

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



DPC3@ISCA '19



*Samuel Pakalapati (Intel Technology Pvt. Ltd. and BITS Pilani) and
Biswabandan Panda (Indian Institute of Technology Kanpur)*



Why a Bouquet?

No single IP based prefetcher performs well across all applications ☹️



Our Goal: Idealistic Though 😊



L1 hit rate of 100% (a dream 😊)

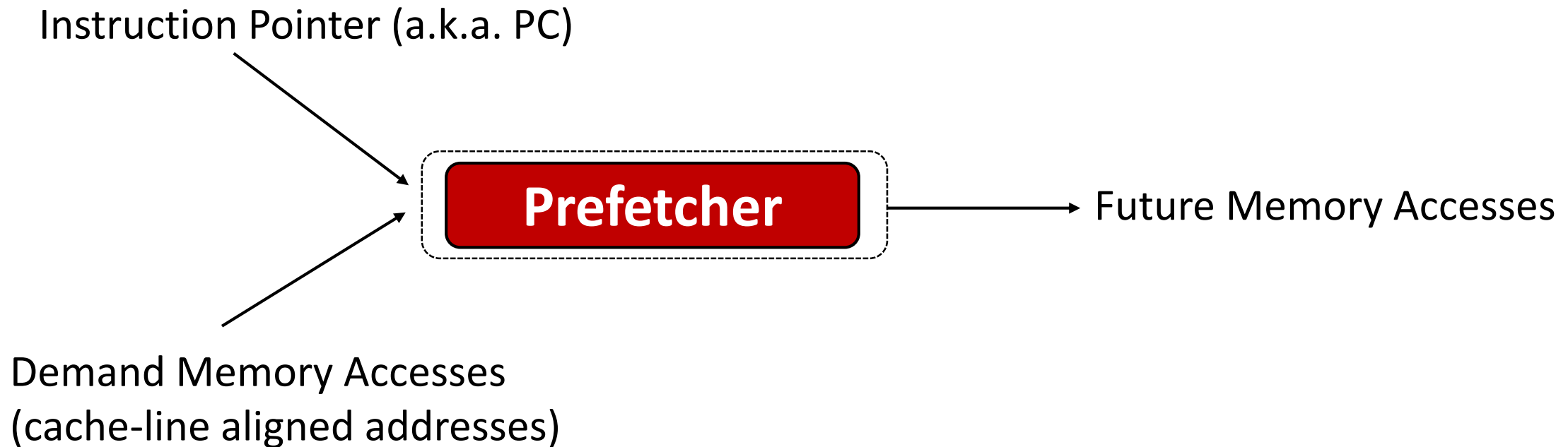
RIP Memory wall 😊

Reality with SPEC CPU 2017 benchmarks provided by DPC3:

L1 hit rate of **88.12%** 😞

What about L2? **23.55%** 😞 😞

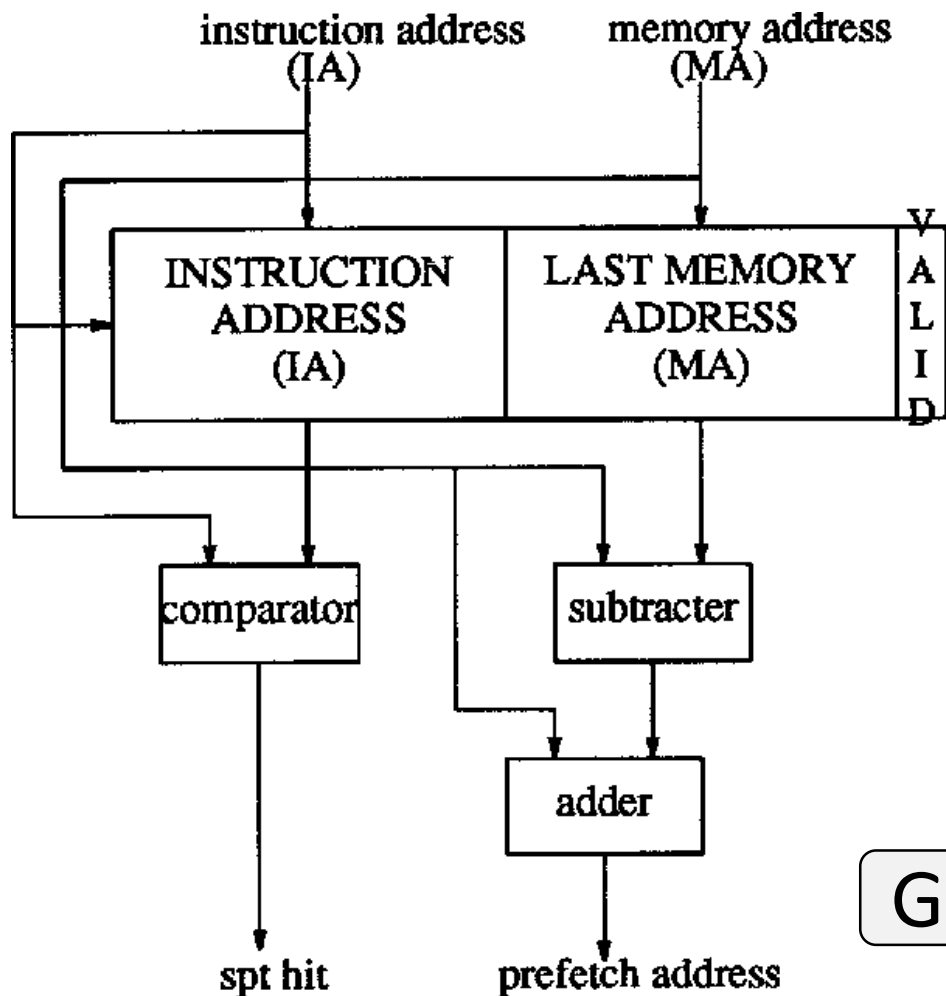
Zooming into the Prefetcher



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]



IP	Last-address	Stride

Prefetch Address = Current Address + Stride

Good for constant strides

Our Bouquet



First IP prefetcher: Constant stride

Constant-stride prefetcher (CS class)

IP_index	IP_tag	Valid?	Last_page	Page_offset	Stride	Confidence



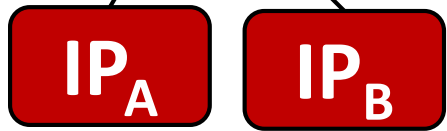
[0,63], Cache line offset within a 4KB OS page

If (current_page=last_page) then stride within a page

Page boundary learning:
If (current_page=last_page±1)
Stride = $64 \pm (\text{page_offset_new} - \text{page_offset_old})$

Valid Bit?

IP_index	IP_tag	Valid?	Last_page	Page_offset	Stride	Confidence



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,

Constant stride of 2

IP

X, X+3, X+4, X+2

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)

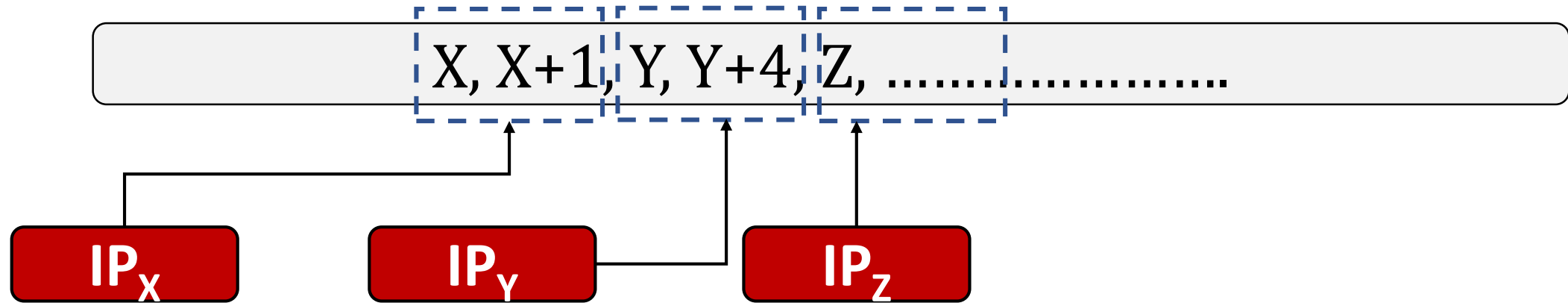
[Kim et al., DPC-2/MICRO '16]

