may belong to the relaxed group skyline. To this end, and as it will be illustrated further, we associate with each skyline attribute A_i ($i \in \{1, \cdots, d\}$) a pair of parameters $(\gamma_{i1}, \gamma_{i2})$ where γ_{i1} and γ_{i2} respectively denote the bounds of the relaxation zone allowed to the attribute A_i. A vector of pairs of parameters, denoted γ, is then defined as

$$\gamma = ((\gamma_{11}, \gamma_{12}), \cdots, (\gamma_{d1}, \gamma_{d2})).$$

It is worthy to note that γ, called a relaxation parameter vector, is a user-defined[6]. It defines the set of values w.r.t each attribute that user can tolerate despite they are ruled out when applying the dominance $\prec_{gs}$.

4.1. Fuzzy group dominance

RG-SKY, the relaxed group skyline of G-SKY, relies on a particular dominance relationship (inspired from the work [2]) that allows enlarging the group skyline with the most interesting groups among those ruled out when computing G-SKY using Definition 1. This dominance relationship makes use of the fuzzy relation "Much Preferred ($MP_{\mathbb{G}}$)" to compare two groups g and g'. So, g is an element of RG-SKY if there is no group g' $\in \mathbb{G}$ such that g' is much preferred to g (denoted $MP_{\mathbb{G}}$(g',g)) in all group skyline attributes. Formally, we write:

$$g \in RG - SKY \Leftrightarrow \nexists g' \in \mathbb{G}, MP_{\mathbb{G}}(g', g) \tag{4}$$

Note that g' is much preferred to g in the sense of $MP_{\mathbb{G}}$ if and only if g' is much preferred to g w.r.t. to all group skyline dimension i in $\{1, ..., d\}$. Formally, we write:

$$MP_{\mathbb{G}}(g', g) \Leftrightarrow \forall i \in \{1, ..., d\}, MP_{\mathbb{G}_i}(g'(i), g(i)) \tag{5}$$

where $MP_{\mathbb{G}_i}$ is a defined on the domain D_i of the attribute A_i, $g(i) = F(Q_1[i], ..., Q_l[i])$ (resp. $g'(i) = F(Q'_1[i], ..., Q'_l[i])$ and F is an aggregate function. Recall that $(g'(i), g(i)) \in MP_{\mathbb{G}_i}$ means that $\mu_{MP_{\mathbb{G}_i}}(g'(i), g(i)) > 0$ (or $MP_{\mathbb{G}_i}(g'(i), g(i)) > 0$ for short). In a similar way, $(g', g) \in MP_{\mathbb{G}}$ means also $\mu_{MP_{\mathbb{G}}}(g', g) > 0$ (or $MP_{\mathbb{G}}(g', g) > 0$ for short).

Note that $MP_{\mathbb{G}_i}(g'(i), g(i))$ expresses the extent to which the value $g'(i)$ is much preferred to the value $g(i)$. Since $MP_{\mathbb{G}_i}$ is of a gradual nature, each element g of RG-SKY is associated with a degree ($\in$[0,1]) expressing the extent to which g belongs to RG-SKY. Now in fuzzy set terms, one can write equation (4) as follows (where the quantifiers $\forall$ and $\exists$ are modeled by the min and max operators respectively):

[6] The user predefine the values or the degree of tolerance of each dimension in a form of a vector of relaxation.

$$\mu_{RG-SKY}(g) = 1 - \max_{g' \in \mathbb{G}-\{g\}} \min_i \mu_{MP_{\mathbb{G}_i}}(g'(i), g(i)) = \min_{g' \in \mathbb{G}-\{g\}} \max_i (1 - \mu_{MP_{\mathbb{G}_i}}(g'(i), g(i))) \tag{6}$$

The semantics of the fuzzy relation $MP_{\mathbb{G}_i}$ can be expressed by the following formulas (7) (see also figure 3).

$$\mu_{MP_{\mathbb{G}_i}^{(\gamma_{i1},\gamma_{i2})}}(g'(i), g(i)) = \begin{cases} 0 & \text{if } g'(i) - g(i) \leq \gamma_{i1} \\ 1 & \text{if } g'(i) - g(i) \geq \gamma_{i2} \\ \frac{(g'(i)-g(i))-\gamma_{i1}}{\gamma_{i2}-\gamma_{i1}} & \text{elsewhere} \end{cases} \tag{7}$$

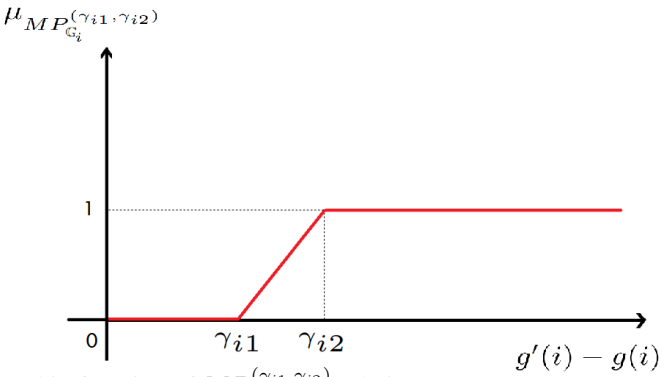

Fig. 3. Membership function of $MP_{\mathbb{G}_i}^{(\gamma_{i1},\gamma_{i2})}$ relation

For instance, if $g'(i) - g(i) \geq \gamma_{i2}$ then $g'(i)$ is completely *much preferred* to $g(i)$. One can also see that if $g'(i) - g(i) > \gamma_{i1}$, $g'(i)$ is not only *preferred* but *much preferred* to $g(i)$ to some extent. In terms of t.m.f., the fuzzy set associated with $MP_{\mathbb{G}_i}$ writes $(\gamma_{i1}, \gamma_{i2}, \infty, \infty)$, and denoted $MP_{\mathbb{G}_i}^{(\gamma_{i1},\gamma_{i2})}$. It is easy to check that $MP_{\mathbb{G}_i}^{(0,0)}$ corresponds to the crisp preference relation expressed by means of the regular relation *"greater than"*.

Now, let $RG\_SKY^{(\gamma)}$ be the relaxed group skyline computed on the basis of the relaxation vector $\gamma = ((\gamma_{11}, \gamma_{12}), \cdots, (\gamma_{d1}, \gamma_{d2}))$ in the case of d skyline attributes. One can easily check that the group skyline $G - SKY = RG\_SKY^{(\mathbf{0})}$ with $\mathbf{0} = ((0,0), \cdots, (0,0))$.

One can also check that the following monotonicity property holds.

Proposition 1. *Let γ and γ' be two relaxation parameter vectors. Then, the following propriety holds:*

$$\gamma' \leq \gamma \Rightarrow RG - SKY^{(\gamma')} \subseteq RG - SKY^{(\gamma)}$$

Proof. (Sketch) Let $\gamma = ((\gamma_{11}, \gamma_{12}), \cdots, (\gamma_{d1}, \gamma_{d2}))$ and $\gamma' = ((\gamma'_{11}, \gamma'_{12}), \cdots, (\gamma'_{d1}, \gamma'_{d2}))$ two relaxation parameter vectors. $\gamma' \leq \gamma \Rightarrow \forall i \in \{1, ..., d\}\ \gamma'_{i1} \leq \gamma_{i1}$ and $\gamma'_{i2} \leq \gamma_{i2}$. This implies that $\forall i \in \{1, ..., d\}\ MP_{\mathbb{G}_i}^{(\gamma_{i1},\gamma_{i2})} \subseteq MP_{\mathbb{G}_i}^{(\gamma'_{i1},\gamma'_{i2})}$. Based on (6), one can deduce that $RG - SKY^{(\gamma')} \subseteq RG - SKY^{(\gamma)}$ holds.

