An Efficient R-Tree Implementation over Flash-Memory Storage Systems

Chin-Hsien Wu, Li-Pin Chang, Tei-Wei Kuo,
Proceedings of the 11th ACM international symposium on
Advances in geographic information systems,
November 07-08, 2003, New Orleans, Louisiana, USA

Introduction

• Geographic Information Systems (GIS) is one of the many popular applications over hand-held devices
  – How to efficiently access?

• R-tree is usually implemented as non-memory-resident index structures for the access of a large collection of spatial data

• Implementations over disks could not be applied directly over flash memory
  – Flash memory could not be over-written unless it is erased first
  – An erasable unit of a typical flash memory is relatively large
  – Garbage collection overheads
  – Limitation on the erase cycle count

Background

• R-tree
• Flash memory characteristics
  – A NAND flash is organized by blocks and pages
  – Write-once, do not overwrite data on update
    • “out-place update” strategy
    • Adapt RAM-resident translation table
    • Garbage collection: block-recycling policy
**Problem**

- Because of the characteristics of flash memory, updates of nodes must be done with out-place writings.
- The out-place updates for R-tree operations could result in the consumption of free pages:
  - Even though only a small portion of a node is modified.
  - Trigger garbage collection -> increase energy consumption.

**R-Tree Implementation**

- The reservation buffer
  - The write buffer residing in the main memory.
  - Any newly generated objects would be temporarily held.
  - Represent operations which have not yet been applied.
- Index Units
  - The physical representation of R-tree node.
  - Composed of pointers and flags.
- The Node Translation Table
  - Maintain the mapping of index units and the corresponding R-tree nodes.

**Table. Performance of a Typical NAND**

<table>
<thead>
<tr>
<th></th>
<th>Page Read</th>
<th>Page Write</th>
<th>Block Erase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>50</td>
<td>300</td>
<td>1881</td>
</tr>
<tr>
<td>Energy Consumption (uJ)</td>
<td>99</td>
<td>237.6</td>
<td>422.4</td>
</tr>
</tbody>
</table>

**Manipulations of Data Structure**

- Packing of index units
  - NP-Hard.
- Updating of the node translation table
  - Each R-tree node could be efficiently reconstructed.

**Node compaction**

- The node translation table is an array of list.
- A system parameter \( w \) is used the maximum number of items in lists.
- Once the number of items in a list grows over \( w \), the list must be compacted.
Experimental Results

- The system prototype was equipped with a 4MB NAND flash memory
  - Only index units created from the spatial objects were stored
- Two geographic files describing the roads and buildings of Taipei
  - Number of spatial objects: Buildings: 8590, roads: 7340

![The Taipei map](image)

Initiation Time for R-Tree Index Structures

- Measured the average response time of the insertions
- The average response time could also reflect the overheads

![Fig. avg. response time for geographic files](image)

![Fig. Number of pages being written for the geographic files](image)

![Fig. Number of pages being read for the geographic files](image)

Node Compaction Overheads

- To insert the spatial objects
  - Sequentially insert
  - Randomly pick one to insert

![Performance for Data Modifications](image)

- Spatial objects are randomly chosen
- PR do not bring any garbage collection due to a small number of pages written
**Conclusion**

- Propose an efficient R-tree implementation over flash-memory storage systems
- The implementation is over the flash translation layer
- Improve the performance & reduce the energy consumption

- Future work
  - Further exploit the energy consumption issue

**References**

- Chin-Hsien Wu, Li-Pin Chang, Tei-Wei Kuo, "An efficient R-tree implementation over flash-memory storage systems", Proceedings of the 11th ACM international symposium on Advances in geographic information systems, p.17-24, November 07-08, 2003, New Orleans, Louisiana, USA