Towards Optimal Two-Dimensional Indexing for Constraint Databases

E. Bertino B. Catania

Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Via Comelico 39/41, 20135 Milano, Italy
e-mail: {bertino,catania}@dsi.unimi.it

B. Shidlovsky

Rank Xerox Research Center, Grenoble Laboratory, 6, chemin de Maupertuis, 38240 Meylan, France
e-mail: chidlovskii@grenoble.rxrc.xerox.com

Abstract

We address the problem of indexing conjunctions of linear constraints with two variables. We show how containment and intersection selection problems for constraint databases can be reduced to the point location problem by using a dual transformation. The proposed representation is then used to develop an efficient secondary storage solution for one important particular indexing case.

Key words: databases, data structures, computational geometry.

1 Introduction

Constraint programming is very attractive from a database point of view because it is completely declarative and because often constraints represent the communication language of several high-level applications [5]. Constraints can be added to relational database systems at different levels. At the data level, quantifier free conjunctions of constraints finitely represent possibly infinite sets of relational tuples. Thus, constraints are a powerful mechanism for modeling spatial and temporal concepts, where often infinite information should be represented. For example, the conjunction $1 \leq X \leq 2 \ \wedge \ 2 \leq Y \leq 3$, where X and Y are real variables, represents the infinite set of tuples having a real number between 1 and 2 as value for the X attribute and a real number between 2 and 3 as value for the Y attribute. Thus, the conjunction identifies a rectangle. Such conjunction of constraints is called *generalized tuple* and the possibly infinite set of relational tuples it represents is called *extension* of the generalized tuple. A finite set of generalized tuples is called *generalized relation* [5]. Different logical theories can be used to model different information. At the query language level, constraints increase the expressive power of simple relational languages by allowing mathematical computations. The integration of constraints in existing query languages introduces several issues. In particular, constraint query languages should preserve all the good features of relational languages. For example, they should be closed and bottom-up evaluable [5]; moreover they should also preserve efficiency. Therefore, new data structures should be defined for querying and updating constraint databases, with worst case time and space comparable to those of data structures for relational databases [3].

At least two constraint language features should be supported by index structures in constraint databases:

- **ALL selection**. It retrieves, from a given generalized relation r, all generalized tuples whose extension is contained in the extension of a given generalized tuple specified in the query, called *query generalized tuple*. If the extension of a generalized tuple t is contained in the extension of a query generalized tuple q, we denote this fact by $All(q,t)$. Given a generalized relation r and a query generalized tuple q, we denote by $ALL(q,r)$ the set $\{t|t \in r, All(q,t)\}$.
- **EXIST selection**. It retrieves, from a given generalized relation r, all generalized tuples whose extension has a non-empty intersection with the extension of a query generalized tuple q. If the extensions of t and q have a non-empty intersection, we denote this fact by $Exist(q,t)$. Given a generalized relation r and a query generalized tuple q, we denote by $EXIST(q,r)$ the set $\{t|t \in r, Exist(q,t)\}$. Since $ALL(q,r) \subseteq EXIST(q,r)$, it is more convenient to define the query $EXIST_e(q,r) = EXIST(q,r) \setminus ALL(q,r)$ and therefore $EXIST_e(q,r) \cap ALL(q,r) = \{\}$. In a similar way, we denote by $Exist_e(q,t)$ the fact $Exist(q,t) \wedge \neg All(q,t)$. [1]

All existing indexing techniques for constraint databases only support $EXIST$ selection and assume that index values are intervals [1,6,9]. Some other solutions are based on approximation of the extension of generalized tuples [2,8] and therefore cannot provide a good worst-case performance. The aim of this paper is to propose a simple indexing mechanism for both $EXIST_e$ and ALL selections against two-dimensional linear constraints. [2] Constraints are assumed to be represented by using the linear polynomial constraint theory

[1] Note that $All, Exist_e$, and $Exist$ are predicates.

[2] Results of $EXIST$ selections can be obtained by simply merging ALL and $EXIST_e$ selection results.

[5] and the query generalized tuple is assumed to be a half-plane (also called *query half-plane*).

The main result presented in the paper shows how ALL and $EXIST_e$ selection problems can be uniformly reduced to the point location problem [7] by using the concept of geometric duality [4]. With the dual transformation, a query half-plane from the primal plane is transformed into a point in the dual plane meanwhile a generalized tuple is transformed into a pair of disjoint and convex domains, whose boundaries are respectively an upward open and a downward open polygon. [3]

If the angular coefficient of the line associated with a query half-plane belongs to a fixed set of cardinality k, a solution in external storage can be provided such that it occupies $O(k\ N/B)$ pages, ALL and $EXIST_e$ selections are performed in $O(\log_B N/B + T/B)$ time and updates are performed in $O(k \log_B N/B)$ time, where B is the number of generalized tuples stored in one page, N/B is the number of pages required to store N generalized tuples, and T/B is the number of pages required to store the T generalized tuples representing the query result. To the best of our knowledge, this is the first approach with low complexity to the problem of indexing two-dimensional linear constraints.

