Spatial Indexing in Flash Memories: Proposal of an Efficient and Robust Spatial Index with Durability

Anderson Chaves Carniel¹, Supervisor: Cristina Dutra de Aguiar Ciferri¹

¹Department of Computer Science – University of São Paulo (USP)
São Carlos, SP 15.560-970, Brazil
accarniel@gmail.com, cdac@icmc.usp.br

Mês e Ano de Ingresso no Doutorado: 11/2014
Mês e Ano Previsto para Defesa do Doutorado: 11/2018
Conclusão de Exame de Qualificação: 06/07/2016

Resumo. Para melhorar o tempo de processamento de consultas espaciais, sistemas de banco de dados espaciais utilizam índices espaciais, tal como o R-tree e suas variantes R+-tree, R*-tree e Hilbert R-tree. Esses índices são projetados para discos magnéticos e não sendo adequados para serem utilizados em memórias flash, que vem substituindo os discos magnéticos devido as suas vantagens. Porém, os índices espaciais devem ser revistos para garantir um bom desempenho nas memórias flash devido as suas características intrínsecas, tal como o custo assimétrico nas operações de escrita e leitura. Os poucos índices espaciais existentes para essas memórias possuem diversas limitações, tal como a consideração apenas das características intrínsecas das memórias flash em detrimento das características da indexação espacial, o que pode degradar significativamente o desempenho e a robustez do índice. Nesse sentido, esse projeto de doutorado objetiva desenvolver um índice robusto, com tempo de processamento eficiente de suas operações e com durabilidade dos dados.

Palavras-chave: banco de dados espaciais, indexação espacial, memórias flash, avaliação experimental, processamento de consultas espaciais

Abstract. In order to improve the spatial query processing, spatial database systems employ spatial indices, such as the R-tree and its variants the R+-tree, the R*-tree, and the Hilbert R-tree. These indices are designed for magnetic disks and are not adequate to be used in flash memories, which have replacing the magnetic disks due to its advantages. However, spatial indices need to be redesigned to guarantee a good performance in flash memories due to its unique characteristics, such as the asymmetry between the read and write costs. The few existing spatial indices for flash memories has several limitations, such as the only consideration of the unique characteristics of flash memories without consider the spatial indexing characteristics, and as consequence, it can significantly degrade the performance and robustness of the spatial index. In this sense, this PhD project aims to develop a robust spatial index, with an efficient time processing in its operations, and with durability of data.

Keywords: spatial databases, spatial indexing, flash memories, experimental evaluation, spatial query processing
1. Introduction

Spatial database systems and Geographical Information Systems (GIS) enable the storage and data management of spatial objects in many applications. For this purpose, spatial data types for points, lines, and regions have been modeled in order to represent real-world phenomena. Examples are points that represent buildings, lines that represent rivers, and regions that represent cities. Spatial queries commonly ask for spatial objects that satisfy some topological predicate (e.g., overlap, inside) according to a given object. For instance, a spatial selection that find all rivers intersecting a city. Another example is the spatial join that returns all pairs of spatial objects from two sets of spatial objects that satisfy a topological predicate, e.g., find all pairs of rivers and cities that intersect.

Since the spatial query processing with topological predicates tends to be a very expensive operation, spatial indices are employed to speed up this processing. The most commonly adopted spatial indices have hierarchical structures, such as the R-tree and its variants, R+-tree, R*-tree, and Hilbert R-tree. In general, these indices deal with spatial objects stored in magnetic disks, and thus consider the slow mechanical access as well as the cost of search and rotational delay of magnetic disks. On the other hand, flash memories have widely been used as the main storage device in mobile phones, laptops, and servers of data. The popularization of flash memories has been increased due to their better characteristics compared to magnetic disks, such as (i) smaller size, (ii) lighter weight, (iii) lower power consumption, (iv) better shock resistance, and (v) faster read and write operations.

Although the spatial indices are easily portable to be used in flash memories, this direct approach would not take advantage of all advantages of flash memories since it does not consider other characteristics of them and as a result, a sub-optimal approach. A main characteristic of flash memories is the asymmetry between read and write costs, where a write operation requires more time and power consumption than a read operation. Hence, existing spatial indices should be redesigned in order to take into account the unique characteristics of flash memories.

Spatial indices for flash memories have been proposed in the literature. In general, they consist in adaptations of the R-tree that consider the unique characteristics of flash memories. However, they face several problems. First, they focus only in the unique characteristics of flash memories and do not investigate the impact of these characteristics on the spatial indexing. This limitation impacts on the robustness of the index, i.e., the order and type of operations (e.g., insert, search, and update) performed on the index may negatively affect its performance. Second, a majority of these spatial indices establishes a trade-off between efficiency and durability. That is, these approaches sacrifices data consistency (i.e., durability) in favor to the efficiency by employing buffers in main memory containing the most recent modifications of the index to decrease the number of random writes in flash memories. Thus, after a system crash or a power loss, the modifications stored in these buffers are lost. Finally, these approaches do not conduct an extensive experimental evaluation that vary searching, inserting, deleting, and updating operations.

