# Application of geometrical approaches to Information Retrieval

© Vasyl Tereshchenko

National Taras Shevchenko University of Kyiv
vtereshch@gmail.com

## Abstract

In this paper we propose an idea of transformation relational database problems to computational geometry problems to develop more efficient algorithms for discovering useful information from databases. We consider in detail relational algebra operations - the base of relational language foundation - and give adequate geometrical interpretation for each of them.

## 1 Introduction

Over the past ten years the relational database management systems (DBMS) have become wide applicable in different areas such as automated design system, CAE system, geographic information system, office information system and so on. However, the relational database management systems have limited capacity from the object's modeling viewpoint. That makes the DBMS to be non-applicable for the complicated specialized applications. Also, the recent progress of communication and network technologies makes it easy to accumulate a large collection of unstructured or semi-structured texts data [2, 5, 6]. In this context, the problem of searching more efficient algorithms to discover useful information from large non-structured databases that differs from existent information retrieving methods is a point of big interest [7, 4]. The work [3] is worth to be mentioned, since it is devoted to problem of discovering data in large semistructured text collections.

The paper proposes an algorithm based on one of the computational geometry methods that is called the regional search algorithm and speeds up search substantially. So nomogenously the subject about the possibility to transform relational database problems to computational geometry problems has been occurred taking into account a high efficient of geometric algorithms.

## 2 The geometrical approach to information retrieval

**Definition.** Problem $A$ are transduced into Problem $B$, if:

1. The input data for Problem $A$ are transduced into corresponding input data for Problem $B$.

2. The Problem $B$ is solved.

3. The result of Problem $B$ resolving is transduced into correct result for Problem $A$.

**Theorem.** The search problems in relational database are transformed to computational geometry search problems in time $O(N)$.

*Proof.* To prove the theorem, it is necessary to prove the fulfillment of three conditions mentioned above. To this end, let us formalize input data sets of the relational database search problem in terms and concepts of the geometric search problem, and per contra, results of geometric search problem solution interpreted in terms of databases.

Let each tuple of relation $R$ put in accordance to some point (or IOW $n$-plex) of geometric space $E_R$. Let each attribute of relation $R$ put in accordance to some coordinate axis in the following way: axis value area is defined by domain, the attribute is specified under so that value of each tuple element corresponds to some coordinate value of corresponding space point. Such a correspondence is "one-one". Ex facte, input data for relational database problems are transformed into corresponding input data for computational geometry problem in time $O(N)$ and the received computational geometry problem solution is transformed into correct solution for relational database problem in time $O(N)$ also. Let us consider the main operations of relational algebra that is the base for relational languages creation. And by using examples of relational algebra search queries, we proved their geometrical realization (condition 2), and hence, the transformation of two classes of problems, mentioned above.

**Selection** ($S = \sigma_{predicate}(R)$).

Selection is a unary operation. The result of the selection is a new relation $S$ containing only those tuples of the input relation $R$ that holds the specified condition (predicate). Let the relation $R$ of the relational database put in accordance to the subspace $E_R$ of the space $E^n$. As it was mentioned above, the rank d of the relation R defines the dimension of the corresponding subspace $E_R$, Figure 1.

Predicate in the selection operation defines some domain (plane of the rank $k < d$). We are interested in all those points of the subspace $E_R$ that lie within the defined domain. Thereby, the predicate determines the search region in the subspace $E_R$, and under the geometric interpretation the result of the selection operation is the query about the points set of the subspace $E_R$ that lie within the queried region. Thus, the regional search corresponds to the selection operation. There were proposed several solutions of the regional search. Among them, the algorithm based on the orthogonal range tree

**Proceedings of the Spring Researcher's Colloquium on Database and Information Systems, Moscow, Russia, 2011**

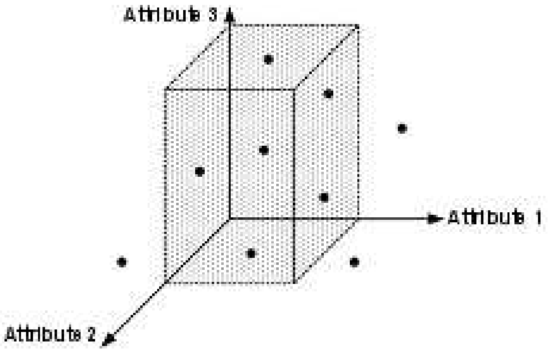

Figure 1:

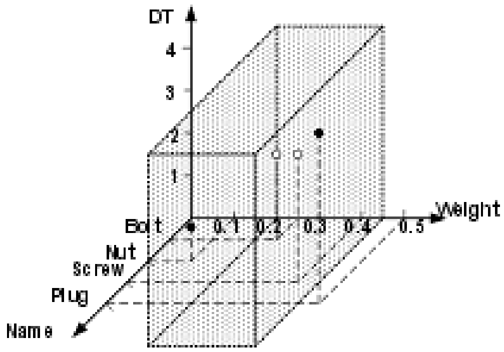

Figure 2:

method and described in Preparata and Shamos [1] is worth to be mentioned. The very algorithm uses a data structure called the orthogonal range tree that requires $O(log^{d-1}N)$ time per query, $O(Nlog^{d-1}N)$ space and $O(Nlog^{d-1}N)$ preprocessing time, where the $N$ is the number of points and $d$ is the space dimension. For an example let us consider the following relations:

PRODUCER (PR, Surname, City, Status)
CUSTOMER (CS, Surname, City)
DETAIL (DT, Name, Weight)
CPD ({CS, PR, DT}, Quantity, Price)

***Query 1.*** *Find out the list of all the details with the weight in range (0.2; 0.45).*

This query is composed in such a way:

$\sigma_{0.2<weight<0.45}(Detail)$

The given query has the following geometric view, Figure 2.

**Projection** ($S = \Pi_{atr.1,...,atr.n}(R)$).

A projection is also a unary operation. It determines a new relation $S$ that encloses a vertical subset (i.e., an attributes subset) of the input relation R obtained by deriving the values of the defined attributes and by removing all duplicate tuples from the result. Ex facte, the projection operation of the relation $R$ corresponds to the projection of the points collection of the space $E_R$ over some coordinate plane $\pi$, defined by the coordinate axis $l_1, \ldots, l_n$, that correspond to the attributes $atr.1, \ldots, atr.n$ in the projection operation $\Pi_{atr.1,\ldots,atr.n}(R)$, Figure 3.

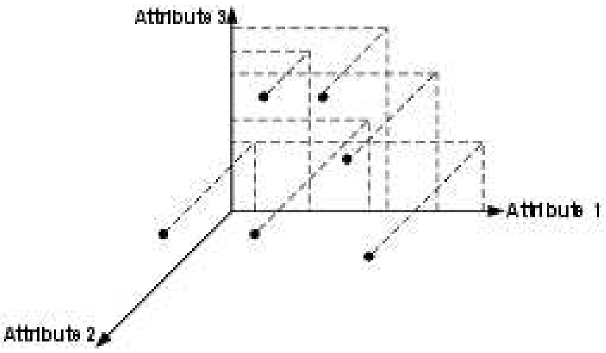

Figure 3:

Assume that we are given a point $P = (x^{(1)}, x^{(2)}, \ldots x_P^{(n)})$ over a subspace $E_R$, $dim(E_R) = d$; where $L = \{l_1, l_2, \ldots, l_d\}$ is a set of the coordinate axis of the subspace $E_R$. Then $\Pi_{atr.i}(R) \equiv pr_{li}(E_R) = \{P' | P' = pr_{li}P, \forall P \in E_R\}$= $\{P' | P' = (0, \ldots, 0, x_P^{(i)}, 0, \ldots, 0), \forall P \in E_R\}$, $\Pi_{atr.i,\ldots,atr.j}(R) \equiv pr_\pi(E_R)$=$\{P' | P' = pr_\pi P, \pi\, l_i, \ldots, l_j, \forall P \in E_R\}$= $\{P' | P' = (0, \ldots, 0, x_P^{(i)}, \ldots, x_P^{(j)}, 0, \ldots, 0), \forall P \in E_R\}$.

**Union** ($R \cup S$).

The union of two relations $R$ and $S$ with tuples $I$ and $J$ correspondingly results their concatenation by formation a new relation enclosing the maximal number of tuples $(I + J)$, if the duplicated tuples are expunged. The relations $R$ and $S$ should be a union compatible (i.e., they should have the same number of attributes with coincident domains). Let the relations $R$ and $S$ of the relational database put in accordance to the subspace $E_R$ and $E_S$ of the space $E^n$ correspondingly. In the geometric space the union compatibility corresponds to the following conditions:

1. Relations R and S have the same number of attributes ↔ corresponding subspaces $E_R$ and $E_S$ have the same dimension

$R \leftrightarrow E_R$
$S \leftrightarrow E_S => dim(E_R) = dim(E_S)$

2. Domains coincides ↔ corresponding subspaces $E_R$ and $E_S$ are given under the same field. Thus, the union compatibility of the relations $R$ and $S$ corresponds to the isomorphism of the subspaces $E_R$ and $E_S$. Under the geometric interpretation the union of two relations $R$ and $S$ corresponds to the union of point sets of subspaces $E_R$ and $E_S$.

**Intersection** ($R \cap S$).

The necessary condition for intersection of two relations $R$ and $S$ is their union compatibility.