Lemma 1. *Let $\gamma = ((0, \gamma_{12}), \cdots, (0, \gamma_{d2}))$, $\gamma' = ((\gamma'_{11}, \gamma'_{12}), \cdots, (\gamma'_{d1}, \gamma'_{d2}))$ and $\forall\ i \in \{1, ..., d\}\ \gamma_{i2} < \gamma'_{i2}$, the following result holds as well:*

$$RG - SKY^{(0)} \subseteq RG - SKY^{(\gamma)} \subseteq RG - SKY^{(\gamma')}$$

Table 5. Hotels conference example

Hotels	Price (Euro)	Distance (Km)
h_1	50	10
h_2	100	6
h_3	150	5
h_4	40	11

Example 3. (Continued) To illustrate the interest of the RG-SKY set, let us consider a simple example of 4 hotels with close values as depicted in Table 5. Assume the user wants to retrieve the K = 3 best groups of hotels by minimizing the two dimensions price and the distance, to get the closest hotels and the cheapest.

To this end, we proceed as follows:

1. Generate the skyline points: It is easy to check SKY = $\{ h_1, h_2, h_3, h_4\}$ because they are incomparable.
2. Generate the groups of size 3 using the binomial coefficient[7] and apply the MIN aggregation function (F_{min}):
   $g_1 = \{h_2, h_3, h_4\}$, $F_{min}(g_1)$=<min(100,150,40), min(6,5,11)>=<40, 5>
   $g_2 = \{h_1, h_3, h_4\}$, $F_{min}(g_2)$=<min(50,150,40), min(10,5,11)>=<40, 5>
   $g_3 = \{h_1, h_2, h_4\}$, $F_{min}(g_3)$=<min(50,100,40), min(10,6,11)>=<40, 6>
   $g_4 = \{h_1, h_2, h_3\}$, $F_{min}(g_4)$=<min(50,100,150), min(10,6,5)>=<50, 5>
3. By applying Definition 1, one can check that the group skyline is
   G-SKY = $\{g_1, g_2\}$.
   Unfortunately, the user receives only 2 skyline groups even if all tuples are skyline points while (s)he needs 3 groups to make a decision. To satisfy the user's needs, we call then the RG-SKY method.

RG-SKY method:
Let us first assume that the relaxation vector $\gamma = ((0.5, 1), (0.5, 1))$, i.e. $(0.5, 1)$ both for the skyline attributes "Price" and "Distance". According to equation (7), one can check that the fuzzy relation $MP_{\mathbb{G}_{Price}}$ can write:

$$\mu_{MP^{(0.5,1)}_{\mathbb{G}_{Price}}}(v, u) = \begin{cases} 0 & \text{If } v - u \leq 0.5 \\ 1 & \text{If } v - u \geq 1 \\ \frac{v-u-0.5}{1-0.5} & \text{Otherwise} \end{cases} \quad (8)$$

The fuzzy relation $MP_{\mathbb{G}_{Distance}}$ can also be written in a similar way.
Let us now compute the fuzzy set RG-SKY using equation (6) (where $i = 1$ and $i = 2$ denote the attributes *Price* and *Distance* respectively):

$\mu_{RG-SKY}(g_3) = 1 - \max\limits_{g' \in \{g_1, g_2, g_4\}} \min\limits_{i \in \{1,2\}} \mu_{MP_{\mathbb{G}_i}}(g'(i), g_3(i))$
$\mu_{RG-SKY}(g_3)$ = 1 - max[min($\mu_{MP_{\mathbb{G}_1}}(g_1(1), g_3(1))$,$\mu_{MP_{\mathbb{G}_2}}(g_1(2), g_3(2))$), $min(\mu_{MP_{\mathbb{G}_1}}(g_2(1),$ $g_3(1))$, $\mu_{MP_{\mathbb{G}_2}}(g_2(1), g_3(2)))$, $min(\mu_{MP_{\mathbb{G}_1}}(g_4(1), g_3(1))$, $\mu_{MP_{\mathbb{G}_2}}(g_4(2), g_3(2)))$]

[7] The binomial coefficient is noted $C(n, k)$ and reads choose k among n and is defined by the formula $C(n, k) = n!/(k!(n - k)!)$ with $n!$ stands for the factorial of n.

$\mu_{RG-SKY}(g_3)$ = 1 - max[min(0, 0), min(0, 0), min(1, 0)] = 1 - 0 = 1

In a similar way, we obtain $\mu_{RG-SKY}(g_4) = 1$. Then,

RG-SKY = $\{1/g_1, 1/g_2, 1/g_3, 1/g_4\}$.

One can observe that RG-SKY contains the G-SKY elements (i.e., g_1 and g_2 with a degree equals 1) and some new groups that were not in G-SKY with a degree equals 1 (i.e., g_3 and g_4). RG-SKY is more larger than G-SKY and can satisfy the initial user query by returning for instance the groups g_1, g_2 and g_3.
Now, if RG-SKY contains more than K groups, the K best groups are returned. In case of ties, the user can establish a rank-order on the basis of the preferences w.r.t. the skyline attributes (in our case, the *Price* and *Distance* attributes). As for the case where RG-SKY contains less than K groups, one can revise the relaxation vector and re-execute the RG-SKY method. ■

4.2. Some basic properties

We establish here a set of desirable properties that are of interest for computation purpose. Some of them are the fuzzy counterparts of group skyline proprieties introduced in [14]. Let $\gamma = ((\gamma_{11}, \gamma_{12}), \cdots, (\gamma_{d1}, \gamma_{d2}))$ and $\gamma' = ((\gamma'_{11}, \gamma'_{12}), \cdots, (\gamma'_{d1}, \gamma'_{d2}))$ be two relaxation parameter vectors (where $MP^{\gamma}_{\mathbb{G}}(g, g')$
> 0 means $(g, g') \in MP^{\gamma}_{\mathbb{G}}$):

Proposition 2. *(Min-Asymmetry) Let g and g' be two groups of* $\mathbb{G}$*,*

$$\textit{If } (g, g') \in MP^{\gamma}_{\mathbb{G}} \textit{ then } (g', g) \notin MP^{\gamma}_{\mathbb{G}}.$$

Proof. Proposition 2 can also be written in the form: If $MP^{\gamma}_{\mathbb{G}}(g, g') > 0$ then $MP^{\gamma}_{\mathbb{G}}(g', g)$ $= 0$. Now, due to the asymmetry property of the fuzzy preferences [28], one can write: $min(MP^{(\gamma_{i1}, \gamma_{i2})}_{\mathbb{G}_i}(g(i), g'(i)), MP^{(\gamma_{i1}, \gamma_{i2})}_{\mathbb{G}_i}(g'(i), g(i))) = 0, \forall i \in \{1, ..., d\}$. Namely, if $MP^{(\gamma_{i1}, \gamma_{i2})}_{\mathbb{G}_i}(g(i), g'(i)) > 0$ then $MP^{(\gamma_{i1}, \gamma_{i2})}_{\mathbb{G}_i}(g'(i), g(i)) = 0$.

