AVM-BTB: Adaptive and Virtualized Multi-level Branch Target Buffer

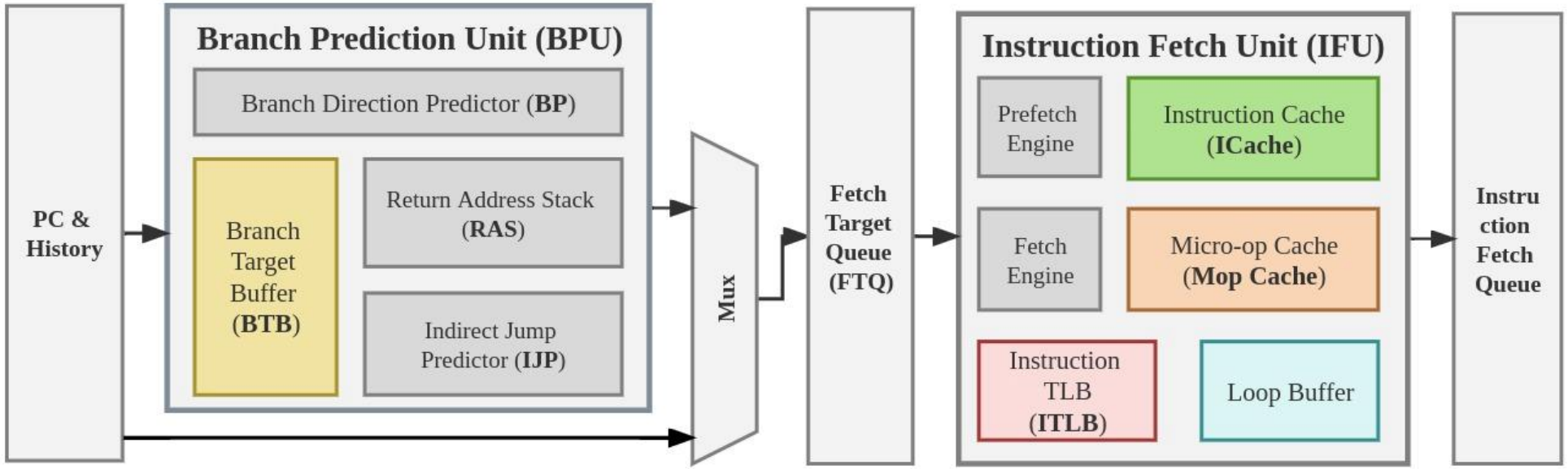
Yunzhe Liu[12], Xinyu Li[12], Tingting Zhang[13], Tianyi Liu[4], Qi Guo[1], Fuxin Zhang[12], Jian Wang[12]

1.State Key Lab of Processors, ICT, CAS
2.University of Chinese Academy of Sciences
3.Loongson Technology Co. Ltd.
4.The University of Texas at San Antonio

■ 概览

Background

Fetch directed instruction prefetch (FDIP)



Start IP: 0x00009ac
Len: 32Byte

Start IP	Len
0x00009a0	24
0x0001080	32
0x00010a0	56
0x00009ac	32

Fetch

Prefetch

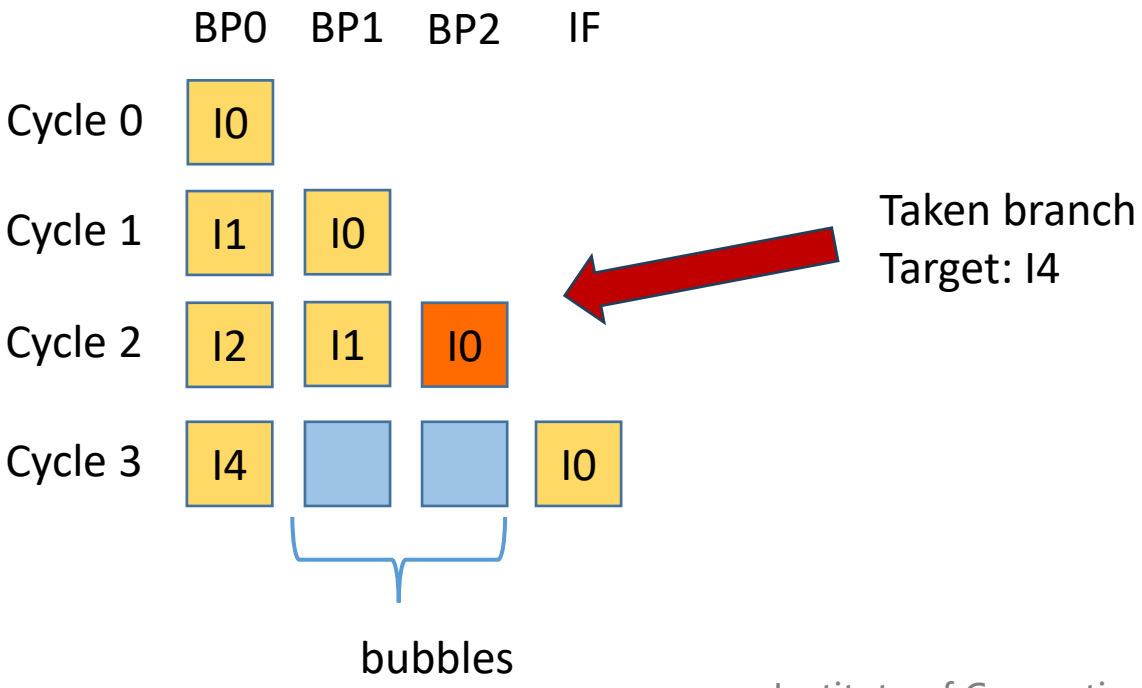
Background

Branch Target Buffer (BTB)

