Bouquet of Instruction Pointers: Instruction Pointer Classifier-based Hardware Prefetching

DPC3@ISCA ‘19

Samuel Pakalapati (Intel Technology Pvt. Ltd. and BITS Pilani) and Biswabordan Panda (Indian Institute of Technology Kanpur)
Why a Bouquet?

No single IP based prefetcher performs well across all applications 😞
Our Goal: Idealistic Though 😊

Core

L1
L1 Prefetcher

L1 hit rate of 100% (a dream 😊)
RIP Memory wall 😊

Reality with SPEC CPU 2017 benchmarks provided by DPC3:
L1 hit rate of 88.12% 😞
What about L2? 23.55% 😞 😞
Zooming into the Prefetcher

Instruction Pointer (a.k.a. PC)

Prefetcher

Future Memory Accesses

Demand Memory Accesses (cache-line aligned addresses)

We use the IP information: can eliminate compulsory misses 😊

Started with the simplest IP prefetcher: IP-Stride
IP-Stride Prefetcher [Fu et al. MICRO ‘92]

Prefetch Address = Current Address + Stride

Good for constant strides
Our Bouquet

First IP prefetcher: Constant stride
Constant-stride prefetcher (CS class)

If \((\text{current\_page} = \text{last\_page})\) then stride within a page

Page boundary learning:
If \((\text{current\_page} = \text{last\_page} \pm 1)\)
Stride = \(64 \pm (\text{page\_offset\_new} - \text{page\_offset\_old})\)

<table>
<thead>
<tr>
<th>IP_index</th>
<th>IP_tag</th>
<th>Valid?</th>
<th>Last_page</th>
<th>Page_offset</th>
<th>Stride</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
[0,63], Cache line offset within a 4KB OS page
Valid Bit?

<table>
<thead>
<tr>
<th>IP_index</th>
<th>IP_tag</th>
<th>Valid?</th>
<th>Last_page</th>
<th>Page_offset</th>
<th>Stride</th>
<th>Confidence</th>
</tr>
</thead>
</table>

Two different IP_tags can map to same IP_index

IPA: V=1, IPB mapped to same entry: V=0,
IPA: V=0: IPA mapped to same entry: V=1
If V=0 but IP_tag is different then clear the entry and make confidence zero

~ 2-way associative cache, minimize collisions
Constant Stride Class

- **IP**: $X, X+2, X+4, \ldots \ldots\ldots \quad \text{Constant stride of 2}
- **IP**: $X, X+3, X+4, X+2 \ldots \quad \text{Variable stride of ?}

Signature Path Prefetching, DPC-2, MICRO '16
Our Bouquet

First IP prefetcher: Constant stride
Second IP prefetcher: Complex stride
Complex Stride (CPLX Class)  
[Kim et al., DPC-2/MICRO ‘16]  

<table>
<thead>
<tr>
<th>IP</th>
<th>Signature</th>
<th>Stride</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP_A</td>
<td>Sig_A (+1, +2, +3)</td>
<td>-3</td>
<td>2/3</td>
</tr>
</tbody>
</table>

We call it Delta Prediction Table (DPT)
From Stride to Stream: Global Stream

$X, X+1, Y, Y+4, Z, \ldots$  

$IP_X$ drives the global stream: $Y=X+2$ and $Z=X+7$

$IP$ independence can provide better coverage and timeliness
Our Bouquet

First IP prefetcher: Constant stride
Second IP prefetcher: Complex stride
Third IP prefetcher: Global stream
Global Stream (GS Class)

<table>
<thead>
<tr>
<th>IP</th>
<th>Stream Valid?</th>
<th>Stream Direction</th>
<th>Stream Strength?</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP&lt;sub&gt;X&lt;/sub&gt;</td>
<td>Yes (0/1)</td>
<td>+/-</td>
<td>Strong</td>
</tr>
</tbody>
</table>

X, X+1, Y, Y+1, Z, ..................

