Linearly Compressed Pages: A Main Memory Compression Framework with Low Complexity and Low Latency

Gennady Pekhimenko
Advisers: Todd C. Mowry & Onur Mutlu
Executive Summary

- Main memory is a limited shared resource
- **Observation**: Significant data redundancy
- **Idea**: Compress data in main memory
- **Problem**: How to avoid latency increase?
- **Solution**: Linearly Compressed Pages (LCP):
  - fixed-size cache line granularity compression
  1. Increases capacity (**69%** on average)
  2. Decreases bandwidth consumption (**46%**)
  3. Improves overall performance (**9.5%**)
Challenges in Main Memory Compression

1. Address Computation

2. Mapping and Fragmentation

3. Physically Tagged Caches
Address Computation

Uncompressed Page

Address Offset

0
64
128
(N-1)*64

Compressed Page

Address Offset

0
?
?
?
Mapping and Fragmentation

Virtual Page
(4kB)

Physical Page
(?kB)

Virtual Address

Physical Address

Fragmentation
Physically Tagged Caches

Critical Path

Address Translation

Core

TLB

L2 Cache Lines

tag  data
ntag  data
tag  data

Virtual Address

Physical Address
# Shortcomings of Prior Work

<table>
<thead>
<tr>
<th>Compression Mechanisms</th>
<th>Access Latency</th>
<th>Decompression Latency</th>
<th>Complexity</th>
<th>Compression Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM MXT</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td><em>[IBM J.R.D. ’01]</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Shortcomings of Prior Work

<table>
<thead>
<tr>
<th>Compression Mechanisms</th>
<th>Access Latency</th>
<th>Decompression Latency</th>
<th>Complexity</th>
<th>Compression Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM MXT [IBM J.R.D. ’01]</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Robust Main Memory Compression [ISCA’05]</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
</tbody>
</table>
# Shortcomings of Prior Work

<table>
<thead>
<tr>
<th>Compression Mechanisms</th>
<th>Access Latency</th>
<th>Decompression Latency</th>
<th>Complexity</th>
<th>Compression Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM MXT [IBM J.R.D. ’01]</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Robust Main Memory Compression [ISCA’05]</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td><strong>LCP: Our Proposal</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

IBM MXT [IBM J.R.D. ’01]:
- Access Latency: ✗
- Decompression Latency: ✗
- Complexity: ✗
- Compression Ratio: ✓

Robust Main Memory Compression [ISCA’05]:
- Access Latency: ✗
- Decompression Latency: ✓
- Complexity: ✗
- Compression Ratio: ✓

LCP: Our Proposal:
- Access Latency: ✓
- Decompression Latency: ✓
- Complexity: ✓
- Compression Ratio: ✓
# Linearly Compressed Pages (LCP): Key Idea

**Uncompressed Page (4kB: 64*64B)**

| 64B | 64B | 64B | 64B | ... | 64B |

**4:1 Compression**

**Compressed Data** (1kB)

**Metadata** (64B):

? (compressible)

**Exception Storage**
LCP Overview

• Page Table entry extension
  – compression type and size
  – extended physical base address

• Operating System management support
  – 4 memory pools (512B, 1kB, 2kB, 4kB)

• Changes to cache tagging logic
  – physical page base address + cache line index
    (within a page)

• Handling page overflows

• Compression algorithms: BDI [PACT’12], FPC [ISCA’04]
LCP Optimizations

• **Metadata** cache
  – Avoids additional requests to metadata

• Memory bandwidth reduction:
  
  [Image showing 4 64B regions being transferred to a single region]
  
  1 transfer instead of 4

• Zero pages and zero cache lines
  – Handled separately in TLB (1-bit) and in metadata
    (1-bit per cache line)

• Integration with cache compression
  – BDI and FPC
Methodology

• Simulator
  – x86 event-driven simulators
    • Simics-based [Magnusson+, Computer’02] for CPU
    • Multi2Sim [Ubal+, PACT’12] for GPU

• Workloads
  – SPEC2006 benchmarks, TPC, Apache web server, GPGPU applications

• System Parameters
  – L1/L2/L3 cache latencies from CACTI [Thoziyoor+, ISCA’08]
  – 512kB - 16MB L2, simple memory model
**Compression Ratio Comparison**

SPEC2006, databases, web workloads, 2MB L2 cache

LCP-based frameworks achieve competitive average compression ratios with prior work
Bandwidth Consumption Decrease

SPEC2006, databases, web workloads, 2MB L2 cache

LCP frameworks significantly reduce bandwidth (46%)

![Bar chart showing bandwidth consumption decrease for different configurations.](chart.png)
## Performance Improvement

<table>
<thead>
<tr>
<th>Cores</th>
<th>LCP-BDI</th>
<th>(BDI, LCP-BDI)</th>
<th>(BDI, LCP-BDI+FPC-fixed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6.1%</td>
<td>9.5%</td>
<td>9.3%</td>
</tr>
<tr>
<td>2</td>
<td>13.9%</td>
<td>23.7%</td>
<td>23.6%</td>
</tr>
<tr>
<td>4</td>
<td>10.7%</td>
<td>22.6%</td>
<td>22.5%</td>
</tr>
</tbody>
</table>

LCP frameworks significantly improve performance
Conclusion

• A new main memory compression framework called LCP (Linearly Compressed Pages)
  – Key idea: fixed size for compressed cache lines within a page and fixed compression algorithm per page

• LCP evaluation:
  – Increases capacity (69% on average)
  – Decreases bandwidth consumption (46%)
  – Improves overall performance (9.5%)
  – Decreases energy of the off-chip bus (37%)
Linearly Compressed Pages: A Main Memory Compression Framework with Low Complexity and Low Latency

Gennady Pekhimenko

Advisers: Todd C. Mowry & Onur Mutlu