Proposition 3. *(Min-Transitivity) Let g, g' and g" be three groups of* $\mathbb{G}$*,*

$$\textit{If } (g, g') \in MP^{\gamma}_{\mathbb{G}} \textit{ and } (g', g") \in MP^{\gamma'}_{\mathbb{G}} \textit{ then } (g, g") \in MP^{\gamma+\gamma'}_{\mathbb{G}}.$$

Proof. Proposition 3 can writes also in the form: If $MP^{\gamma}_{\mathbb{G}}(g, g') > 0$ and $MP^{\gamma'}_{\mathbb{G}}(g', g") >$ 0 then $MP^{\gamma+\gamma'}_{\mathbb{G}}(g, g") > 0$. Now, due to the transitivity property of the fuzzy preferences [28] and to the fuzzy addition formula [9], one can write:
$min(MP^{(\gamma_{i1}, \gamma_{i2})}_{\mathbb{G}_i}(g(i), g'(i)), MP^{(\gamma'_{i1}, \gamma'_{i2})}_{\mathbb{G}_i}(g'(i), g"(i))) \leq MP^{(\gamma_{i1}+\gamma'_{i1}, \gamma_{i2}+\gamma'_{i2})}_{\mathbb{G}_i}(g(i),$
$g"(i)), \forall i \in \{1, ..., d\}$.

Proposition 4. *If g* $\subset$ *SKY(*$\mathbb{D}$*) then g* $\in$ *RG-SKY(*$\mathbb{D}$*) does not always hold.*

Proof. To show that this Proposition is not always true, it suffices to exhibit a counterexample. Consider a 2-dimensional (points, rebounds) dataset of 4 players: p_1 = (0,4), p_2 = (1,2), p_3 = (2,1), and p_4 = (4, 0). It is easy to see that $SKY = \{p_1, p_2, p_3, p_4\}$. Let $g = \{p_1, p_4\}$ ($\subset$ SKY). One can check that $g \notin$ RG-SKY for $\gamma = ((0, 1), (0, 1))$.

Proposition 5. *(The converse of Proposition 4) If $g \in$ RG-SKY($\mathbb{D}$) then $g \subset$ SKY($\mathbb{D}$) does not always hold.*

Proof. Let us also find a counterexample. Consider a 2-dimensional (points, rebounds) dataset of 3 players: p_1 = (0, 2), p_2 = (1, 0), and p_3 = (2,1), we have SKY=$\{p_3\}$ and G-SKY=$\{\{p_1, p_3\}\}$ (i.e. contains 1 group of 2 points). For $\gamma = ((0,1),(0,1))$, one can check that $\{p_2, p_3\} \in RG - SKY$ while $\{p_2, p_3\} \notin SKY$.

Table 6. Comparison of the general Skyline algorithms

Name	Indexed	Limits and issues
Index [30]	✓	- Does not support user-defined preferences (the order of the returned points is fixed and depends on the distribution of the data values)
Bitmap [30]	✓	- The memory consumption limit due to the conversion of points to Bitmap structure - Bitmap also handles inefficiently updates because it implies the recalculation of all the bit vectors t - Does not work on several dimensions (expensive) - Does not allow the user to express preferences - Mandatory to code all the point values
NN [16,31]	✓	- Performance problem, in case, we have a single element, the algorithm continue the division to 4 regions whereas the program can check in advance the number of the existing points - It is not efficient if the data is not mass
BBS [25,27,31]	✓	- Does not work when dimensions exceed 5
BNL [3,31]	✗	- Requires a lot of iterations before the final skyline is calculated (it analyzes all the data) - It has a limit on the size of the window, the complexity of the algorithm depends on this size - A non-negligible comparison time is necessary - The skyline points are not defined progressively (they change)
D&C [3]	✗	- The skyline points are not defined progressively - Problem if the data is so small (the process becomes useless), it is more efficient when dealing with a large amount of data - The algorithm scans the entire database
SFS [5]	✗	- Scans all data - The necessity to define a good monotonous function
LESS [10]	✗	- Elimination-filter (EF) mechanism can become full - All the data must be scanned at least once

5. RG-SKY computation

RG-SKY computation is expected to be a part of a Decision Support System (DSS) [22]. Here, we provide (i) a diagram which gives an overview of how the RG-SKY approach works and (ii) the two proposed algorithms for computing the RG-SKY set.

5.1. RG-SKY diagram

Figure 4 provides an overview of the RG-SKY approach and illustrates the chronology of its steps in a comprehensive way. Two cases can be distinguished:

– **Case 1:** User request satisfaction - Relaxation unneeded
In this case, the decision maker sends a request with a set of conflicting conditions to the DSS system (that integrates the RG-SKY computation process), the traditional G-SKY computation returns a satisfactory answer and the RG-SKY process is then not triggered.

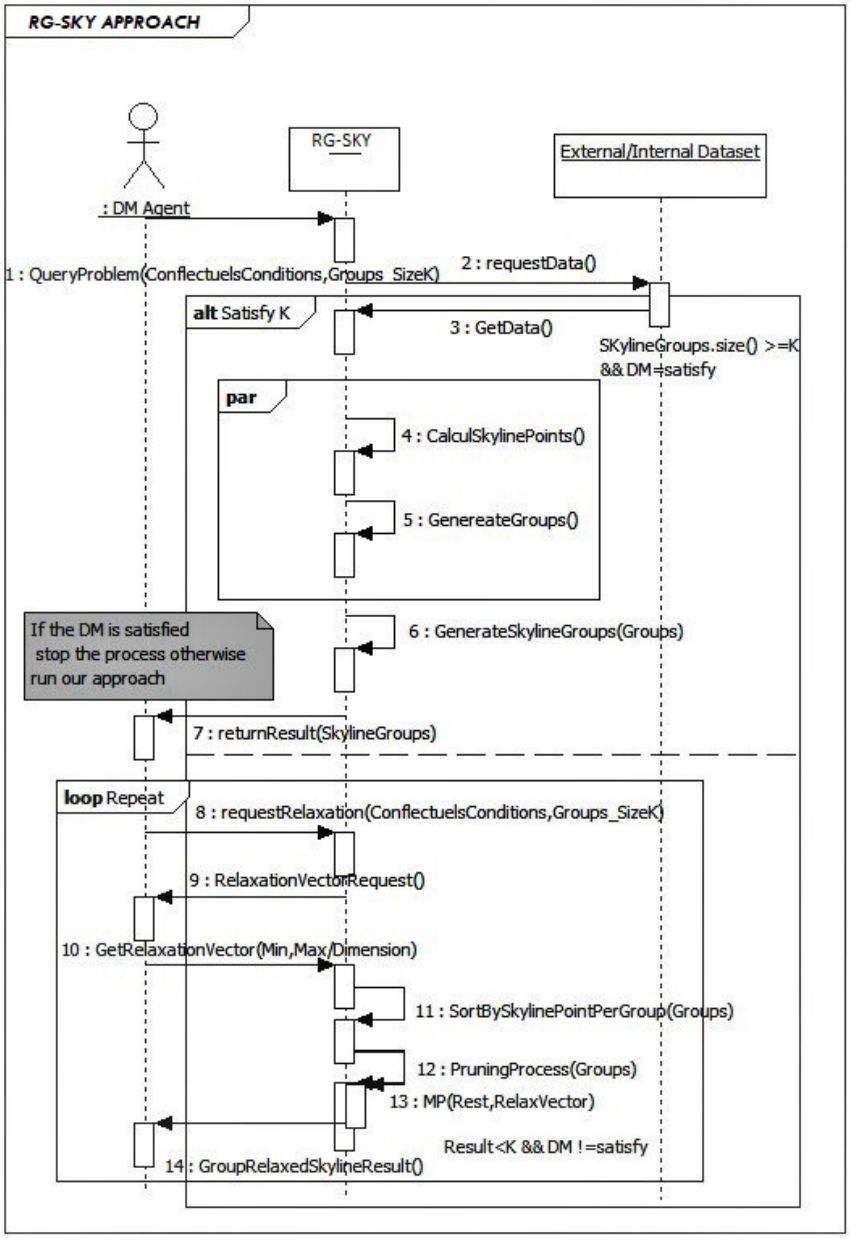

Fig. 4. Sequence diagram of the general RG-SKY approach

– **Case 2:** User request dissatisfaction - Relaxation needed
The traditional G-SKY computation returns an unsatisfactory answer. Before starting the relaxation process, the system first gets the relaxation parameter vector from the user. The system also executes some pre-processing and some pruning techniques to speed-up the RG-SKY computation. For instance, the following optimization strategies are implemented:

  - **Sorting groups by their number of skyline points:** this allows creating an hierarchy of the groups that helps intelligently to scan the research space.
  - **Pruning techniques:** they are based on the previous established propositions and properties.

  The RG-SKY computed represents a fuzzy set in the sense that each group is associated with a degree (expressing to what extent is not "much dominated" by any other group). If RG-SKY contains more K groups, the top-K groups are returned to the user and then the process stops. Otherwise, we revise the relaxation parameter vector (making it more permissive) and we re-compute the RG-SKY.

RG-SKY naive algorithm This subsection presents the first algorithm (Algorithm 1) proposed to implement the RG-SKY approach. It does not use any optimization technique to reduce the research space.

Algorithm 1: RG-SKY naive algorithm

Result: Relaxed Group Skyline
Input : G-SKY: Skyline groups, $\mathbb{G}$: Groups, K: integer, γ: relaxation Vector
Output: RG-SKY

```
RG-SKY ← G-SKY                    // The first step is the generation of G-SKY
GetRestGroups(G-SKY,𝔾);
SortbySkyPoints(G-REST);          // pre-treatment phase
DeleteZeroSkyGroups(G-REST);      // pruning phase
j ⟵ 1;
while (G-REST.length ≤ k) do
    // level: denotes the number of skyline points of the current group
    for(i=0;i<N;i++){
    Compile μ_MP (level[j].get(i),γ);      // g'∈Rest, μ_MPi(g'i,gi)
    j++;
end
+ Return RG-SKY
```

RG-SKY Salsa algorithm This algorithm is an improved version of the naive algorithm where we avoid analyzing all the groups generated to extract the skyline groups. This choice relies on the comparative study conducted on a set of well-known skyline algorithms, as explained below.

Algorithm 2: RG-SKY SALSA algorithm

Input : G-SKY: Group Skyline, $\mathbb{G}$: Groups, k: integer, γ: relaxation Vector, F: A monotone sorting function
Output: RG-SKY: Relaxed Group Skyline

```
1  RG-SKY ← G-SKY
2  GetRestGroups(G-SKY,𝔾);          // G-Rest
3  SortbySkyPoints(G-Rest);
4  DeleteZeroSkyGroups(G-REST);
5  RG-SKY ⟵ G-SKY; stop ⟵ false; pstop ⟵ undefined;
6  Sort G-REST according to F;      // F(G-REST);
7  while (¬ pstop and G-REST ≠ ∅ ) do
8  |  G ⟵ get next group from G-REST;
9  |  G-REST⟵ G-REST\{G}
10 |  if ¬ MP(G,G-SKY) then RG-SKY ⟵ RG-SKY ∪ {G}, update pstop then
11 |  if pstop > G-SKY then
12 |     stop := true;
13 end
14 Return RG-SKY;
```

To integrate the RG-SKY approach in our Decision Support System Model proposed in [22], knowing that group skyline query processing results in an expensive procedure, it is then important to choose the best adaptive skyline algorithm for our context. The choice of SaLSa (Sorting and Limit skyline algorithm) is justified by the fact that it overcomes the main limitations found in the other general skyline algorithms, as summarized in Table 6. In this comparative study, we have considered only the well-known algorithms in the skyline field. For a complete overview on skyline algorithms, see for instance [24,26].

SaLSa algorithm used in the traditional skyline query extraction, is an improvement of SFS and LESS (see Table 6). It strives to avoid scanning the entire sorted dataset as opposed to the previous propositions, it is the first algorithm that exploits the values of a monotonic notation (limitation) function to sort the data set to read and compare. SaLSa differs from the other generic algorithms because it consistently limits the number of points read and the dominance tests. The design of SaLSa is based on two key concepts: First, a sorting step of the input data and, second, suitably choose a sorting function that does not privilege any attribute over the others (the function does not influence the correctness of SaLSa but only its performance). For these reasons, we adapt the Salsa algorithm to the group skyline problem and to optimize also the relaxation process of the RG-SKY approach. During the filter phase, the algorithm reads and examines the rest of the groups. Each time a new group is read, it is compared to the current skyline group list. If a group dominates the current group, it is ignored, otherwise it is inserted to the relaxed skyline groups list (as a final relaxed group skyline) and the algorithm checks its termination trigger (Pstop). If the current threshold Pstop is less than or equal to the *fmin* value of the point, the algorithm ends and returns the entire skyline RG-SKY group list. This termination condition ensures that no data groups examined later should be part of the RG-SKY list, thereby the algorithm avoids analyzing the entire rest of the dataset.

There are many limiting functions in the literature, but the optimal function that can limit any input relation more than others do, is the Minimum Coordinate Function (MCF) comparing to Sum and Val presented in [1] (MCF is noted MinC, first it sorts groups considering the minimum coordinate value of the current group and simultaneously Sum function of group elements is calculated and used in case of ties).

Table 7. Set of parameters (where N stands for the number of input groups)

Parameter	Values	default value
Groups [N]	NBA={500}, HOUSE={5911}, WEATHER={4438}, Correlated={10,50,100,500,2000-3612281}, Anti-correlated={2700}, Independent={1500}	10 50 100 500 1000
Dataset distribution schema	Correlated, Independent, Anticorrelated	Correlated
Number of group dimensions [l]	2, 3, 4, 5	2
Relaxation vector [(γ_1, γ_2)]	$\gamma_1 \in [0, 1], \gamma_2 \in [0, 1]$	(0, 1)

6. Experimental study

Table 8. Specifications of real datasets

Dataset	Cardinality	Dimensionality
NBA	17,264	8
House	127,931	6
Weather	566,268	15

This section presents the experimental study carried out. It validates the effectiveness and the relevance of the RG-SKY approach to relax small group skylines and also measures some performance related to the computational time.

6.1. Experimental environment

The algorithms are tested on a Dell Inc Machine, System Model: Precision T1650 and run in a Windows 10 Education 64-bit (10.0, Build 16299) environment, using a 3.4GHz Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz(8 CPUs) with a main memory of 8GB RAM, in sequential mode (1 thread) and a 500 GB of disk. Dataset benchmark is generated using the method described in [3] following three conventional distribution schema (correlated, anti-correlated and independent) and also ”randataset”. The approach was developed in Eclipse Modeling Tools Version: Oxygen.3 Release (4.7.3), using Java.v9 language.