‑ If n/2 GHB hits, valid
‑ If 3n/4 hits, strong

GHB
(Global History Buffer)
n entries

X+1, X+2, ....... X+PrefetchDegree
Our Bouquet

- First IP prefetcher: Constant stride
- Second IP prefetcher: Complex stride
- Third IP prefetcher: Global stream
- Fourth prefetcher: Next-line
No-IP: Next-line (NL Class)

Prefetch Address = Current Address + 1

Detrimental to performance in case of irregular accesses

**SPECULATIVE NL:**
- NL is ON
- *L1 Misses Per Kilo Cycles (MPKC) is low (< 15 for single-core)*
- NL is OFF
- Otherwise
The Bouquet

Constant Stride (CS class)
Complex Stride (CPLX class)
Global Stream (GS class)
Next Line (NL class)

Design Choice: A hardware table for each class?
Our Proposal: IPCP, a single hardware table for all the classes
Our Proposal (IPCP at L1)

**L1 access [IP, Access address]**

**Priority of classes:**
MS > CS > CPLX > NL

**Prefetch Degree:**
GS: 6, CS and CPLX: 3
Our Proposal (IPCP at L2)

- **L1 Prefetcher**: GS, CS, CPLX, NL, NO
- **Trained Stride**, **Stream Direction**

<table>
<thead>
<tr>
<th>IP</th>
<th>Valid?</th>
<th>Class_type</th>
<th>Stride</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- No IP classification at the L2, table construction based on *metadata*
- No prefetching for CPLX class
- Prefetch Degree: 4 for GS and 4 for CS if MSHR is less than half full else 3
Metadata

L1 Prefetch Packet

Metadata

Stride (7 bits)  Class-type (3 bits)  SPEC_NL (1 bit)

Stream direction in case of GS class type
## Hardware Overhead

<table>
<thead>
<tr>
<th>Table</th>
<th>Entry size * #Entries</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Table</td>
<td>77 * 1024 (L1) + 17 * 1024 (L2) bits</td>
<td>12.03 KB</td>
</tr>
<tr>
<td>DPT Table</td>
<td>9 * 4096 bits</td>
<td>4.6 KB</td>
</tr>
<tr>
<td>GHB Table</td>
<td>16 * 58 bits</td>
<td>928 bits</td>
</tr>
<tr>
<td>Others</td>
<td>100 bits</td>
<td>86 bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>16.7 KB</strong></td>
</tr>
</tbody>
</table>
Single-core Performance [SPEC CPU 2017]

On average: 43.75% improvement
Multi-core: 25 mixes, 22% improvement
Distribution of IP Classes

On average, all classes trigger equally
Comparison with the State-of-the-art: Performance [Higher the better]

<table>
<thead>
<tr>
<th>Method</th>
<th>Average Improvement in %</th>
</tr>
</thead>
<tbody>
<tr>
<td>BO [HPCA '16, DPC-2 Winner]</td>
<td>34.53</td>
</tr>
<tr>
<td>SPP+ Perceptron Filter [ISCA '19]</td>
<td>40.40</td>
</tr>
<tr>
<td>IPCP</td>
<td>43.75</td>
</tr>
</tbody>
</table>
### Key Takeaways

| Access patterns can be **classified** based on IPs (IPCP) |
| Classification at the L1, **reuse** at the L2 through metadata |
| **Simple** and **modular** collection of prefetchers |
| Prefetchers like ISB [MICRO ‘13] and IMP [MICRO ‘15] can be added to the bouquet seamlessly |
| **High** performance and **low** hardware overhead |
Dream 😊❓

With IPCP, L1 hit rate jumps from 88.11% to 92.43% 😊

With IPCP, L2 hit rate jumps from 23.55% to 51.82% 😊
“Great things are done by a series of small things brought together”

Vincent Van Gogh, Dutch painter

Thank You