The paper is organized as follows. Section 2 introduces assumptions and preliminary notations. The reduction of the selection problems to the point location problem is presented in Section 3. Section 4 proposes complexity results for indexing linear constraints. Finally, Section 5 presents some concluding remarks.

2 Preliminaries

The point location problem is defined as follows [7]. Let $F = \{F_1, ..., F_n\}$ where each F_i is a point set in the plane and let p be a point in the plane. The problem of locating point p with respect to F is the problem of determining all F_i such that $p \in F_i$.

In the remainder of this paper, we make the following assumptions:

- ALL and $EXIST_e$ selections are performed with respect to two real variables x and y. We assume that variable x represents the horizontal axis.

[3] An open polygon is a finite chain of line segments with the first and last segments being half-lines (assuming the segments to be ordered with respect to the first coordinate of their left vertex). An open polygon is upward (downward) open if the projections of both half-lines on the vertical axis approach $+\infty$ ($-\infty$).

The query generalized tuple has form $y \geq ax + b$ (also called *down-query*) or $y \leq ax + b$ (also called *up-query*) and represents a half-plane. We assume that $a, b \in \mathbb{R}$.

- The projection of any generalized tuple on x and y is a generalized tuple of the form $t \equiv C_1 \wedge ... \wedge C_m$,[4] where each C_i, $i = 1, ..., m$, is one of the following constraints:[5]
  - *down-constraint*: $C_i \equiv y \geq a_i x + b_i$. A generalized tuple containing only down-constraints is called *down-generalized tuple* and its extension is a convex domain having as boundary an upward open polygon.
  - *up-constraint*: $C_i \equiv y \leq a_i x + b_i$. A generalized tuple containing only up-constraints is called *up-generalized tuple* and its extension is a convex domain having as boundary a downward open polygon.
  - *eq-constraint*: $C_i \equiv y = a_i x + b_i$. We can reduce this constraint to the previous ones, by substitution with the conjunction of constraints $y \geq a_i x + b_i \wedge y \leq a_i x + b_i$.

  We assume that $a_i, b_i \in \mathbb{R}$, $i = 1, ..., m$. We denote by $e(t)$ the extension of a generalized tuple t and by $p(t)$ the boundary of such extension (considered as a set of points). Similarly, the line associated with a constraint C is denoted by $p(C)$. Given a constraint C and a generalized tuple t, we say that C is *non-redundant* with respect to t if the tuple $t' \equiv t \wedge C$ is not equivalent to t (thus, $e(t)$ and $e(t')$ do not coincide). Otherwise, C is *redundant* with respect to t.

  A generalized tuple is *satisfiable* if its extension is not empty, i.e., the generalized tuple is satisfiable in the domain associated with the chosen theory. In the remainder of the paper, we consider only satisfiable generalized tuples. Satisfiable generalized tuples of the type described above and relations containing only such generalized tuples are called *regular*.

The dual transformation T we use is similar to the one defined in [4]. The coordinates of a point in the dual plane are denoted by x^d and y^d; x^d represents the horizontal axis. Let l be a line $y = ax + b$ in the *primal plane* $\mathcal{P}$. In the *dual plane* $\mathcal{D}$, l is transformed into point $T(l) \equiv (x^d = a, y^d = b)$. Due the same transformation, a point p in $\mathcal{P}$ is transformed into a line in $\mathcal{D}$. If the point p is given by the intersection of two lines l_1 and l_2 in $\mathcal{P}$, then the line $T(p)$ is required to connect points $T(l_1)$ and $T(l_2)$ in $\mathcal{D}$. It is easy to show that if the point p in $\mathcal{P}$ is $(x = a, y = b)$, the corresponding dual line $T(p)$ is $y^d = -ax^d + b$.

---

[4] The symbol $\equiv$ is used to denote syntactic equality.

[5] Let t be a generalized tuple defined on variables $x_1, ..., x_n$. The projection of t on variables $x_{i_1}, ..., x_{i_k}$, $i_j \in \{1, ..., n\}$, $j = 1, ..., k$, is the formula obtained by existentially quantifying t with respect to variables $\tilde{x} \equiv \{x_1, ..., x_n\} \setminus \{x_{i_1}, ..., x_{i_k}\}$ (thus obtaining the formula $\exists \tilde{x}\ t$) and then removing the quantifier by applying a quantifier elimination algorithm. This is a typical operation in constraint databases.

The dual transformation cannot be applied to vertical lines. Therefore, we assume that regular generalized tuples do not contain constraints like $x \leq a$ or $x \geq a$ (called *vertical line constraints*). If some generalized tuples contain such constraints, we can always rotate the plane associated with the generalized tuples in such a way that no rotated tuple contains vertical line constraints. For simplicity, we still denote the coordinates of the rotated plane by x and y.