6.2. Experimental tests

The tests can be classified into two main parts: Real Data (NBA, HOUSE, WEATHER) and Synthetic data where we change the data type and size, the groups and tuples dimensions and finally generate data for the purpose of the relaxation parameter vector tests

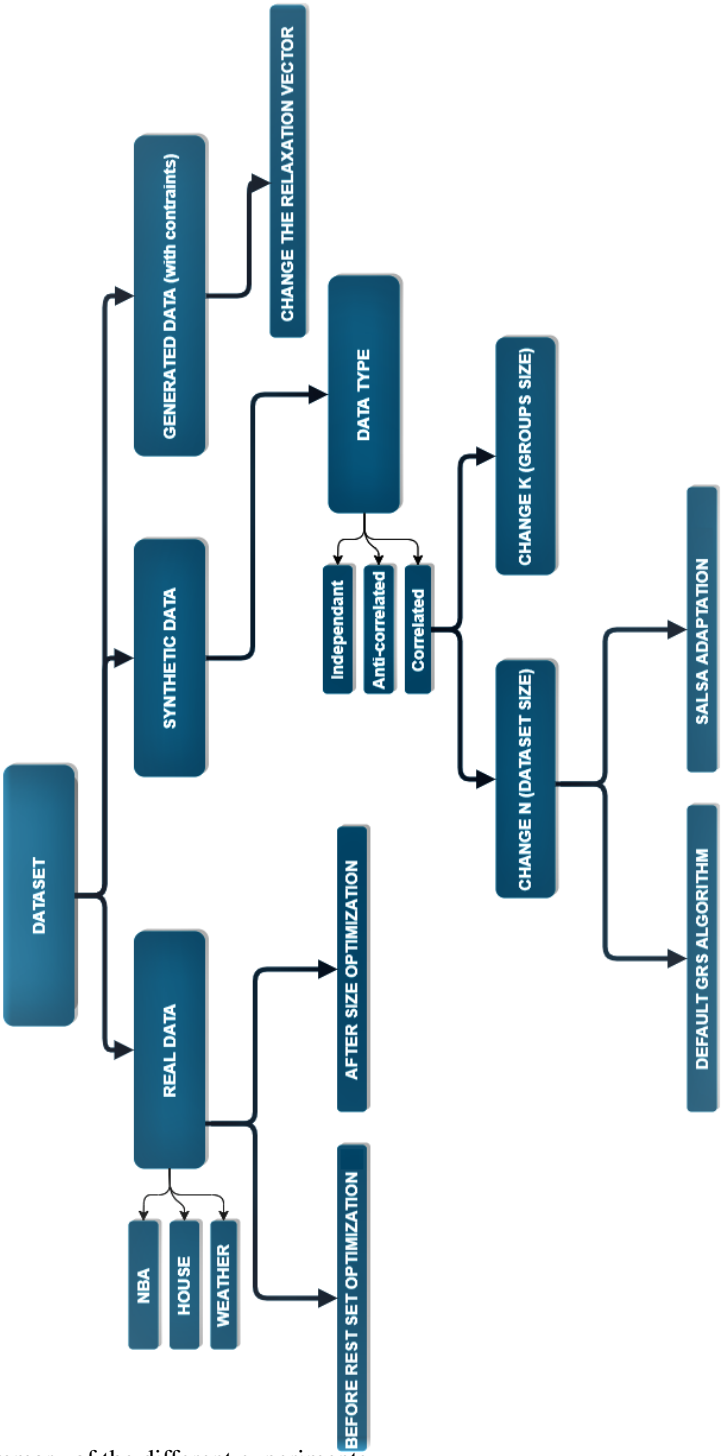

Fig. 5. Summary of the different experiments

(The choice a very low values can motivate our approach and prove the utility of getting results even if the user-defined values are small). Existing skyline researchers [14,38] use the data generator proposed in [3] with standard parameters so that results can be directly compared. As our work targets the same kind of data, we propose to follow the same methodology to evaluate our proposal. The data generator in Borzsonyi et al. (2001) generates database tuples (or points) with varying numbers of dimensions (or attributes). Tuples are generated using one of the following three value distributions: – correlated (corr): tuples, which are good in one dimension, tend to be good in other dimensions, too; – anti-correlated (anti): tuples, which are good in one dimension, are bad in at least one other dimension; –independent (indep): tuples are generated using a uniform distribution. We also include three real datasets (see Table 8 for the specification of those datasets) that are commonly used to evaluate skyline algorithms [4]: NBA (statistics of basketball players during regular seasons), HOUSE (money spent in one year by an American family for six different types of expenditures) and WEATHER (average monthly precipitation totals and elevation at over half a million sensor locations). Finally, a last test is done on generated real data using Skyline generator (randdataset). We summarize our input values (constant/variable) in table 7 and our experimental tests in figure 5. Note that in all our experiments, we make use of the aggregation function "MIN". Since this function returns less skyline groups than other functions such MAX and SUM (as shown in reference [38]). This behavior of the "MIN" function is more interesting for the relaxation purpose.

6.3. Experimental results

Case 1: Real Data
Figure 6 shows the number of relaxed skyline groups in a different data type and the execution time of the RG-SKY approach using the adapted aggregation function RG-SKY provides more groups comparing to the set G-SKY. One can observe that for the three datasets (NBA, WEATHER, HOUSE) G-SKY contains only one group.

The execution time depicted in Figure 6 and resulting from Algorithm 1 is not similar for the three datasets due to their different correlations and sizes, while this time is similar for the (correlated) NBA dataset.

After this first execution, we propose an optimized algorithm (Algorithm 2) that leads to 97.20% improvement of the naive version (i.e., Algorithm 1) in terms of execution time. This why for all the next experiments, Algoritm 2 is used.

Case 2: Synthetic Data (Correlated)

– **Date type variation** Figure 7 shows the execution time and the returned number of relaxed skyline groups in different data types: correlated, anti-correlated and independent data. As can be seen, the set G-SKY always contains one group for the three datasets compared to RG-Sky which returns more than one group, except for correlated data that returns the same number of skyline groups.
On the other hand, one can observe that independent and anti-correlated data are time consuming compared to the correlated data. We note also that our RG-SKY Salsa algorithm (Algorithm 2) is equal or less time consuming compared to the existing G-SKY naive algorithm (Except for anti-correlated data).
This is why we decide to continue our experiments only on correlated data.

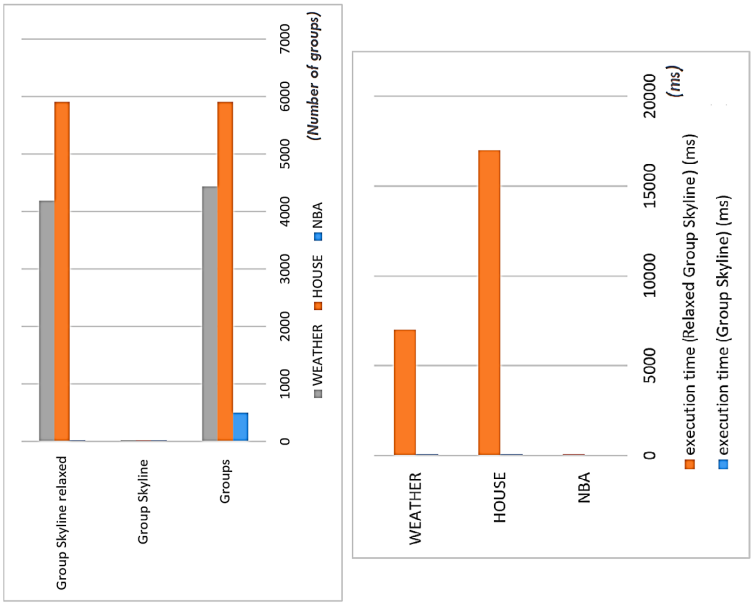

Fig. 6. Real data execution time and the number of relaxed skyline groups returned