The goal of this PhD project is to propose a novel spatial index for flash memories that takes into account their unique characteristics together with key aspects of spatial indexing in order to guarantee efficiency, robustness, and durability. Hence, this PhD project has focus on the applicability of spatial indices in spatial databases and GIS man-
aged in flash memories. Our goal is also to conduct an extensive experimental evaluation to analyze the time processing of our index with related work and spatial indices designed for magnetic disks. As a result, we expect a higher popularization of flash memories in the spatial database context with the utilization of our spatial index. We also expect to advance in the state of the art in the indexing of spatial objects stored in flash memories. Finally, this index would be different from generic indices for flash memories since it will consider spatial features, such as the complexity of the spatial data types, and the spatial organization in order to improve the spatial query processing.

This paper is organized as follows. Section 2 summarizes underlying concepts from spatial indexing. Section 3 briefly describes the unique characteristics of flash memories. Section 4 surveys related work. Section 5 details this PhD project, while Section 6 summarizes ongoing activities. Section 7 concludes the paper and presents future work.

2. Spatial Databases and Spatial Indexing

Spatial queries [Gaede and Günther 1998] are an important processing of spatial databases. Topological predicates (e.g., overlap, disjoint, inside) [Schneider and Behr 2006, Clementini and Cohn 2014] frequently are employed in spatial queries to retrieve spatial objects that satisfy some predicate. The determination of a topological predicate between two spatial objects is performed by checking the existence of intersections among their interiors, exteriors, and boundaries, which leads to the 9-Intersection Model [Schneider and Behr 2006]. Examples of spatial queries are spatial selection and spatial join. Spatial selection (or spatial range query), asks for all objects from a set of spatial objects $D$ that satisfy a topological predicate $P$ for a given spatial object $s$, i.e., $\{o|o \in D \land P(o, s) = \text{true}\}$. If $s$ is a rectangular-shaped object, then $s$ is called window query. If $s$ is a point object and $R = \text{intersects}$, it leads to a point query. Given two sets of spatial objects $A$ and $B$, a spatial join query returns all pairs of objects $(o, o') \in A \times B$ that satisfy a topological predicate $P$, i.e., $\{(o, o')|o \in A \land o' \in B \land P(o, o') = \text{true}\}$. In general, due to the computation of topological predicates, the spatial query processing is a very expensive operation.

To improve the spatial query processing with topological predicates, spatial indices are employed, such as the R-tree [Guttman 1984]. The R-tree is a hierarchical index composed by internal and leaf nodes, where indexed spatial objects are maintained in leaf nodes. In order to index spatial objects of any type, improve the processing of topological predicates, and minimize the storage of the index, the Minimum Bounding Rectangles (MBR) of the objects are used. A leaf node of the R-tree comprises entries in the format $(mbr_o, p_o)$, where $mbr_o$ refers to the MBR of the spatial object $o$ and $p_o$ is a pointer that guarantees the direct access to $o$. An internal node of the R-tree comprises entries in the format $(mbr_c, p_c)$, where $mbr_c$ is the MBR that encompasses all the entries of the child node $c$ and $p_c$ is a pointer to the child node $c$. The minimum and maximum capacity of internal and leaf nodes are determined by $m$ and $M$ respectively such that $m \leq M/2$. If the maximum capacity of a node is achieved in an insertion operation, a split operation is performed. Variants of the R-tree are also proposed, such as the R*-tree [Beckmann et al. 1990]. These variants mainly modify the algorithm of the insertion of spatial objects and the split operation.
3. Flash Memories

Flash memories are present in several products, such as flash drives, memory cards, and Solid-State Drives (SSDs). These memories have asymmetry of read and write costs and other unique characteristics. The first characteristic is that the flash memory is block-oriented, that is, data is stored in flash blocks, where a block contains a fixed number of pages. Thus, the number of pages is the same for all blocks.

The second characteristic is that the flash memory provides the read, write, and erase operations with specific features [Brewer and Gill 2011]. The read operation is a page level operation and is the fastest flash memory operation (∼25us) since it can occur in anywhere of the memory without restrictions. The erase operation is a block level operation and it is the slowest flash memory operation (∼2ms) since it sets the bit 1 for all pages contained in the target block. The write operation is a page level operation that is able to set bits to 0 in erased blocks only. Hence, two situations are possible. The first situation is to set all the corresponding bits to 0 in a previously erased block. Such situation leads to a sequential write operation which is a low latency operation (∼200us). The second situation is to perform the write operation in pages located in an already written block. Such situation leads to the erase-before-update operation, which saves the pages located in the block, erases this block (i.e., an erase operation), and then applies the writing operation together with the saved pages. Hence, this situation is the most expensive operation.

