DEX: a High-Performance Graph Database Management System

* dex

A High-Performance Graph Database Management System

Authors:
Norbert Martínez-Bazan
Sergio Gómez-Villamor
Francesc Escalé-Claveras

* sparsity technologies
performance in action

DAMA-UPC. DATA MANAGEMENT
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Outline

- Introduction
- DEX
  - Logical graph model
  - Internal representation
- Software architecture
- Experimental results
- Conclusions
- Future work
Introduction

- [2006] DEX started by DAMA-UPC
- [2010] Sparsity Technologies is a spin-out from DAMA-UPC
  - Sparsity commercializes and provides services
  - DAMA-UPC does development and research

- DEX Versions
  - V2.0  March/2009
  - V3.0  October/2009
  - V4.0  November/2010
DEX

- DEX is a graph database:
  - Data and schema both are represented as a graph
  - Data operations are based on graph operations
  - Graph-based integrity restrictions


- Focus:
  - Management of very large graphs
  - High-performance on query operations
Logical graph model

- **Labeled**: nodes and edges are “typed”
- **Directed**: edges can have a fixed direction
- **Attributed**: nodes and edges can have multiple single-valued attributes
- **Multigraph**: two nodes can be connected by multiple edges
Internal representation

- Requirements
  - Split the graph into smaller structures
    - Favour the caching
    - Move to main memory just significant parts
  - OIDs instead of objects
    - Reduce memory requirements
  - Specific structures to improve traversals
    - Index edges of a node
  - Attributes fully indexed
    - Improve queries based on value filters
Internal representation

- Our approach:
  - Map + Bitmaps → Link

- Link: bidirectional association between values and OIDs
  - Two functionalities:
    - Given a value → a set of OIDs (a bitmap)
    - Given an OID → the value
Internal representation

A Graph as a combination of Bitmaps:
- 1 Bitmap for each node or edge type
- 1 Link for each attribute
- 2 Links for each edge type:
  - Out-going and in-going edges

Software architecture

- **DEXCORE**
  - Complete C++ library
    - Storage and query
  - Linux / Windows / MacOSX
  - 32 / 64 bits

- **JDEX**
  - DEXCORE functionality provided as a Java library
Software architecture

DEXCORE:

- **IO**
  - Segment: Logical space of pages
  - Pool: Groups of segments
  - Storage: I/O device
  - Cache: I/O management
    - Replacement policy

- **Data:**
  - Paged out-of-core structures
  - Bitmaps, Maps, Links, …

- **Graph:**
  - A combination of structures
  - DbGraph and RGraphs

- **DEX:**
  - Database and Session management
Software architecture

Implementation details:

- 37-bit unsigned integer OIDs
  - + 137 billion objects per graph
- Bitmaps are compressed
  - Clusters of 32 consecutive bits
  - Just existing clusters are stored
- Groups of OIDs for each type
  - Higher density of consecutive bits into bitmaps
- Maps are B+ trees
  - A compressed UTF-8 storage for UNICODE strings
## Experimental results

- **Load tests**

<table>
<thead>
<tr>
<th></th>
<th>IMDB</th>
<th>Wikipedia</th>
<th>RMAT (sf=28)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB</td>
<td>2.4 GB</td>
<td>7.6 GB</td>
<td>83 GB</td>
</tr>
<tr>
<td>Physical Mem</td>
<td>9 GB</td>
<td>9 GB</td>
<td>60 GB</td>
</tr>
<tr>
<td>Load time</td>
<td>21 min</td>
<td>2h 6min</td>
<td>15h</td>
</tr>
<tr>
<td>Nodes</td>
<td>13 M</td>
<td>19 M</td>
<td>230 M</td>
</tr>
<tr>
<td>Edges</td>
<td>22 M</td>
<td>180 M</td>
<td>2147 M</td>
</tr>
<tr>
<td>Values</td>
<td>48 M</td>
<td>283 M</td>
<td>230 M</td>
</tr>
<tr>
<td>Insertions per sec</td>
<td>65 K</td>
<td>62 K</td>
<td>48 K</td>
</tr>
</tbody>
</table>
Experimental results

- **Query tests**
  - IMDB database (2.4 GB)
  - Queries:
    - A: full extraction of a movie, multiple 1-hop traversal [4K edges]
    - B: distance between two actors [8K edges]
    - C: extract all movies that match a given pattern [315K edges]

<table>
<thead>
<tr>
<th></th>
<th>In-memory</th>
<th>128 MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.13 sec</td>
<td>0.13 sec</td>
</tr>
<tr>
<td>B</td>
<td>1.52 sec</td>
<td>1.79 sec</td>
</tr>
<tr>
<td>C</td>
<td>384 sec</td>
<td>385 sec</td>
</tr>
</tbody>
</table>
Conclusions

- We propose **DEX**, a high performance graph database querying system for **labeled and directed attributed multigraphs**

- We propose a graph representation based on the intensive use of bitmaps

- We perform an experimental performance analysis to show the ability of DEX to store and query **very large graphs**
Future work

- Trillions of objects
- Transactional system
- Distributed system
- Query language
  - Query optimization
- High-level graph operations
  - Pattern matching
Questions?

sgomez@sparsity-technologies.com

Sparsity Technologies  http://www.sparsity-technologies.com

DAMA-UPC  http://www.dama.upc.edu