IP	Signature	Stride	Confidence
IP_A	$Sig_A (+1, +2, +3)$	-3	2/3

+1 +2 +3 -3 +1 +2 + 3 -4 +1 +2 +3 -3

We call it Delta Prediction Table (DPT)

From Stride to Stream: Global Stream



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

IP independence can provide better coverage and timeliness

Our Bouquet

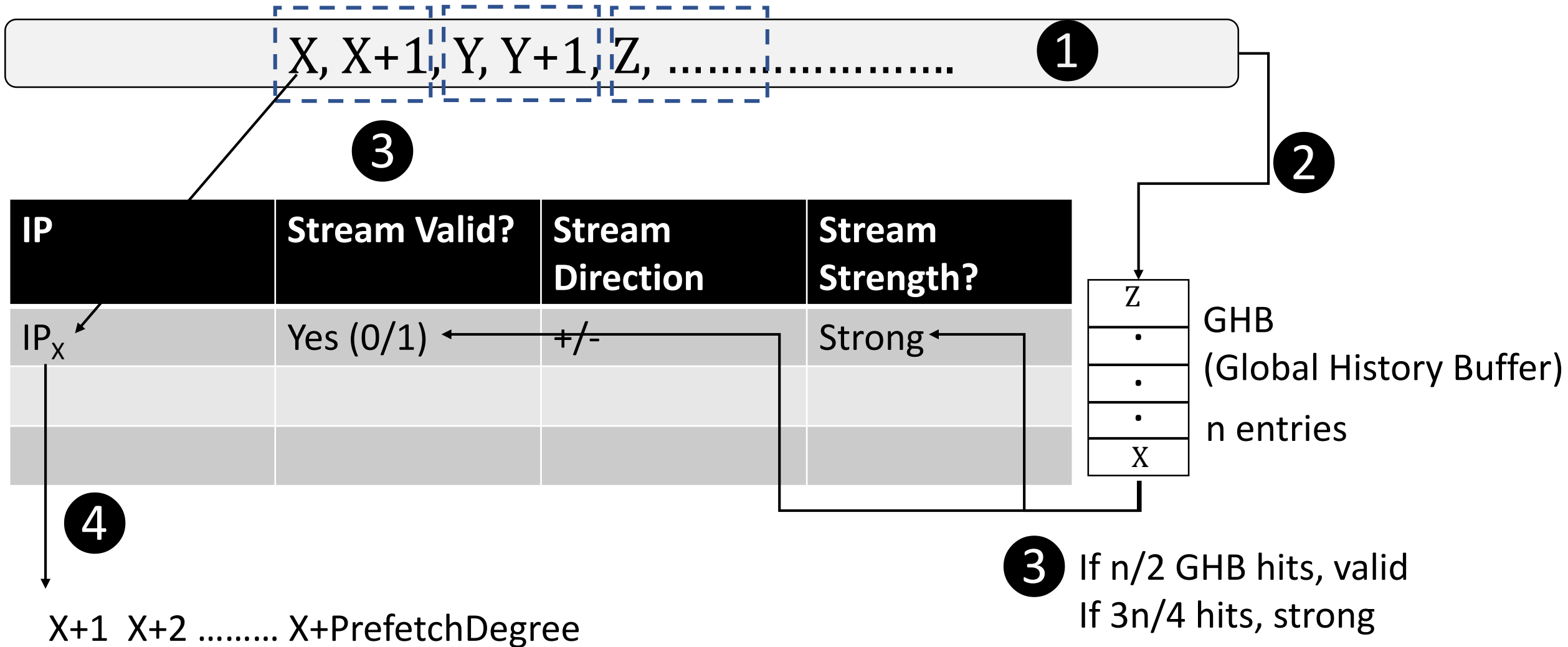


First IP prefetcher: Constant stride

Second IP prefetcher: Complex stride

Third IP prefetcher: Global stream

Global Stream (GS Class)



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]

CS

CPLX

GS

IP	Valid?	Page no.	Page offset	Stride	Confidence	Signature	Stream valid?	Direction	Strength

Priority of classes:

GS > CS > CPLX > NL

Prefetch Degree:

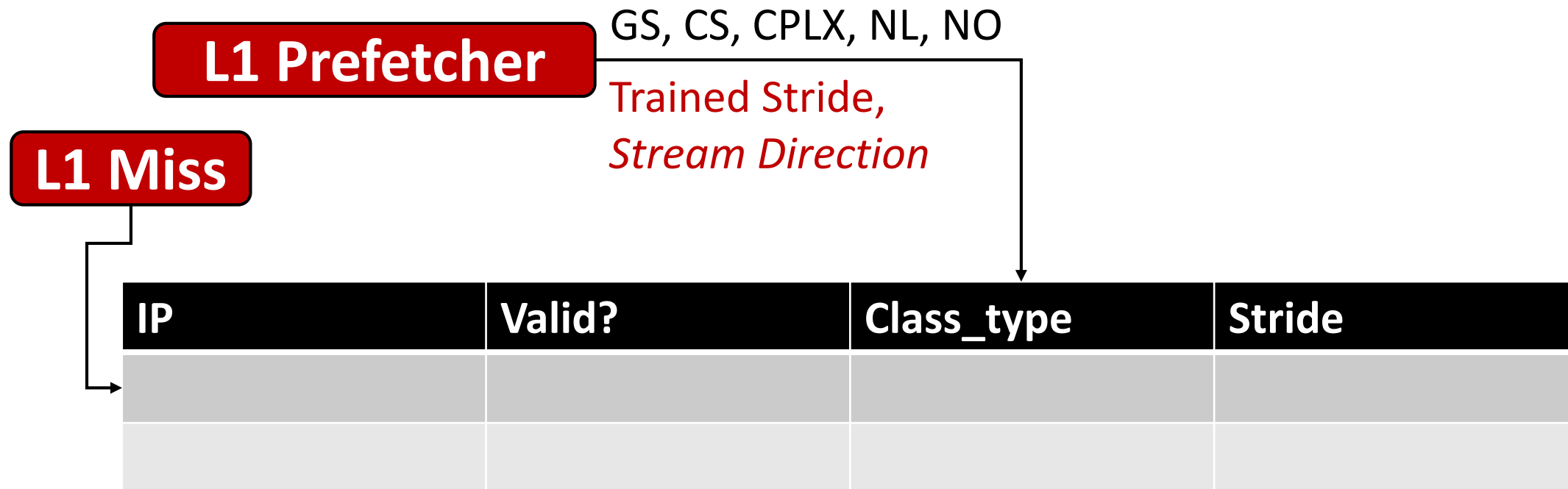
GS: **6**, CS and CPLX: **3**

Stride Confidence