Valid: 1	Tag: 12	Type: 2	Target: 46	Rep_policy: 3

Cover once-taken branches

Identify branches & provide branch targets



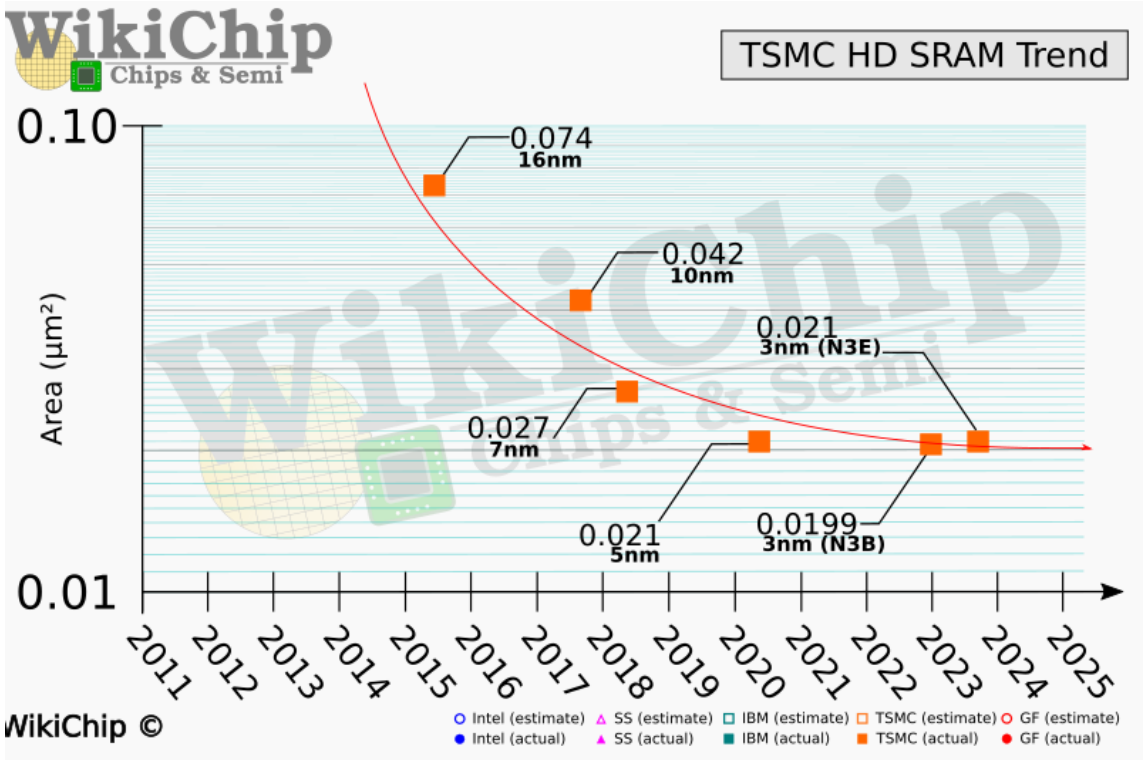
Research Problem & Challenge

- Limited SRAM & increasing demands of BTB

BRANCH PREDICTOR STORAGE, IN KBYTES

Bit storage (KB)	SHP	L1BTBs	L2BTB	Total
M1/M2	8.0	32.5	58.4	98.9
M3	16.0	49.0	110.8	175.8
M4	16.0	50.5	221.5	288.0
M5	32.0	53.3	225.5	310.8
M6	32.0	78.5	451.0	561.5

Grayson, Brian et al. "Evolution of the Samsung Exynos CPU Microarchitecture." 2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA) (2020): 40-51.



Research Problem & Challenge

- Limited SRAM & increasing demands of BTB
- Limited potential benefits of optimizing BTB itself

BTB optimization

- Enlargement
- Entry compression
- Replacement
- Prefetch
- Virtualization

Research Problem & Challenge

- Limited SRAM & increasing demands of BTB
- Limited potential benefits of optimizing BTB itself
- High implementation complexity

Methodology

Simulator: refactored trace-driven Champsim

Simulated Processor Configuration.

