Meet the Walkers

Accelerating Index Traversals for In-Memory Databases

Onur Kocberber¹, Boris Grot², Javier Picorel³, Babak Falsafi¹, Kevin Lim³, Parthasarathy Ranganathan⁴

¹EcoCloud, EPFL  ²University of Edinburgh  ³HP Labs  ⁴Google, Inc.

Our World is Data-Driven!

- Data resides in huge databases
- Indexes are essential for all database operators
- Hash index: a fundamental index structure
- Frequent use case: join via hash index

Dissecting Index Lookups

Hash: Avg. 30% time of each lookup
- Computationally intensive, high cache locality

Walk: Avg. 70% time of each lookup
- Trivial computation, low cache locality

Next lookup: Inherently parallel
- But, beyond the inst. window capacity

Roadmap

1. Specialize
- Design a dedicated unit for hash and walk
  - Hash: Customized hardware for hash and walk
  - Walk: Perform multiple index lookups at a time
  - Generalize: Use a programmable building block

2. Parallelize
- Widx unit: common building block for hash and walk
  - Two-stage RISC core
  - Custom ISA

3. Generalize
- Widx units are programmable
  - Execute functions written in Widx ISA
  - Support limitless number of data structure layouts

Putting it all together: Widx

- Widx runs, core goes idle
- Widx is a two-stage RISC core
- Widx supports limitless number of data structure layouts

Result Highlights

- More details in the paper!
  - Per query results
  - Join kernel evaluation
  - Programming model
  - Bottleneck analysis
  - Widx cycle breakdowns

How Much Time is Spent Indexing?

Measurement on Xeon 5670 CPU with 100GB Dataset on MonetDB

<table>
<thead>
<tr>
<th>Indexing Runtime</th>
<th>Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
</tr>
<tr>
<td>2</td>
<td>1.5</td>
</tr>
<tr>
<td>3</td>
<td>2.0</td>
</tr>
<tr>
<td>4</td>
<td>2.5</td>
</tr>
</tbody>
</table>

Indexing is the biggest contributor to execution time.