3 Dual representation for regular generalized tuples

In this section, we extend the representation in the dual plane proposed for lines to generalized tuples. To this purpose, we first consider down- and up-generalized tuples. Then, we extend such analysis to the case of arbitrary regular generalized tuples.

3.1 Regular down-generalized tuples

Let t be a regular down-generalized tuple $t \equiv C_1 \wedge ... \wedge C_m$, $C_i \equiv y \geq a_i x + b_i$, $i = 1, ..., m$. Without loss of generality, we assume that all C_i's are not redundant with respect to t and sorted in the increasing order of a_i, $i = 1, ..., m$.

A regular down-generalized tuple t from the primal plane $\mathcal{P}$ is transformed in the dual plane $\mathcal{D}$ into a convex set of points (a convex domain), whose boundary is a downward open polygon. We construct the polygon in two steps. First we consider all points $T(p(C_i))$, $i = 1, ..., m$, and construct the chain of line segments $S_i, i = 1, \ldots, m-1$, where S_i connects points $T(p(C_i))$ and $T(p(C_{i+1}))$. The set of points belonging to such chain is denoted by $h(t)$. Basically, points composing a segment S_i correspond to all lines in $\mathcal{P}$ obtained by an anti-clockwise rotation of line $p(C_i)$ to match line $p(C_{i+1})$ around the point where $p(C_i)$ and $p(C_{i+1})$ intersect.

Now consider a line l in the primal plane. If l *touches* the extension $e(t)$ of the down-generalized tuple t, that is, l intersects the boundary of $e(t)$ but not its interior, there are two cases: (i) a vertex of $p(t)$ lies on l; (ii) an edge of $p(t)$ lies on l. In the first case, due to the previous discussion, in the dual plane $T(l)$ belongs to segment S_i, for some $i \in \{1, ..., m-1\}$. In the second case, $T(l)$ coincides with $T(p(C_j))$, for some $j \in \{1, ..., m\}$. Therefore, in both cases $T(l)$ belongs to $h(t)$. The vice versa also holds. Based on these considerations, we can state the following lemma.

Lemma 1 *Let t be a regular down-generalized tuple. Let l be a line in the primal plane $\mathcal{P}$. Then, $T(l) \in h(t)$ iff l touches $e(t)$, that is, $e(l) \cap p(t)$ is a* *non-empty convex set (either a point or a segment).*[6]

To enable processing ALL and $EXIST_e$ selections, we generate a downward open polygon $h^*(t)$ by adding to $h(t)$ two vertical downward oriented half-lines, $x^d = a_1 \wedge y^d \leq b_1$ and $x^d = a_m \wedge y^d \leq b_m$. We denote by $D^-(t)$ the domain in $\mathcal{D}$ *under* the polygon $h^*(t)$. That is, for any point $(x^d = x_1^d, y^d = y_1^d)$ in $D^-(t)$, $a_1 < x_1^d < a_m$ holds and there exists a point $(x^d = x_1^d, y^d = y_2^d) \in h^*(t)$ such that $y_2^d > y_1^d$. Also, we denote by $D^0(t)$ the domain which is the complement of $D^-(t) \cup h^*(t)$ in $\mathcal{D}$. Note that the points contained in $h^*(t) \setminus h(t)$ correspond in the primal plane to lines which are parallel to either $p(C_1)$ or $p(C_m)$ and which do not intersect $e(t)$. Each point $(x^d = x_1^d, y^d = y_1^d) \in D^-(t)$ represents in the primal plane a line not intersecting $p(t)$; such line is parallel to a line $y = x_1^d x + y_2^d$ which does not intersect the interior of $e(t)$ (since point $(x^d = x_1^d, y^d = y_2^d)$ belongs to $h^*(t)$) and satisfies the condition $y_2^d > y_1^d$. Since $p(t)$ is an upward open polygon, this means that $e(t)$ is contained in the half-plane $y \geq x_1^d x + y_1^d$. Finally, D^0 represents in the primal plane all lines intersecting $e(t)$ in at least one point not belonging to $p(t)$.

Using the concepts defined above, the dual transformation of a down-generalized tuple t can be defined as $T(t) \equiv D^-(t) \cup h^*(t)$. The proposed transformation preserves the redundancy relationship between a constraint and a down-generalized tuple. Consider a down-generalized tuple t. A constraint $C \equiv y \geq ax + b$ in $\mathcal{P}$ contains either the entire extension of t (thus, it is redundant with respect to t) or part of $e(t)$ (thus, it is not redundant with respect to t). Using Lemma 1 and the properties of $D^-(t)$, $D^0(t)$, $h(t)$, and $h^*(t)$, it can be shown that this redundancy relationship is transformed in the dual plane in the containment relationship between point $(x^d = a, y^d = b)$ and domain $D^-(t) \cup h^*(t)$, as stated by the following lemma. Note that an up-constraint cannot be redundant with respect to a down-generalized tuple.