The third characteristic is the employment of the Flash Translation Layer (FTL) [Chung et al. 2009]. The main goal of the FTL is to provide an interface to the operational system that allows to use a flash memory as an usual disk. It provides a mapping between logical page addresses, which are used by the operational system, and physical page addresses of the flash memory. The FTL marks each logical page as either free, valid, or invalid. A free logical page is an available page to store data, while a valid page is a page that already contains data. When a write operation is required to be performed on a valid page, this page is logically marked as invalid, and then its new content is stored in a free page. Hence, it avoids an erase-before-update operation. However, if there is a great number of invalid pages and space is required for write operations, a garbage collection is performed and leads to a very expensive operation since many erase-before-update operations can be executed. For more details of FTLs, please see a survey in [Chung et al. 2009].

Commonly, the FTL is included inside of commercial flash memories (e.g., SSDs) and thus, they are not open source. Due to this, the collection of some information like the number of allocated pages and the number of performed garbage collections are not possible. In order to carefully evaluate the performance of an application in a flash memory, several flash memory simulators have been proposed, such as FlashSim [Kim et al. 2009] and Flash-DBSim [Su et al. 2009].

4. Related Work

There are a few approaches that propose spatial indices for flash memories: the RFTL [Wu et al. 2003], the LCR-tree [Lv et al. 2011], the FAST [Sarwat et al. 2013], and the FOR-tree [Jin et al. 2015]. In general, these approaches attempt to adapt the R-tree index to be efficiently used in flash memories. We can distinguish the available
approaches according to four characterizations: (i) the employment of buffers in main memory, (ii) the data consistency (i.e., durability), (iii) the support of a flushing policy, (iv) the conduction of extensive experimental evaluations.

Regarding the first characterization, all the approaches provide a buffer in main memory that stores modifications of the spatial index before to be directly applied in the flash memory. Thus, the number of the random write operations is decreased. A majority of the approaches [Lv et al. 2011, Sarwat et al. 2013, Jin et al. 2015] provides an organization of the modifications in the buffer and thus, they improve the performance to retrieve a node. In addition, the FAST and the FOR-tree store applied modifications of a node in the buffer, which its recovery its facilitated since the node in buffer corresponds to its final version.

Regarding the second characterization, only the FAST provides the data consistency of the buffer by using sequential logs of the buffer data in the flash memory. It allows the data durability by recovering the buffer after a system crash.

Regarding the third characterization, the approaches have distinct ways to perform a flushing operation when the buffer in main memory reaches its maximum capacity. The goal of a flushing operation is to perform a sequential write operation instead of random writes. In the RFTL and the LCR-tree, the flushing operation applies all the modifications stored in the buffer, which implies in a high processing time and causes in a waiting time for the processing of other spatial index operations. On the other hand, the FAST and the FOR-tree employ a flushing policy that chooses a group of nodes (called flushing unit) to be written in batch by the flushing operation. In general, the flushing policy picks the nodes that have the lesser chances to be modify in the future, e.g., with the highest number of modifications and the oldest modifications.

Regarding the fourth characterization, all the approaches mainly conduct experimental evaluations considering the unique characteristics of flash memories and thus, consider the number of writes as main comparison metric. On the other hand, aspects of spatial indexing are also important, such as the spatial query processing. For instance, the FOR-tree eliminates the splitting operation of the R-tree since it requires more random writes to be done in the flash memory. Thus, it allows overflowed nodes. However, it directly impacts in spatial query processing because the modification of the spatial index structure also affects its performance. Experimental analyses also considering spatial indices designed for magnetic disks are also needed in order to identify the situations that these indices have bad performance in flash memories. An initial experiment considering the R*-tree was conducted in [Emrich et al. 2010]. However, extensive experimental evaluations that encompass different spatial indices (i.e., designed for magnetic disks and flash memories) as well as different rates of modification together with spatial query processing are needed.