Z
.
.
.
X

GHB

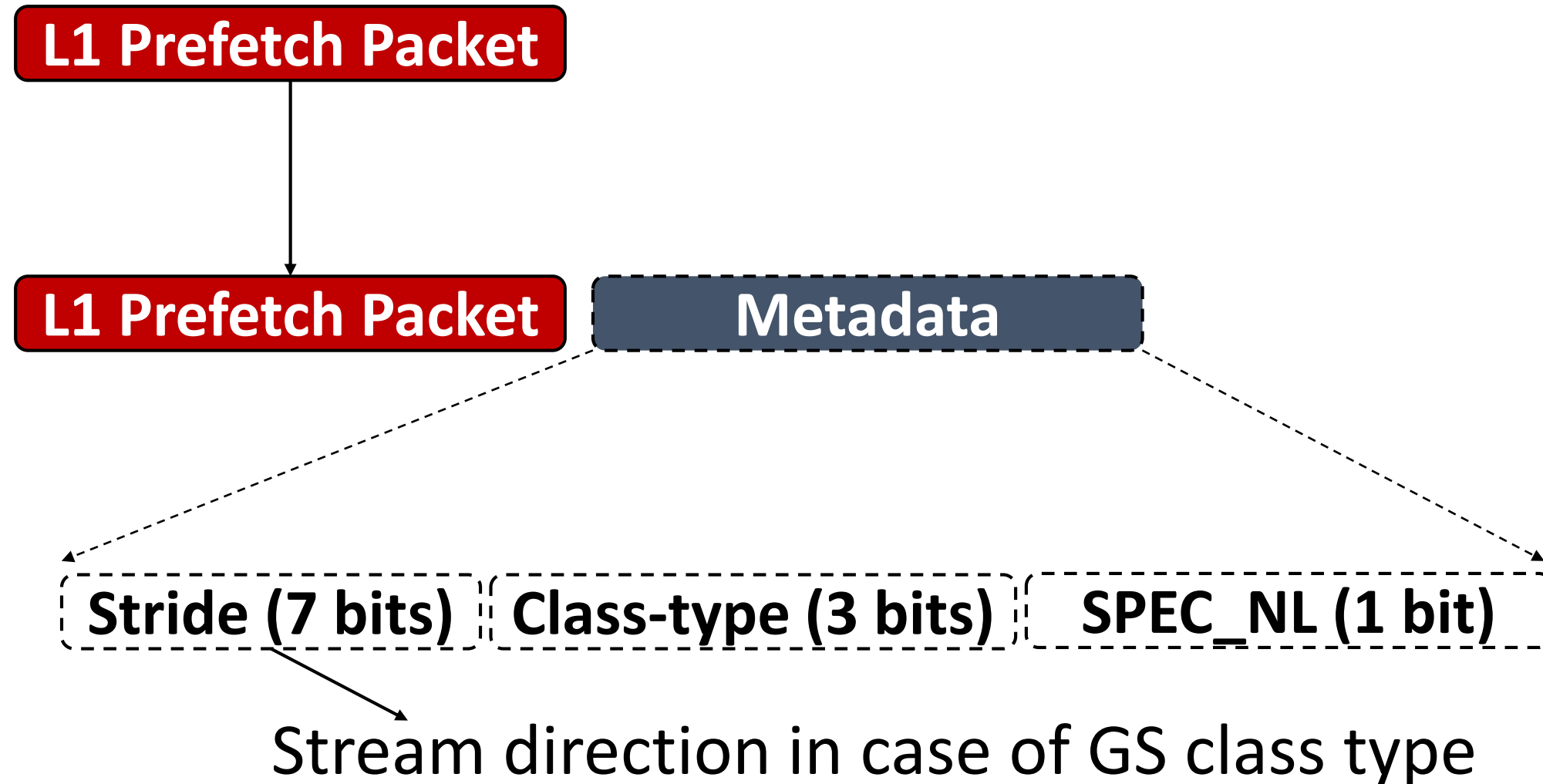
Our Proposal (IPCP at L2)



