Chapter 19
An Experimental Performance Comparison for Indexing Mobile Objects on the Plane

S.Sioutas
Ionian University, Greece

G. Papaloukopoulos
University of Patras, Greece

K. Tsichlas
Aristotle University of Thessaloniki, Greece

Y. Manolopoulos
Aristotle University of Thessaloniki, Greece

ABSTRACT

In this paper, the authors present a time-efficient approach to index objects moving on the plane in order to answer range queries about their future positions. Each object is moving with non small velocity u, meaning that the velocity value distribution is skewed (Zipf) towards u_{min} *in some range* $[u_{min}, u_{max}]$*, where* u_{min} *is a positive lower threshold. This algorithm enhances a previously described solution (Sioutas, Tsakalidis, Tsichlas, Makris, & Manolopoulos, 2007) by accommodating the ISB-tree access method as presented in Kaporis et al. (2005). Experimental evaluation shows the improved performance, scalability, and efficiency of the new algorithm.*

INTRODUCTION

This paper focuses on the problem of indexing mobile objects in two dimensions and efficiently answering range queries over the objects' future locations. This problem is motivated by a set of real-life applications such as intelligent transportation systems, cellular communications, and meteorology monitoring. The basic approach uses discrete movements, where the problem of dealing with a set of moving objects can be considered as equivalent to a sequence of database snapshots of the object positions/extents taken at time instants $t_1 < t_2 < \ldots$, with each time instant denoting the

DOI: 10.4018/978-1-4666-1577-9.ch019

moment where a change took place. From this point of view, the indexing problems in such environments can be dealt with by suitably extending indexing techniques from the area of spatio-temporal databases (Gaede & Gunther, 1998; Salzberg & Tsotras, 1999). In (Manolopoulos, Theodoridis, & Tsotras, 2000) it is exposed how these indexing techniques can be generalized to handle efficiently queries in a discrete spatio-temporal environment.

The common thrust behind these indexing structures lies in the idea of abstracting each object's position as a continuous function of time, $f(t)$, and updating the database whenever the function parameters change. Accordingly an object is modelled as a pair consisting of its extent at a reference time (design parameter) and of its motion vector. One categorization of the aforementioned structures is according to the family of the underlying access method used. In particular, there are approaches based either on R-trees or on Quadtrees as explained in (Raptopoulou, Vassilakopoulos, & Manolopoulos, 2004, 2006). On the other hand, these structures can be also partitioned into those that: (a) are based on geometric duality and represent the stored objects in the dual space (Agarwal, Arge, & Erickson, 2000; Kollios, Gunopulos, & Tsotras, 1999; Patel, Chen, & Chakka, 2004), and (b) leave the original representation intact by indexing data in their native dimensional space (Beckmann, Begel, Schneider, & Seeger, 1990; Papadopoulos, Kollios, Gunopulos, & Tsotras, 2002; Saltenis, Jensen, Leutenegger, & Lopez, 2000; Saltenis et al., 2001; Tao, Papadias, & Sun, 2003). The *geometric duality transformation* is a tool extensively used in the Computational Geometry literature, which maps hyper-planes in R^d to points and vice-versa. In this paper we present and experimentally evaluate techniques using the duality transform as in (Kollios et al., 1999; Papadopoulos et al., 2002) to efficiently index future locations of moving points on the plane.

In the next section, we present a brief overview of the most basic practical methods. We then give a formal description of the problem. Next, we introduce the duality transform methods and then briefly present our main contribution, followed by the ISBs access method that compares favourably with the solutions of (Kollios et al., 1999; Papadopoulos et al., 2002), the TPR* index (Tao et al., 2003), the STRIPES index (Patel et al., 2004) and the LBTs index (Sioutas et al., 2007) as well. In simple words, the new solution is the most efficient in terms of update I/O performance. Moreover, with respect to the query I/O performance, solution of ISBs is 4 or 5 faster than LBTs method and outperforms STRIPES (state of the art as of now) in many settings. Finally, we present a thorough experimental evaluation, followed by a conclusion.

A BRIEF OVERVIEW OF THE RELEVANT METHODS

The TPR tree (Saltenis et al., 2000) in essence is an R*-tree generalization to store and access linearly moving objects. The leaves of the structure store pairs with the position of the moving point and the moving point id, whereas internal nodes store pointers to subtrees with associated rectangles that minimally bound all moving points or other rectangles in the subtree. The difference with respect to the classical R*-tree lies in the fact that the bounding rectangles are time parameterized (their coordinates are functions of time). It is considered that a time parameterized rectangle bounds all enclosed points or rectangles at all times not earlier than current time. Search and update algorithms in the TPR tree are straightforward generalizations of the respective algorithms in the R*-tree; moreover, the various kinds of spatiotemporal queries can be handled uniformly in 1-, 2-, and 3-dimensional spaces.

The TPR-tree served as the base structure for further developments in the area (Saltenis et al.,