5. The PhD Proposal

Spatial indexing is a core aspect in spatial databases and GIS. This importance together with the utilization and popularization of flash memories has raised the emergence of new approaches and spatial indices for flash memories in order to efficiently manage spatial objects in spatial databases and GIS. In this context, the goals of this PhD are detailed in Section 5.1 while the methodology of validation is described in Section 5.2.
5.1. Goals
This PhD project aims to design and implement a spatial index for flash memories that consider both the key features of spatial indexing and the unique characteristics of flash memories. This spatial index will provide: (i) efficiency to process the operation in an acceptable time performance, (ii) robustness, which will efficiently process operations independently of their orders and the volume of data, (iii) durability of data, which will guarantee the consistency of data. Regarding the key features of spatial indexing, we will take into account a spatial organization that contribute for fast searching, inserting, removing, and updating operations on the index. Regarding the unique characteristics of flash memories (Section 3), we will employ techniques that minimize the number of writes in the index to be proposed since it is the slowest flash memory operation. Further, this PhD project aims to conduct an extensive experimental evaluation to compare the performance of the proposed spatial index with the spatial indices designed for magnetic disks (Section 2) and flash memories (Section 4).

5.2. Validation
The methodology of validation of the results of this PhD will be realized by means of experimental evaluations aiming to compare the proposed spatial index with other spatial indices designed for magnetic disks and flash memories. The experimental evaluations will consider a number of factors. First, the origin of data, which will include real datasets extracted from OpenStreetMaps\(^1\) and synthetic datasets. Second, the volume of data, which will aid in the analysis of the scalability. Third, the type of performed operations, which will include different workloads composed by searching, inserting, removing, and updating (i.e., a remotion followed by an insertion) operations in order to analyze the robustness of a spatial index. More specifically, the searching operation will include spatial joins and spatial selections since these are the most common spatial queries. Other factors may be included in development of this PhD project. The main comparison metrics will be the time processing of operations and the number of writing operations. The hardware to be utilized is composed by virtual machines of the Microsoft Azure\(^2\) as well as a local server with two SSDs with different capacities and performance.

6. Ongoing Work
To date, we are studying and analyzing the unique characteristics of flash memories that can affect the spatial indexing. Thus, we expect to collect design criteria that the proposed indexed should reach. Some examples of design criteria are: (i) the usage of buffers in main memory in order to avoid random writes, (ii) the select of some information stored in the buffer to be sequentially flushed in batch in a flushing operation, and (iii) the creation of some strategy to guarantee the consistency of data (e.g., logs).

Another ongoing activity that will help to analyze design criteria for the proposal of a spatial index for flash memories is the development of a framework that includes the most popular spatial indices designed for magnetic disks (e.g., the R-tree and the R*-tree). We have a first version of this framework, called FESTIval\(^3\) (stands for Framework to

---

\(^1\)http://wiki.openstreetmap.org/wiki/Main_Page
\(^2\)https://azure.microsoft.com/en-us/documentation/
\(^3\)http://gbd.dc.ufscar.br/festival/
Evaluate Spatial Indices in non-volatile memories for PostgreSQL) [Carniel et al. 2016]. It is a free PostgreSQL extension and allows the experimental evaluation of different spatial indices (currently, the R-tree and the R*-tree) with different parameters. FESTIval also collects several statistical data related to a performed operation of a spatial index, such as the creation and the spatial query processing.

We are conducting experiments by using the FESTIval according to Section 5.2. The analysis of these experiments will contribute to the refinement of the proposed spatial index and to investigate the following aspects. First, the relation between the performance of spatial indices in magnetic disks and SSDs, which will allow to study the similarities of spatial indexing in both storage systems and help in the proposal of the spatial index for flash memories. Second, the parameterization of the spatial indices in the flash memories, e.g., the relation between the page size and the flushing unit that can directly impact in the performance of the flushing operation. Finally, the variety of requirements for the flushing policy used in the flushing operation that can unify concepts from the spatial indexing and the unique characteristics of flash memories. As a result, we expect to identify the disadvantages to be avoided and the advantages to be used in the spatial index to be proposed.

7. Conclusions and Future Work

Flash memories like SSDs have been very popular in the storage and management of data (e.g., spatial objects) since they have many positive characteristics, such as faster read and write operations than magnetic disks. On the other hand, flash memories have asymmetric read and write costs, where the writing operation required more time and power than the reading operation. With this, several algorithms have to be adapted or even redesigned in order to be efficiently used in flash memories. It includes the spatial indexing, which plays a major role in the spatial query processing. Thus, this PhD project has as main goal to propose a spatial index that considers both the key features of the spatial indexing as well as the unique characteristics of flash memories.

Future work will firstly include experimental evaluations of spatial indices designed for magnetic disks and flash memories (i.e., related work). Then, we aim to propose a new spatial index for flash memories and conduct its experimental analysis. For this, we intent to submit an internship project (BEPE) to FAPESP for the creation of an international research collaboration.

Acknowledgments. This work has been supported by the Brazilian federal research agencies CAPES and CNPq as well as by the São Paulo Research Foundation (FAPESP). A. C. Carniel has been supported by the grant #2015/26687-8, FAPESP. C. D. A. Ciferri has been supported by the grant #2016/04990-3, FAPESP.

References