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

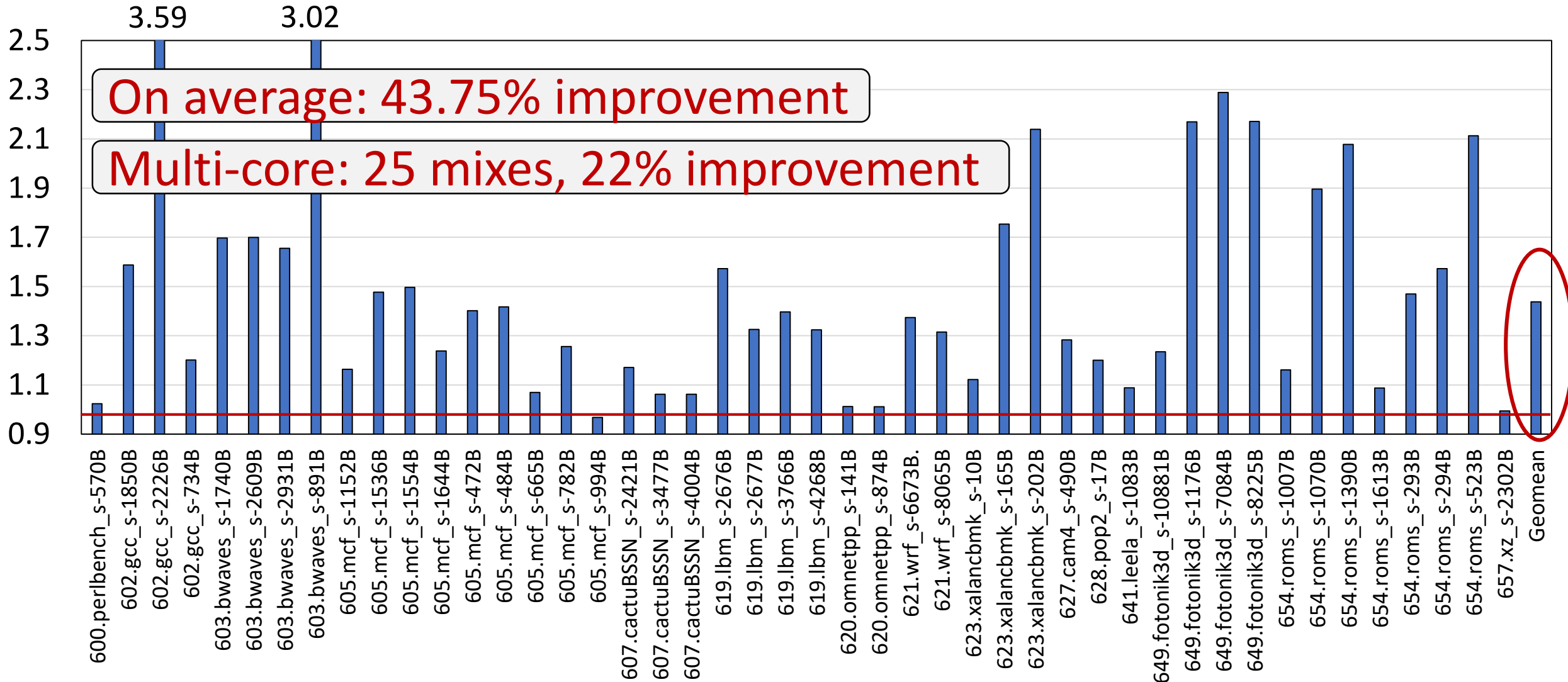
Metadata



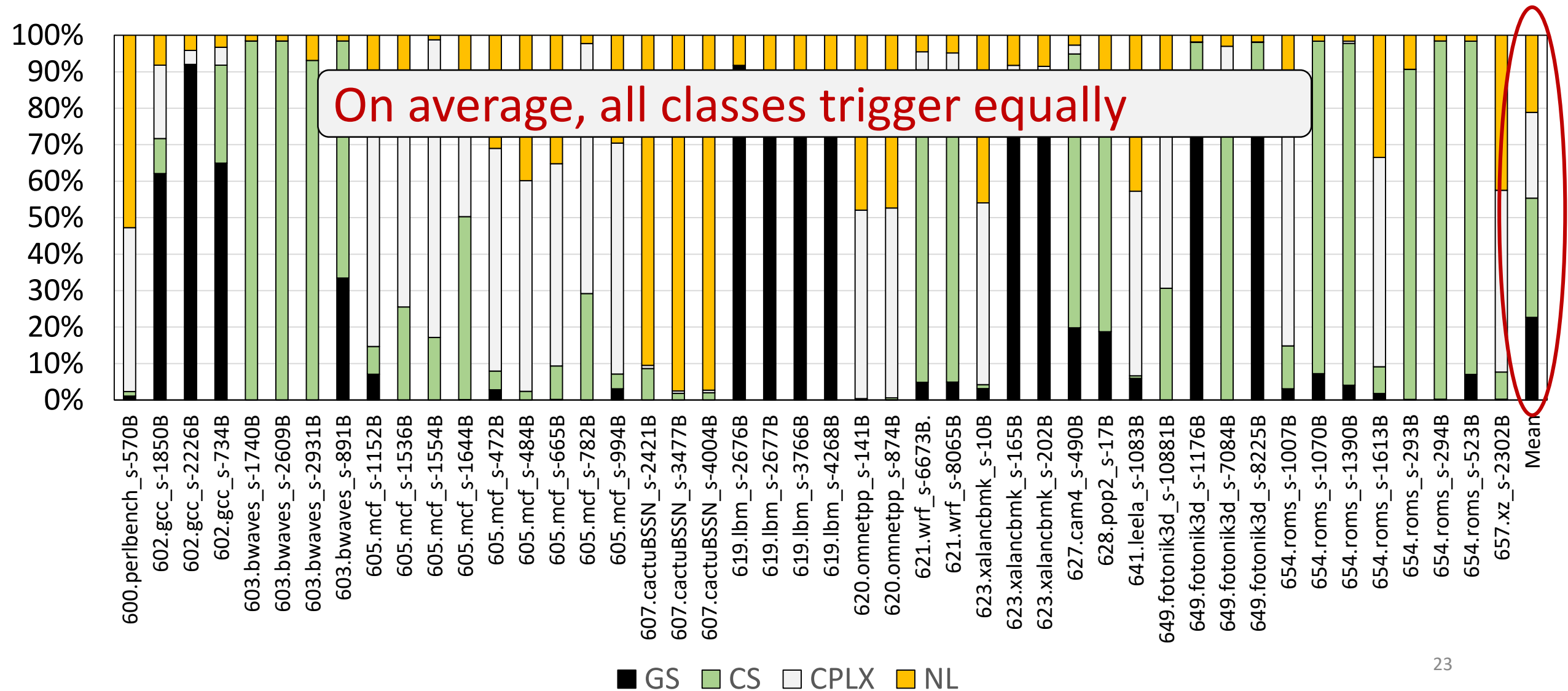
Hardware Overhead

Table	Entry size * #Entries	Total
IP Table	$77 * 1024 \text{ (L1)} + 17 * 1024 \text{ (L2) bits}$	12.03 KB
DPT Table	$9 * 4096 \text{ bits}$	4.6 KB
GHB Table	$16 * 58 \text{ bits}$	928 bits
Others	100 bits	86 bits
		16.7 KB