Consequently, subspaces $E_R$ and $E_S$, that corresponds to the relations $R$ and $S$, should be isomorphic. The intersection of two relations $R$ and $S$ contains all the tuples of $R$ that also belong to $S$. Thus, the intersection of two relations $R$ and $S$ is corresponded to the intersection of subspaces $E_R$ and $E_S$.

**Difference** ($R - S$).

The difference of two relations $R$ and $S$ contains only those tuples of $R$ that do not belong to $S$.

Also, relations $R$ and $S$ should be a union compatible. As it was mentioned above, the union compatibility of the relations R and S corresponds to the isomorphism of the subspaces $E_R$ and $E_S$. Under the geometric interpretation the difference of two relations $R$ and $S$ corresponds to the set difference of point sets of subspaces $E_R$ and $E_S$.

**Cartesian product** ($R \times S$).

The Cartesian product of two relations $R$ and $S$ corresponds to the sum of the subspaces $E_R$ and $E_S$. It should be denoted that this sum is not a direct one, since if the attribute names of relations $R$ and $S$ coincide, the coordinate axis that correspond to these attributes are collinear.

**Division** ($R \div S$).

The division is a binary operation. The result consists of the restrictions of tuples in $R$ to the attribute names unique to $R$, i.e., in the header of $R$ but not in the header of $S$, for which it holds that all their combinations with tuples in $S$ are present in $R$. This operation may be expressed through the other ones:

$$T_1 = \Pi_c(R)$$
$$T_2 = \Pi_c((S \times T_1) - R)$$
$$T = T_1 - T_2$$

The projection, Cartesian product, and difference operations have been already interpreted in terms of geometry. Thus, the division operation could be geometrically interpreted.

**Joins**.

Join operation derivates from Cartesian product since it is equal to selection applied to Cartesian product of those tuples of two relations $R$ and $S$ that meet the condition specified in selection predicate. Thus, join operation of two relations $R$ and $S$ corresponds to the regional search in subspace resulting as sum of the subspaces $E_R$ and $E_S$. Natural join is a binary operator that is written as ($R$, $S$) where $R$ and $S$ are relations. The result of the natural join is the set of all combinations of tuples in $R$ and $S$ that are equal on their common attribute names. The right outer join of relations $R$ and $S$ is written as $R \quad X = S$. The result of the right outer join is the set of all combinations of tuples in $R$ and $S$ that are equal on their common attribute names, in addition to tuples in $S$ that have no matching tuples in $R$.

The ***outer join*** or ***full outer join*** in effect combines the results of the left and right outer joins.

The full outer join is written as $R = X = S$ where $R$ and $S$ are relations. The result of the full outer join is the set of all combinations of tuples in $R$ and $S$ that are equal on their common attribute names, in addition to tuples in $S$ that have no matching tuples in $R$ and tuples in $R$ that have no matching tuples in $S$ in their common attribute names.

## 3 Conclusion

In this paper we considered the problems of information retrieval from relational databases and proved their transformation to the geometric search problems in computational geometry, what allows us to use more efficient geometric algorithms for implementing search functions in hyper-large databases. To this end, we formalized input sets of the relational database search problem in terms and concepts of the geometric search problem, and per contra, results of geometric search problem solution interpreted in terms of databases. Also, by using examples of relational algebra search queries, we proved their geometrical realization, and hence, the transformation of two classes of problems, mentioned above.

The object of our future researches is to develop the general theory of transformation of database problems to computational geometry problems and to create new data structures for informational storing and searching using geometrical methods.

## References

[1] M. I. Shamos F. P. Preparata. Computational geometry. *SpringerVerlag*, 1985.

[2] T. Shinohara H. Arimura, H. Ishizaka. Learning unions of tree patterns using queries. *Theoretical Computer Science*, pages 47–62, 1997.

[3] Ryoichi Fujino Hiroki Arimura, Atsushi Wataki and Setsuo Arikawa. A fast algorithm for discovering optimal string patterns in large text databases. 2004.

[4] N. Cercone J. Han, Y. Cai. Knowledge discovery in databases: An attribute oriented approach. *In Proc. the 18th VLDB Conference*, pages 547–559, 1992.

[5] D. D. Lewis. Challenges in machine learning for text classification. *Proc. 9th Computational Learning Theory*, pages 1–8, 1996.

[6] A. Swami R. Agrawal, T. Imielinski. Mining association rules between sets of items in large databases. *Proc. the ACM SIGMOD Conference on Management of Data*, pages 207–216, 1993.

[7] S. Morishita T. Fukuda, Y. Morimoto and T. Tokuyama. Data mining using twodimensional optimized association rules. *Proc. the ACM SIGMOD Conference on Management of Data*, pages 13–23, 1996.