LARGE PAGES AND LIGHTWEIGHT MEMORY MANAGEMENT IN VIRTUALIZED ENVIRONMENTS: CAN YOU HAVE IT BOTH WAYS?

Binh Pham§, Jan Vesely§, Gabriel H. Loh‡, Abhishek Bhattacharjee§

§Rutgers University  ‡AMD Research
Large Pages Advantages

• Large pages are often used to mitigate address translation overhead
  - Increase TLB reach
  - Reduce page walk latency
Large Pages Disadvantages

- Larger working sets [1]
- Coarse protection granularity [1]
- Hurts performance on NUMA systems due to node imbalance and poor locality [2]


Our work: Large Page Benefits vs Light Weight Memory Management

• Memory management techniques:
  - Page sharing
  - Memory sampling
  - Memory compression
  - Virtual machine migration

• Hypervisor splinters large pages into small pages
Prevalence of Page Splintering and Performance Impact

Splintering Distribution

Address Translation Overhead

Fraction of TLB Misses

Fractions of Runtime

0% 20% 40% 60% 80% 100%
fraction of TLB misses

g-analytics graph500 tigr cactusADM mcf

0% 10% 20% 30% 40%
fraction of runtime

g-analytics graph500 tigr cactusADM mcf

MICRO 2015 Binh Pham – Rutgers University
Patterns in Splintered Pages

Page Tables

<table>
<thead>
<tr>
<th>GVP</th>
<th>GPP</th>
<th>HPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xA00</td>
<td>0xFE00</td>
<td>0x6400</td>
</tr>
<tr>
<td>0xA01</td>
<td>0xFE01</td>
<td>0x6401</td>
</tr>
<tr>
<td>0xA02</td>
<td>0xFE02</td>
<td>0x6402</td>
</tr>
<tr>
<td>0xA03</td>
<td>0xFE03</td>
<td>0x6455</td>
</tr>
</tbody>
</table>

Aligned Page Table Entries

- g analytics
- graph500
- tigr
- cactusADM
- mcf

Fraction of PTEs

0.0 0.2 0.4 0.6 0.8 1

MICRO 2015
Binh Pham – Rutgers University
GLUE Design: Address Interpolation

Page Tables

<table>
<thead>
<tr>
<th>GVP</th>
<th>GPP</th>
<th>HPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x1A00</td>
<td>0xFE00</td>
<td>0x6400</td>
</tr>
<tr>
<td>0x1A01</td>
<td>0xFE01</td>
<td>0x6401</td>
</tr>
<tr>
<td>0x1A02</td>
<td>0xFE02</td>
<td>0x6402</td>
</tr>
<tr>
<td>0x1A03</td>
<td>0xFE03</td>
<td>0x6455</td>
</tr>
</tbody>
</table>

Large Page TLB

<table>
<thead>
<tr>
<th>Spec</th>
<th>Tag</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0x1A</td>
<td>0x64</td>
</tr>
</tbody>
</table>

0x1A00 = 0001 1010 0000 0000

concat

0x6400
GLUE Speculative Entry Insertion

```
<table>
<thead>
<tr>
<th>Spec</th>
<th>Tag</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GVP[35:9]</td>
<td>HPP[35:9]</td>
</tr>
</tbody>
</table>
```

Diagram:
- 4KB L1 TLB
- 2MB L1 TLB
- L2 TLB
- Page Table Walker
- Speculative Entry Insertion
- GVP
- GLarge-HSmall <GVP, HPP>
GLUE Speculation Timeline

No speculation

CPU
TLB
L2 lookup Page table walk TLB insert

Normal execution

L1 correct speculation, verified from PTW

CPU
TLB
L1 spec Spec exec Spec exec 4KB L1 insert
L2 lookup Page table walk

Normal execution

Verify: Spec Correct

Time Saving
Methodology (See paper for details)

- **Workloads:**
  - Spec CPU2006
  - Biobench
  - Cloudsuite

- **System:**
  - 8 3.4 GHz cores, 24 GB RAM
  - Hypervisors: ESX, KVM
  - 8 VMs, 4 GB RAM each

- **Trace-driven simulator:**
  - Collect guest memory traces using Pin
  - Collect hypervisor memory traces using VMware scripts/customized KVM
  - TLB+cache simulator
How often/accurately do we speculate?

Fraction of L1 Misses

- no-spec
- correct-spec
- wrong-spec

g analytics, graph500, tigr, cactusADM, mcf
How effectively do we speculate?

![Bar chart showing the fraction of baseline PTWs off critical path for different benchmarks: g analytics, graph500, tigr, cactusADM, and mcf. The y-axis represents the fraction ranging from 0% to 100%, and the x-axis lists the benchmarks.]
How much performance do we get?

The chart illustrates the speedup for different applications: g analytics, graph500, tigr, cactusADM, and mcf. The y-axis represents speedup, ranging from 0% to 40%. The x-axis lists the applications. The chart shows the performance gain using speculation compared to the ideal scenario.
Conclusion

• Conflicting decision can be made in complex system with many layers.
• There likely still exists patterns that we can exploit.
• Our work demonstrates one example where this process can be handled effectively.
LARGE PAGES AND LIGHTWEIGHT MEMORY MANAGEMENT IN VIRTUALIZED ENVIRONMENTS: CAN YOU HAVE IT BOTH WAYS?

Binh Pham§, Jan Vesely§, Gabriel H. Loh‡, Abhishek Bhattacharjee§

§Rutgers University  ‡AMD Research