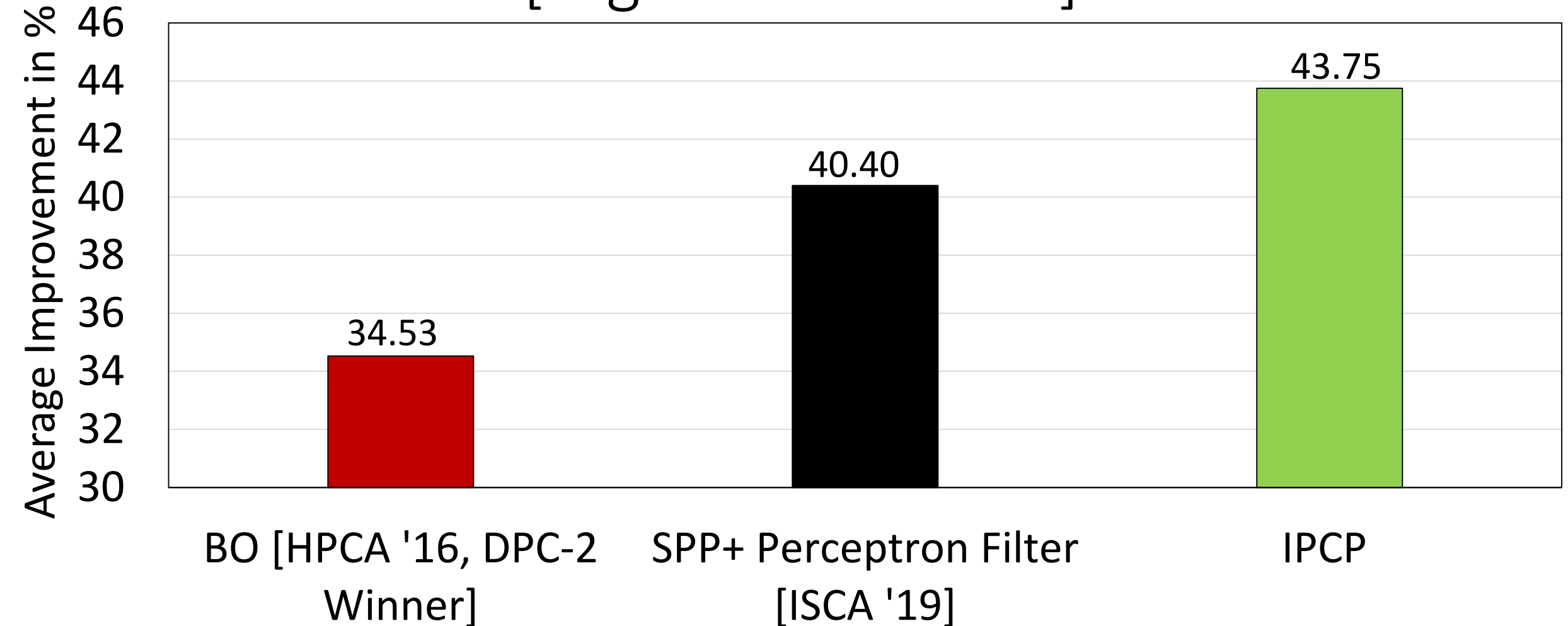
Single-core Performance [SPEC CPU 2017]



Distribution of IP Classes



Comparison with the State-of-the-art: Performance [Higher the better]



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

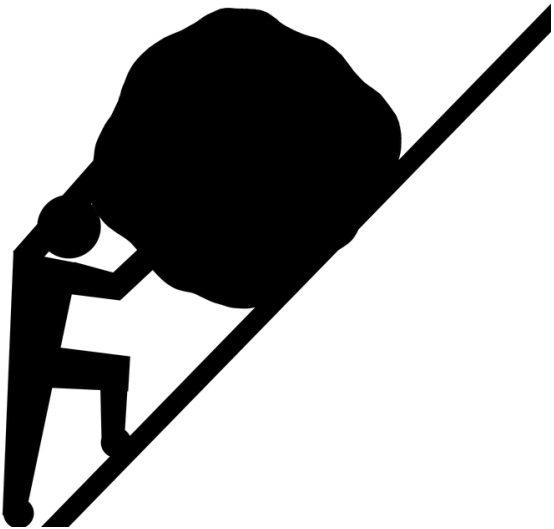
Prefetchers like **ISB [MICRO '13]** and **IMP [MICRO '15]** can be added to the bouquet seamlessly

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