Lemma 2 *Let t be a regular down-generalized tuple. A down-constraint C is redundant with respect to t iff $T(p(C)) \in D^-(t) \cup h^*(t)$. An up-constraint C is never redundant with respect to t.*

Lemma 2 gives a method for reducing ALL and $EXIST_e$ selections to the point location problem. Indeed, given a down-query q in $\mathcal{P}$ of the form $y \geq ax + b$, predicate $All(q, t)$ is satisfied iff $y \geq ax + b$ is redundant with respect to t. If it is not redundant, $Exist_e(q, t)$ is satisfied. Given an up-query q in $\mathcal{P}$ of the form $y \leq ax + b$, $All(q, t)$ is never satisfied. $Exist_e(q, t)$ is satisfied only if $p(q)$ intersects $e(t)$.

Corollary 3 *Let t be a regular down-generalized tuple and let q be a down-query. $All(q, t)$ is satisfied iff $T(p(q)) \in D^-(t) \cup h^*(t)$ and $Exist_e(q, t)$ is satisfied iff $T(p(q)) \in D^0(t)$.*

[6] If l is the line $y = ax + b$, $e(l)$ denotes the extension of constraint $y = ax + b$.

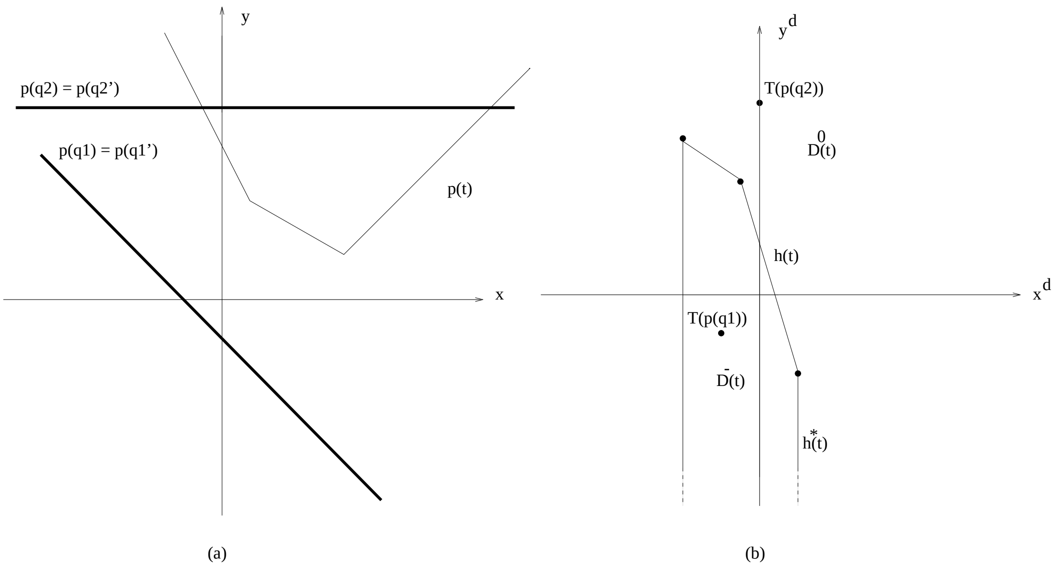

Fig. 1. A regular down-generalized tuple and two query half-planes: (a) in the primal plane $\mathcal{P}$; (b) in the dual plane $\mathcal{D}$.

Corollary 4 *Let t be a regular down-generalized tuple and let q be an up-query. $Exist_e(q,t)$ is satisfied iff $T(p(q)) \in D^0(t) \cup h(t)$. $All(q,t)$ is never satisfied.*

Example 5 *Figure 1(a) illustrates the extension of the down-generalized tuple $t \equiv y \geq -0.5x + 3 \wedge y \geq -2x + 4 \wedge y \geq x - 2$ in the primal plane $\mathcal{P}$. Figure 1(b) shows the dual representation of such generalized tuple. Consider the down-queries $q_1 \equiv y \geq -x - 1$ and $q_2 \equiv y \geq 5$. Figure 1(b) shows that $T(p(q_1)) \in D^-(t)$ and $T(p(q_2)) \in D^0(t)$. According to Corollary 3, it means that $All(q_1,t)$ and $Exist_e(q_2,t)$ are satisfied. Figure 1(a) confirms the results. If we instead consider the up-queries $q'_1 \equiv y \leq -x - 1$ and $q'_2 \equiv y \leq 5$, from Corollary 4 it follows that only the selection $Exist_e(q'_2,t)$ is satisfied. The correctness of this result can be observed in Figure 1(a).*

In the case of up-generalized tuples, results are analogous to the ones presented above. The only difference is that the open polygon $h^*(t)$ in the dual plane is upward open and it is constructed by adding to $h(t)$ two vertical upward oriented half-lines, $x^d = a_1 \wedge y^d \geq b_1$ and $x^d = a_m \wedge y^d \geq b_m$. Conditions for ALL and $EXIST_e$ selections change symmetrically by replacing $D^-(t)$ with $D^+(t)$ and prefix "down-" with prefix "up-". $D^+(t)$ denotes the domain in $\mathcal{D}$ *over* the polygon $h^*(t)$. That is, for any point $(x^d = x_1^d, y^d = y_1^d)$ in $D^+(t)$, $a_1 < x_1^d < a_m$ holds and there exists a point $(x^d = x_1^d, y^d = y_2^d) \in h^*(t)$ such that $y_2^d < y_1^d$.