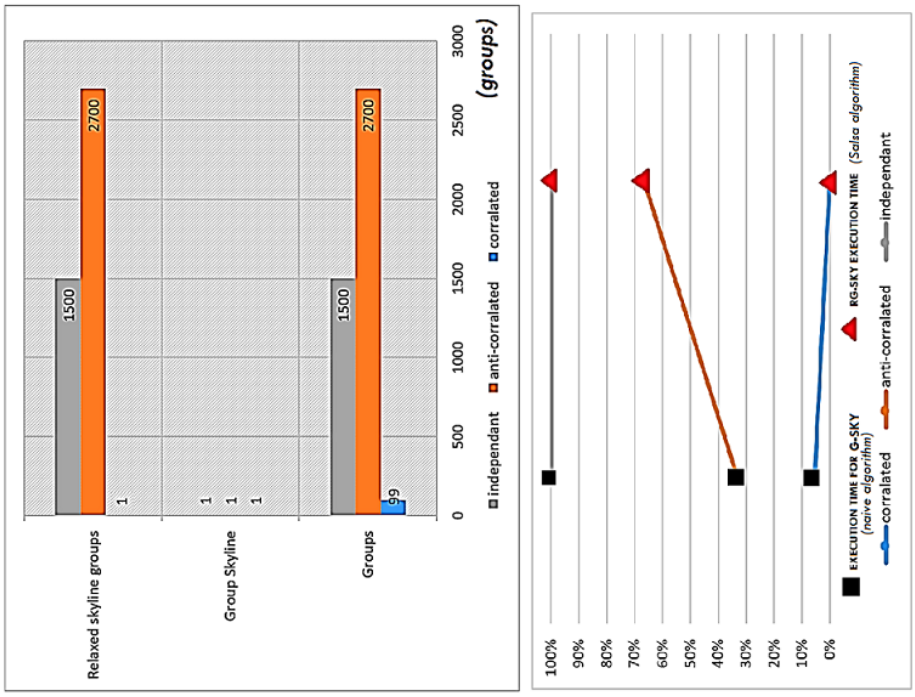

Fig. 7. Synthetic data execution time and the number of relaxed skyline groups in different data types

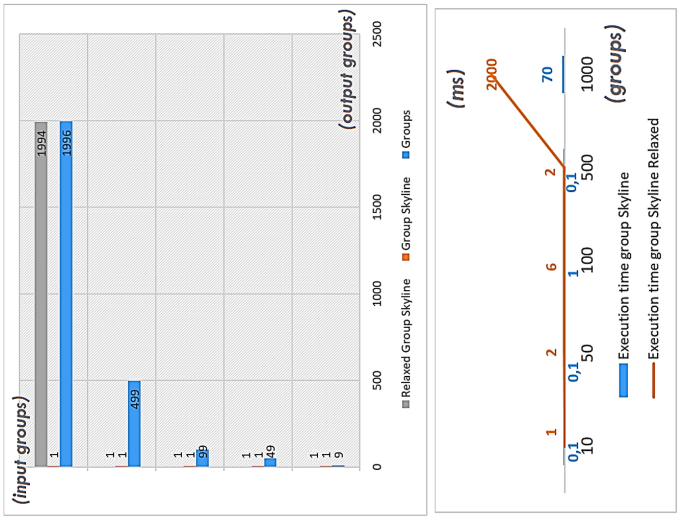

Fig. 8. Synthetic correlated data with different dataset size

– **Data size variation**
Figure 8 shows that the RG-SKY approach is very efficient. We obtain more skyline groups with almost similar time consumption when the input groups do not exceed 500 groups.

– **Group dimension variation**
In this part, we focus and vary the group dimensions (i.e., the parameter l) instead of the number of input groups (i.e. N).
The figure 9 shows that the RG-SKY approach time execution is acceptable when the number of elements (i.e. l) in the group does not exceed 4. For instance, for 100 tuples, if l=3 then the time execution for group generation: G-SKY computation and RG-SKY calculus are respectively 4754 (ms), 7 (ms), 19 (ms). While for l=5, we obtain 3612281 (ms), 3022 (ms), 242000 (ms) respectively.

Case 3: Generated Data

- **Different relaxation vector values**
For the last test, we generate groups using normalized data values in order to analyze the impact of the relaxation vector values on the RG-SKY approach. Due to data normalization, we use the same much preferred relation $MP_{\mathbb{G}_i}(\gamma 1,\gamma 2)$ for all the skyline attributes i.
Figure 10 shows two cases for the $MP_{\mathbb{G}_i}(\gamma 1,\gamma 2)$ relation:
  1. Case 1: $\gamma 1$ is fixed ($\gamma 1$=0) and $\gamma 2$ varies to increase the relaxation zone, the obtained result shows that the size of the relaxed group skyline increases when $\gamma 2$ is larger but the execution time remains acceptable.
  2. Case 2: $\gamma 1$ and $\gamma 2$ are both changing, the obtained result shows that the size of the relaxed group skyline increases similarly as the first case. One can observe that the execution time can be considered as reasonable w.r.t the numbers of the relaxed generated skyline groups.