Let t be a regular generalized tuple of type $C_1 \wedge ... \wedge C_m \wedge D_1 \wedge ... \wedge D_n$, such that $C_i \equiv y \geq a_i x + b_i, i = 1, \ldots, m$, and $D_j \equiv y \leq c_j x + d_j, j = 1, \ldots, n$. All C_i's and D_j's are assumed to be non redundant with respect to t and sorted in the increasing order of angular coefficients a_i, $i = 1, ..., m$, and c_j, $j = 1, ..., n$. In the following, we denote by t_{down} the down-generalized part $C_1 \wedge ... \wedge C_m$ of t and by t_{up} its up-generalized part $D_1 \wedge ... \wedge D_n$.

If $a_1 < c_n$, $p(t_{down})$ and $p(t_{up})$ intersect on the left of $e(t)$. This means that there exists an intersection point $(x = x_1, y = y_1)$ between $p(t_{down})$ and $p(t_{up})$ such that, for all points $(x = x_2, y = y_2)$ belonging to $e(t)$, $x_1 \leq x_2$ holds. Therefore, such point is the leftmost vertex of $p(t)$ in $\mathcal{P}$. Similarly, $p(t_{down})$ and $p(t_{up})$ intersect on the right if $a_m > c_1$ and the intersection gives the rightmost vertex of $p(t)$ in $\mathcal{P}$. If t contains at least one up-constraint and at least one down-constraint, $p(t)$ has at least one of these extreme vertices. Without loss of generality, in the following we consider what happens to redundancy conditions if a left intersection exists. For the right intersection, the discussion is analogous.

The presence of the leftmost vertex in $p(t)$ makes redundant with respect to t some constraints which are not redundant with respect to t_{down} and t_{up}. These constraints are called *left redundant* with respect to t. Let $LV(t)$ be the set of constraints that are left redundant with respect to t and let $LLV(t) = \{p(C) \mid C \in LV(t)\}$. Any line $l \in LLV(t)$ intersects lines $p(C_1)$ and $p(D_n)$, which define the leftmost vertex, on the left of this vertex (see for example line l in Figure 2). Let $y^d = ex^d + f$ be the line in $\mathcal{D}$ defined by points $T(p(C_1)) \equiv (x^d = a_1, y^d = b_1)$ and $T(p(D_n)) \equiv (x^d = c_n, y^d = d_n)$. It can be shown that for each line $l \equiv y = ax + b \in LLV(t)$, a is lower than a_1 or greater than c_n; moreover: (i) if $a < a_1$, point $(x^d = a, y^d = b)$ satisfies the inequality $b \leq ea + f$ and the down-constraint associated with l belongs to $LV(t)$; (ii) if $a > c_n$, point $(x^d = a, y^d = b)$ satisfies the inequality $b \geq ea + f$ and the up-constraint associated with l belongs to $LV(t)$. The vice versa also holds. Therefore, $LLV(t)$ is transformed in the dual plane into two disjoint and convex domains limited by vertical lines $x^d = a_1$ and $x^d = c_n$, and the line defined by points $(x^d = a_1, y^d = b_1)$ and $(x^d = c_n, y^d = d_n)$. From the above considerations, the following result holds (for the case of the rightmost vertex of t, the results are symmetric).

Lemma 6 *Let t be a regular generalized tuple of type $C_1 \wedge ... \wedge C_m \wedge D_1 \wedge ... \wedge D_n$ such that $C_i \equiv y \geq a_i x + b_i, i = 1, \ldots, m$, and $D_j \equiv y \leq c_j x + d_j, j = 1, \ldots, n$. Assume that $a_1 < c_n$ (thus $p(t)$ admits the leftmost vertex). Let $y^d = ex^d + f$ be the line defined by points $T(p(C_1)) \equiv (x^d = a_1, y^d = b_1)$ and $T(p(D_n)) \equiv (x^d = c_n, y^d = d_n)$ in the dual plane. A down-constraint $C \equiv y \geq ax + b$ is* *left redundant with respect to t iff $a < a_1$ and $b \leq ea + f$. An up-constraint $C \equiv y \leq ax + b$ is left redundant with respect to t iff $a > c_n$ and $b \geq ea + f$.*

Using the previous result, in order to reduce the ALL and $EXIST_e$ selection problems to the point location problem, a regular generalized tuple t is transformed in the dual plane into a pair of disjoint and convex domains. The boundaries of such domains are two open polygons which we construct in two steps, as we have done for down-generalized tuples in Subsection 3.1. In the first step, we consider points $T(p(C_i)), i = 1, \ldots, m$, and $T(p(D_j)), j = 1, \ldots, n$, in the dual plane and construct two chains of segments $h_{down}(t)$ and $h_{up}(t)$ as follows:

(1) First, the chain of segments $h_{down}(t)$ is constructed as $h(t_{down})$ and the chain $h_{up}(t)$ is constructed as $h(t_{up})$.

(2) If $a_1 < c_n$, we add the half-line $x^d \leq a_1 \wedge y^d = ex^d + f$ to $h_{down}(t)$ and the half-line $x^d \geq c_n \wedge y^d = ex^d + f$ to $h_{up}(t)$, where $y^d = ex^d + f$ is the line connecting points $T(p(C_1))$ and $T(p(D_n))$.

Points satisfying $x^d < a_1 \wedge y^d = ex^d + f$ or $x^d > c_n \wedge y^d = ex^d + f$ correspond in the primal plane to lines intersecting the extension of the generalized tuple t only in its leftmost vertex.

(3) If $a_m > c_1$, we add the half-line $x^d \geq a_m \wedge y^d = e'x^d + f'$ to $h_{down}(t)$ and the half-line $x^d \leq c_1 \wedge y^d = e'x^d + f'$ to $h_{up}(t)$, where $y^d = e'x^d + f'$ is the line connecting points $T(p(C_m))$ and $T(p(D_1))$.

Points satisfying $x^d > a_m \wedge y^d = e'x^d + f'$ or $x^d < c_1 \wedge y^d = e'x^d + f'$ correspond in the primal plane to lines intersecting the extension of the generalized tuple t only in its rightmost vertex.

The following lemma generalizes Lemma 1 to the case of regular generalized tuples.

Lemma 7 *Let t be a regular generalized tuple. Let l be a line in the primal plane $\mathcal{P}$. Then, $T(l) \in h_{up}(t) \cup h_{down}(t)$ iff $e(l) \cap p(t)$ is a non-empty convex set.*

In the second step, we construct two open polygons $h^*_{down}(t)$ and $h^*_{up}(t)$, obtained from $h_{down}(t)$ and $h_{up}(t)$ by adding either two, one, or none vertical half-lines.

(1) First $h^*_{down}(t)$ is constructed as $h_{down}(t)$ and $h^*_{up}(t)$ is constructed as $h_{up}(t)$. If $p(t)$ has both the leftmost and rightmost vertices, $h^*_{down}(t)$ and $h^*_{up}(t)$ do not change in the following steps.

(2) If $p(t)$ does not admit the leftmost vertex in $\mathcal{P}$, we add the half-line $x^d = a_1 \wedge y^d \leq b_1$ to $h^*_{down}(t)$ and the half-line $x^d = c_n \wedge y^d \geq d_n$ to $h^*_{up}(t)$.

(3) If $p(t)$ does not admit the rightmost vertex, we add the half-line $x^d = a_m \wedge y^d \leq b_m$ to $h^*_{down}(t)$ and the half-line $x^d = c_1 \wedge y^d \geq d_1$ to $h^*_{up}(t)$.

We define $D^-(t)$ and $D^+(t)$ as the domains under $h^*_{down}(t)$ and over $h^*_{up}(t)$, respectively. Moreover, we define domain $D^0(t)$ as the complement of $D^-(t) \cup D^+(t) \cup h^*_{up}(t) \cup h^*_{down}(t)$ in plane $\mathcal{D}$. The dual transformation of a regular generalized tuple t can be now defined as $T(t) \equiv D^-(t) \cup h^*_{down}(t) \cup D^+(t) \cup h^*_{up}(t)$. The polygons $h^*_{down}(t)$ and $h^*_{up}(t)$, constructed as above, satisfy the following property.

Proposition 8 *Let t be a regular generalized tuple. Then, $h^*_{down}(t)$ is under $h^*_{up}(t)$.* [7]

Proof. We prove the lemma by contradiction. Let t be a regular generalized tuple. Assume that there exists a vertical line $q \equiv x^d = a$ in $\mathcal{D}$ such that q intersects $h^*_{up}(t)$ and $h^*_{down}(t)$ in points $(x^d = a, y^d = y^d_1)$ and $(x^d = a, y^d = y^d_2)$ and $y^d_1 < y^d_2$. In this case, there exists a point $(x^d = a, y^d = b)$ such that $y^d_1 < b < y^d_2$ in $\mathcal{D}$, which belongs to both $D^-(t)$ and $D^+(t)$. Due to results presented in Subsection 3.1 and Lemma 6, this means that in $\mathcal{P}$ both half-planes $y \leq ax + b$ and $y \geq ax + b$ contain $e(t)$ and line $y = ax + b$ does not touch it (Lemma 7). As we have a contradiction here, $h^*_{up}(t)$ is always over $h^*_{down}(t)$. $\square$

From the previous result it follows that $D^-(t)$ and $D^+(t)$ are disjoint domains and $h^*_{up}(t)$ and $h^*_{down}(t)$ are possibly touching open polygons. From Lemma 6 and Proposition 8 the following important result follows.