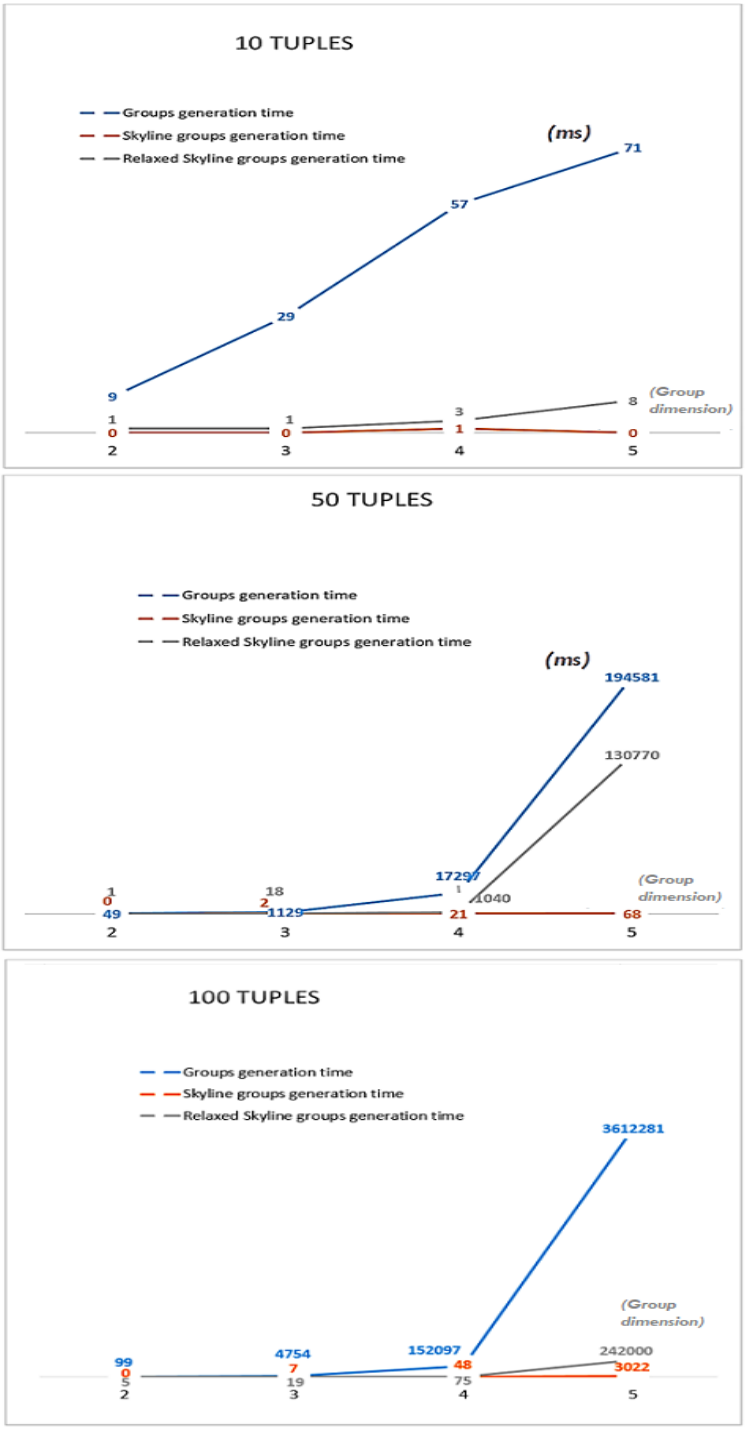

Fig. 9. Synthetic correlated data with a dynamic values of K-group (dimensions) and N-group (Number of groups)

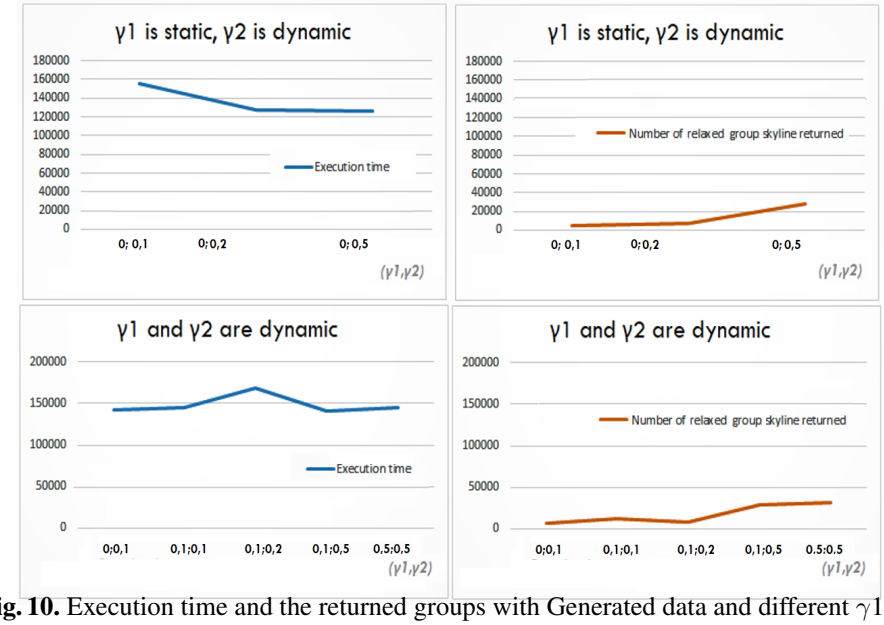

Fig. 10. Execution time and the returned groups with Generated data and different $\gamma 1$ and $\gamma 2$ values

7. Conclusion

In this paper, we addressed a new problem in the skyline community, that is, the problem of small group skylines. An approach for relaxing this kind of skyline, called RG-SKY, is discussed. It allows enlarging the group skyline at hand with interesting groups in a controlled way, and thus makes the decision easier. Moreover, to better meet the needs and expectations of the decision makers, the RG-SKY approach offers them the choice of the appropriate relaxation thanks to the different input parameters. The key concept of this approach is a particular fuzzy relation named much preferred whose semantics is user-defined. In addition, two algorithms to compute the relaxed group skyline are proposed, the first is a naive version of the RG-SKY method and the second is an optimized version using the Salsa algorithm which is used for the first time to extract groups instead of individual points in the relaxation context. The experimental study shows that the RG-SKY approach is a good alternative in terms of execution time, number of the relaxed skyline groups and user satisfaction.

As for future work, we plan to optimize the performance by taking the group generation part into consideration by eliminating groups using hierarchy methods based on the number of skyline points per group. We plan also to explore the parallel computation for the progressive relaxation generation of group skyline to optimize the time consumption of the approach, Finally, we try to generate the relaxation vector automatically taking into account the user attribute preferences.

References

1. Bartolini, I., Ciaccia, P., Patella, M.: Efficient sort-based skyline evaluation. ACM Trans. Database Syst. 33, 31:1–31:49 (2008)
2. Belkasmi, D., Hadjali, A., Azzoune, H.: On fuzzy approaches for enlarging skyline query results. Applied Soft Computing 74, 51–65 (2019)
3. Borzsony, S., Kossmann, D., Stocker, K.: The skyline operator. In: proc. 17th Inter. Conf. on data engineering. pp. 421–430. IEEE (2001)
4. Chester, S., Šidlauskas, D., Assent, I., Bøgh, K.S.: Scalable parallelization of skyline computation for multi-core processors. In: 2015 IEEE 31st Inter. Conf. on Data Engineering. pp. 1083–1094. IEEE (2015)
5. Chomicki, J., Godfrey, P., Gryz, J., Liang, D.: Skyline with presorting. In: 19th Inter. Conf. on Data Engineering. pp. 717–719 (March 2003)
6. Chung, Y.C., Su, I.F., Lee, C.: Efficient computation of combinatorial skyline queries. Information Systems 38(3), 369–387 (2013)
7. Cui, X., Dong, L.: An efficient algorithm to compute compositional skyline. In: IOP Conf. Series: Materials Science and Engineering. vol. 466, p. 012021. IOP Publishing (2018)
8. Dong, L., Liu, G., Cui, X., Li, T.: Finding group-based skyline over a data stream in the sensor network. Information 9(2), 33 (2018)
9. Dubois, D., Kerre, E., Mesiar, R., Prade, H.: Fuzzy interval analysis. In: Fundamentals of fuzzy sets, pp. 483–581. Springer (2000)
10. Godfrey, P., Shipley, R., Gryz, J.: Maximal vector computation in large data sets. In: proc. of the 31st Inter. Conf. on Very Large Data Bases. pp. 229–240. VLDB '05 (2005)
11. Goncalves, M., Tineo, L.: Fuzzy dominance skyline queries. In: Inter. Conf. on Database and Expert Systems Applications. pp. 469–478. Springer (2007)
12. Guo, X., Li, H., Wulamu, A., Xie, Y., Fu, Y.: Efficient processing of skyline group queries over a data stream. Tsinghua Science and Technology 21(1), 29–39 (2016)
13. Guo, X., Xiao, C., Ishikawa, Y.: Combination skyline queries. In: Transactions on Large-Scale Data-and Knowledge-Centered Systems VI, pp. 1–30. Springer (2012)
14. Im, H., Park, S.: Group skyline computation. Information Sciences 188, 151–169 (2012)
15. Jiang, T., Zhang, B., Lin, D., Gao, Y., Li, Q.: Incremental evaluation of top-k combinatorial metric skyline query. Knowledge-Based Systems 74, 89–105 (2015)
16. Kossmann, D., Ramsak, F., Rost, S.: Shooting stars in the sky: An online algorithm for skyline queries. In: proc. of the 28th Inter. Conf. on Very Large Data Bases. pp. 275–286. VLDB '02, VLDB Endowment (2002)
17. Li, K., Yang, Z., Xiao, G., Li, K., et al.: Progressive approaches for pareto optimal groups computation. IEEE Transactions on Knowledge and Data Engineering 31(3), 521–534 (2018)
18. Lin, M.Y., Lin, Y.L., Hsueh, S.C.: Discovering group skylines with constraints by early candidate pruning. In: Inter. Conf. on Advanced Data Mining and Appli. pp. 49–62. Springer (2017)
19. Liu, J., Xiong, L., Pei, J., Luo, J., Zhang, H.: Finding pareto optimal groups: Group-based skyline. proc. of the VLDB Endowment 8, 2086–2097 (2015)
20. Liu, J., Xiong, L., Zhang, Q., Pei, J., Luo, J.: Eclipse: Generalizing knn and skyline. arXiv preprint arXiv:1906.06314 (2019)
21. Nadouri, S., Hadjali, A., Sahnoun, Z.: Group skyline computation: An overview. In: proc. of the 36th Computer Workshop of Organizations and Information Systems and Business Intelligence Decision Making, Big Data and Data Science, INFORSID, France, May 28-31, 2018. (2018)
22. Nadouri, S., Ouhammou, Y., Sahnoun, Z., Hadjali, A.: Towards a multi-agent approach for distributed decision support systems. In: 2018 IEEE 27th Inter. Conf. on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE). pp. 72–77. IEEE (2018)
23. Nadouri, S., Sahnoun, Z., Hadjali, A.: Using g-skyline to improve decision-making. In: proc. of the 3rd Inter. Conf. on Advanced Aspects of Software Engineering, ICAASE 2018, Constantine, Algeria, December 1-2, 2018. pp. 141–150 (2018)

24. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: ACM SIGMOD Inter. Conf. on Management of Data. pp. 467–478. SIGMOD '03, ACM, New York, NY, USA (2003)
25. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: ACM SIGMOD Inter. Conf. on Management of Data. pp. 467–478. SIGMOD '03, ACM, New York, NY, USA (2003)
26. Papadias, D., Tao, Y., Fu, G., Seeger, B.: Progressive skyline computation in database systems. ACM Trans. Database Syst. 30(1), 41–82 (Mar 2005)
27. Papadias, D., Tao, Y., Fu, G., Seeger, B.: Progressive skyline computation in database systems. ACM Trans. Database Syst. 30(1), 41–82 (Mar 2005)
28. Perny, P., Roubens, M.: Fuzzy preference modeling. In: Fuzzy sets in decision analysis, operations research and statistics, pp. 3–30. Springer (1998)
29. Su, I.F., Chung, Y.C., Lee, C.: Top-k combinatorial skyline queries. In: Inter. Conf. on Database Systems for Advanced Applications. pp. 79–93. Springer (2010)
30. Tan, K.L., Eng, P.K., Ooi, B.C.: Efficient progressive skyline computation. In: 27th Inter. Conf. on Very Large Data Bases. pp. 301–310. VLDB '01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001)
31. V, N.N.: Spatial skyline query algorithms. `https://viblo.asia/p/spatial-skyline-query-algorithms`, online-access June 2021
32. Yang, Z., Zhou, X., Mei, J., Zeng, Y., Xiao, G., Pan, G.: Identifying most preferential skyline product combinations. Inter. Journal of Pattern Recognition and AI 31(11) (2017)
33. Yang, Z., Zhou, X., Zeng, Y., Zeng, F., Zhou, Y.: Identifying most preferential skyline product combinations under price promotion. In: 2016 12th Inter. Conf. on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD). pp. 1824–1828. IEEE (2016)
34. Yu, W., Liu, J., Pei, J., Xiong, L., Chen, X., Qin, Z.: Efficient contour computation of group-based skyline. IEEE Transactions on Knowledge and Data Engineering (2019)
35. Yu, W., Qin, Z., Liu, J., Xiong, L., Chen, X., Zhang, H.: Fast algorithms for pareto optimal group-based skyline. In: proc. of the 2017 ACM on Conf. on Information and Knowledge Management. pp. 417–426. ACM (2017)
36. Zadeh, L.A.: Fuzzy sets. Information and control 8(3), 338–353 (1965)
37. Zhang, K., Gao, H., Han, X., Wang, J.: Finding k-dominant g-skyline groups on high dimensional data. IEEE Access 6, 58521–58531 (2018)
38. Zhang, N., Li, C., Hassan, N., Rajasekaran, S., Das, G.: On skyline groups. IEEE Transactions on Knowledge and Data Engineering 26, 942–956 (2014)
39. Zhou, X., Li, K., Yang, Z., Li, K.: Finding optimal skyline product combinations under price promotion. IEEE Transactions on Knowledge and Data Engineering 31(1), 138–151 (2018)
40. Zhu, H., Li, X., Liu, Q., Xu, Z.: Top-k dominating queries on skyline groups. IEEE Transactions on Knowledge and Data Engineering pp. 1–1 (2019)
41. Zhu, H., Li, X., Liu, Q., Zhu, H.: Computing skyline groups: an experimental evaluation. Tsinghua Science and Technology 24(2), 171–182 (April 2019)
42. Zhu, H., Zhu, P., Li, X., Liu, Q.: Computing skyline groups: An experimental evaluation. In: proc. of the ACM Turing 50th Celebration Conf. - China. pp. 48:1–48:6. ACM TUR-C '17, ACM, New York, NY, USA (2017)
43. Zhu, H., Zhu, P., Li, X., Liu, Q.: Top-k skyline groups queries. In: EDBT. pp. 442–445 (2017)
44. Zhu, H., Zhu, P., Li, X., Liu, Q., Xun, P.: Parallelization of group-based skyline computation for multi-core processors. Concurrency and Computation: Practice and Experience 29(18) (2017)

Sana Nadouri is a PhD student in Computer science at the university of Constantine 2, Constantine, Algeria and the National Engineering School for Mechanics and Aerotechnics (ISAE-ENSMA), Poitiers, France. She is a member of the Laboratory LIRE and the

laboratory LIAS. The areas of her scientific interest focus on Artificial intelligence, decision making and data extraction and optimization. The complete list of her publications is available in http://www.lias-lab.fr/members/sananadouri.

Allel Hadjali is currently Full Professor in Computer Science at the National Engineering School for Mechanics and Aerotechnics (ISAE-ENSMA), Poitiers, France. He is a member of the Data and Model Engineering research team of the Laboratory of Computer Science and Automatic Control for Systems (LIAS). His research interests are Massive Data Exploitation and Analysis, Extraction, Recommendation and Explainabiliy in Learning Machine Models. The complete list of his publications is available in http://www.lias-lab.fr/members/allelhadjali.

Zaidi Sahnoun got his Engineer degree from the university of Constantine, Algeria and the Master and PhD degrees from RPI, Troy N.Y, USA. He has held many scientific positions (head of the Computer Science Department, director of the LIRE laboratory and dean of the faculty of Computer Science and Information Technology at Constantine 2 University). Currently, Professor SAHNOUN is retired as a Full Professor and he is a member of the LIRE Laboratory. His main research interests are Software and Knowledge Engineering, Multi Agent Systems and Artificial Intelligence.

Received: October 20, 2021; Accepted: April 15, 2022.