Proposition 9 *Let t be a regular generalized tuple. Let $q(\theta)$ be a query generalized tuple $y \ \theta \ ax + b$, where $\theta \in \{\geq, \leq\}$. Then:*

- *$All(q(\geq), t)$ iff $T(q(\geq)) \in D^-(t) \cup h^*_{down}(t)$;*
- *$All(q(\leq), t)$ iff $T(q(\leq)) \in D^+(t) \cup h^*_{up}(t)$;*
- *$Exist_e(q(\geq), t)$ iff $T(q(\geq)) \in D^0(t) \cup h_{up}(t)$;*
- *$Exist_e(q(\leq), t)$ iff $T(q(\leq)) \in D^0(t) \cup h_{down}(t)$.*

Example 10 *Figure 2(a) illustrates the regular generalized tuple $t \equiv y \geq -0.5x + 3 \wedge y \geq -2x + 4 \wedge y \geq x - 2 \wedge$ $y \leq x + 3 \wedge y \leq -x + 6$ in the primal plane $\mathcal{P}$ and Figure 2(b) shows its dual representation. Given the down-queries $q_1 \equiv y \geq -x - 1$, $q_2 \equiv y \geq 5$, $q_3 \equiv y \geq 4.5$, $q_4 \equiv y \geq x$, we can see in Figure 2(b) that $T(p(q_1)) \in D^-(t)$, $T(p(q_2)) \in D^+(t)$, $T(p(q_3)) \in h_{up}(t)$ and $T(p(q_4)) \in D^0(t)$. It follows from Proposition 9 that $All(q_1, t)$, $Exist_e(q_3, t)$, and $Exist_e(q_4, t)$ are satisfied. Figure 2(a) shows that this result is correct. If we consider the up-queries $q'_1 \equiv y \leq -x - 1$, $q'_2 \equiv y \leq 5$, $q'_3 \equiv y \leq 4.5$,*

[7] This means that if a vertical line $x^d = a$ in $\mathcal{D}$ intersects $h^*_{up}(t)$ and $h^*_{down}(t)$ in points $(x^d = a, y^d = y^d_1)$ and $(x^d = a, y^d = y^d_2)$, then $y^d_1 \geq y^d_2$.

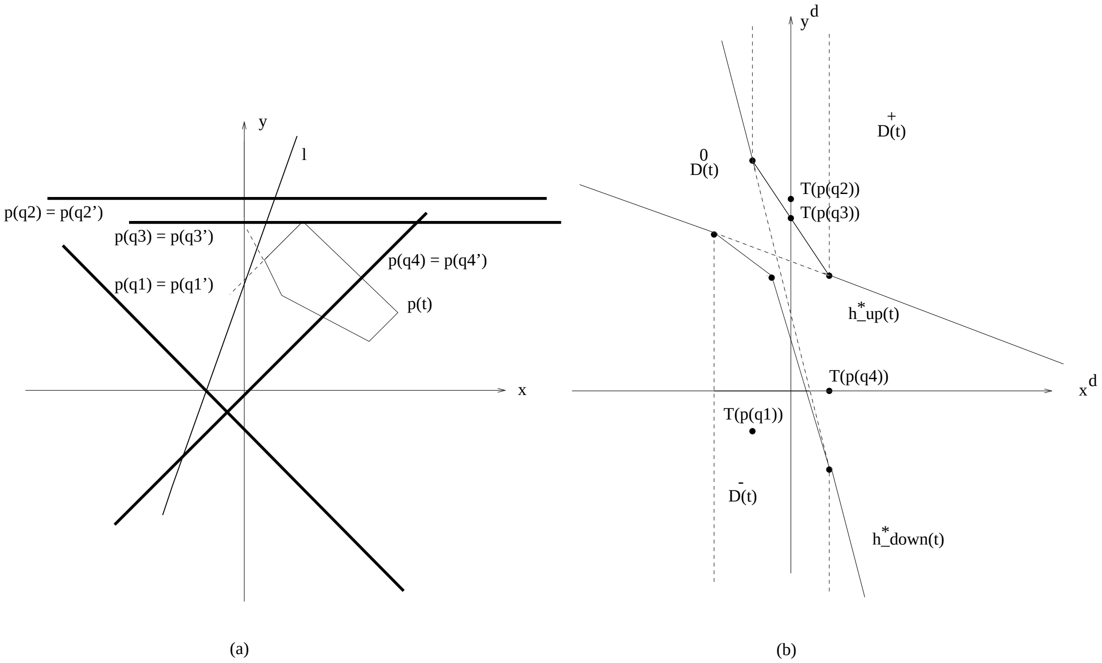

Fig. 2. A regular generalized tuple and some query half-planes: (a) in the primal plane $\mathcal{P}$; (b) in the dual plane $\mathcal{D}$.

and $q_4 \equiv y \leq x$*, from Proposition 9 it follows that* $All(q_2', t)$*,* $All(q_3', t)$*, and* $Exist_e(q_4', t)$ *are satisfied. Figure 2(a) confirms the results.*

4 Index structure

Suppose that the angular coefficients of the lines associated with query half-planes are known and given by a set S. In such a case, ALL and $EXIST_e$ selections can be efficiently reduced to the 1-dimensional interval management problem for which multiple indexing techniques were proposed, in both main memory [4,7] and secondary storage [1].

The reduction is based on the following consideration. For any tuple t, a vertical line $x^d = a$ in plane $\mathcal{D}$ can intersect both $h_{down}(t)$ and $h_{up}(t)$, or one of them, or none of them. In the first (most general) case, the line is split into three open intervals $]-\infty, y_1^d[,]y_1^d, y_2^d[$ and $]y_2^d, +\infty[$, where y_1^d and y_2^d are intersections of the line with $h_{down}(t)$ and $h_{up}(t)$, respectively. By Proposition 9, if the query generalized tuple $q(\theta)$ is given by $y\ \theta\ ax + b, a \in S$, and value b belongs to the interval $]-\infty, y_1^d]$, then predicate $All(q(\geq), t)$ is satisfied. If $b \in [y_2^d, +\infty[$, or $b \in [y_1^d, y_2^d[$, or $b \in]y_1^d, y_2^d]$, predicates $All(q(\leq), t)$, $Exist_e(q(\leq), t)$, and $Exist_e(q(\geq), t)$ are satisfied, respectively.

Thus, to perform selections against a set of regular generalized tuples, it is sufficient to maintain three 1-dimensional interval sets for each value in set

S. Management of 1-dimensional intervals is a classic problem from computational geometry [7]. An optimal solution to the problem in secondary storage has been recently proposed in [1]. It requires linear space and logarithmic time for query and update operations applied on a set of N intervals. This proves the following theorem.

Theorem 11 *Let r be a regular generalized relation containing N regular generalized tuples. Let q be a query half-plane. Let T be the cardinality of the set $ALL(q,r)$ ($EXIST_e(q,r)$). If the angular coefficient of $p(q)$ is contained in a predefined set of cardinality k, there is an indexing structure for storing r in $O(k\ N/B)$ pages such that $ALL(q,r)$ and $EXIST_e(q,r)$ selections are performed in $O(\log_B N/B + T/B)$ time and generalized tuple update operations are performed in $O(k\ \log_B N/B)$ time.*

5 Concluding remarks

The paper has proposed a geometric representation for regular two-dimensional linear constraints and has shown how selection problems for constraint databases can be reduced to geometric point location problems with respect to this dual representation. The representation has then be used to show that, if the angular coefficient of the line associated with a query half-plane belongs to a predefined set, an optimal secondary storage solution to the indexing problem for ALL and $EXIST_e$ selections (and therefore also for the $EXIST$ selection) exists.

References

[1] L. Arge and J.S. Vitter. Optimal Dynamic Interval Management in External Memory. In *Proc. of the Int. Conf. on Foundations of Computer Science*, pages 560-569, 1996.

[2] A. Brodsky, C. Lassez, J.L. Lassez, and M. Maher. Separability of Polyhedra and a New Approach to Spatial Storage. In *Proc. of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 54-65, 1995.

[3] D. Comer. The ubiquitous B-tree. *Computing Surveys*, 11(2):121-138, 1979.

[4] H. Edelsbrunner. Algorithms in Combinatorial Geometry. *EATCS Monographs on Theoretical Computer Science*, Vol. 10, 1987.

[5] P.C. Kanellakis, G.M. Kuper, and P.Z. Revesz. Constraint Query Languages. *Journal of Computer and System Sciences*, 51(1):26–52, 1995.

[6] P.C. Kanellakis, S. Ramaswamy, D.E. Vengroff, and J.S. Vitter. Indexing for Data Models with Constraints and Classes. In *Proc. of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 233–243, 1993.

[7] F.P. Preparata and M.I. Shamos. *Computational Geometry - an Introduction*, Springer Verlag, New York, 1985.

[8] D. Srivastava. Subsumption and Indexing in Constraint Query Languages with Linear Arithmetic Constraints. *Annals of Mathematics and Artificial Intelligence*, 8(3-4):315–343, 1993.

[9] S. Ramaswamy. Efficient Indexing for Constraints and Temporal Databases. In *Proc. of the Sixth Int. Conf. on Database Theory*, pages 419-431, 1997.