Component	Parameters
Core	6-wide OoO, 512-entry Re-order Buffer 128-entry FTQ, 128-entry Branch History Queue 256/256-entry INT/FP Physical Register File 96/70/38-entry INT&FP/LD/ST Scheduler 144-entry Load Queue, 128-entry Store Queue 5 ALU, 2 BRU, 3 FPU, 3 LD-AGU, 2 ST-AGU
BPU	64KB Tage-SC-L, 4096-entry ITAGE, 32-entry RAS 64-entry fully-associative L0BTB 1024-entry 2-way L1BTB Max branch predicted per cycle: 2 Max taken branch predicted per cycle: 1
Uop Cache	4096-entry 8-way, LRU, 8 uops/cycle, Uop Size: 56-bit, Max uops per way: 8 Imm operand size: 32-bit, Max imm per way: 4
TLBs	64-entry 4-way ITLB, 64-entry 4-way DTLB 1.5K-entry STLB
Caches	64KB 8-way L1I (VIPT), 48KB 12-way L1D (VIPT) 2MB 16-way L2C, 8MB 32-way LLC

Simulated BTB Configuration.

Name	Unique configuration
Base	4K-entry 4-way L2BTB
Base-8K	8K-entry 8-way L2BTB
Base-12K	12K-entry 12-way L2BTB
AVM-BTB	4K-entry 4-way L2BTB + AVM-BTB
PDede	6K-entry PDede as L2BTB
BTB-X	8K-entry BTB-X as L2BTB
Confluence	4K-entry 4-way L2BTB + Confluence
Shotgun	1.5K-entry 4-way C-BTB + 5K-entry 4-way U-BTB as L2BTB
Phantom BTB	4K-entry 4-way L2BTB + 24K-entry Phantom BTB in LLC

Methodology

Simulated Traces:

- IPC-1 : client, server, and SPEC.
- CVP-1 : compute, crypto, and server.
- Frontend-bound (FEB) workloads: tpcc, wikipedia, finagle-http, finagle-chirper, verilator, kafka and tomcat.

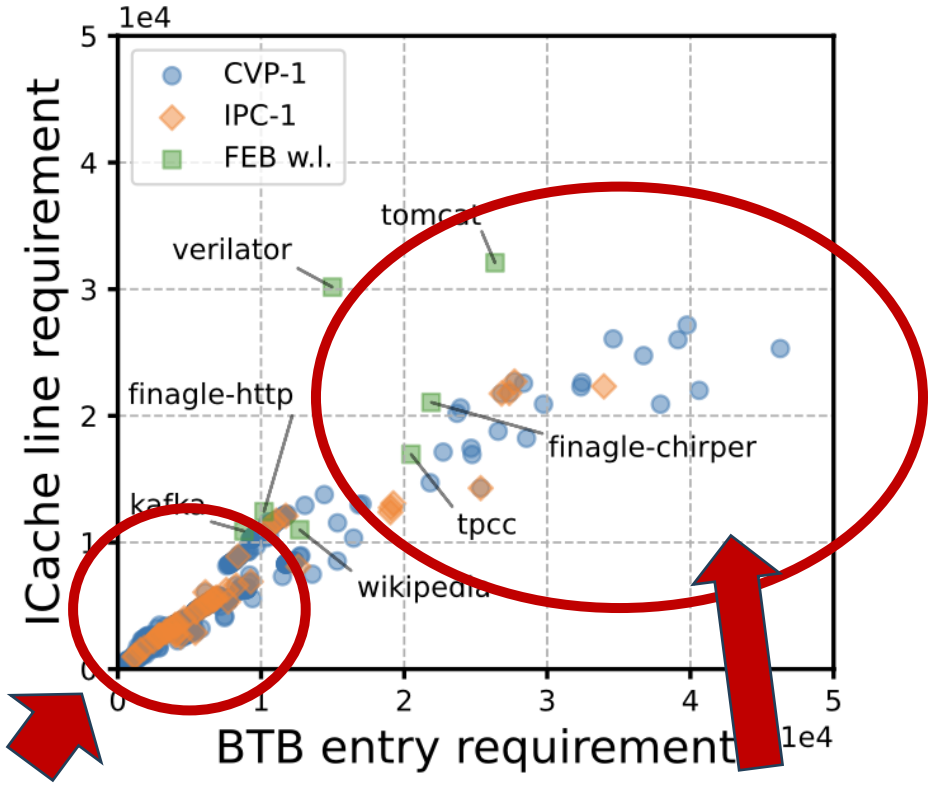
Workload	ICache MPKI	L2Cache MPKI	Branch MPKI
tpcc	17.46	2.24	8.14
wikipedia	15.23	2.66	7.65
finagle-chirper	28.12	3.46	16.18
finagle-http	34.33	1.69	15.81
kafka	3.11	5.25	0.68
tomcat	19.80	4.29	12.83
verilator	50.46	20.45	25.22
CVP_compute	0.27	6.83	5.63
CVP_crypto	0.50	1.24	0.74
CVP_server	39.08	1.81	12.32
IPC_client	4.20	3.46	3.81
IPC_server	28.87	4.95	8.12
IPC_SPEC	2.54	22.55	3.02

Observation 1: BTB Requirement

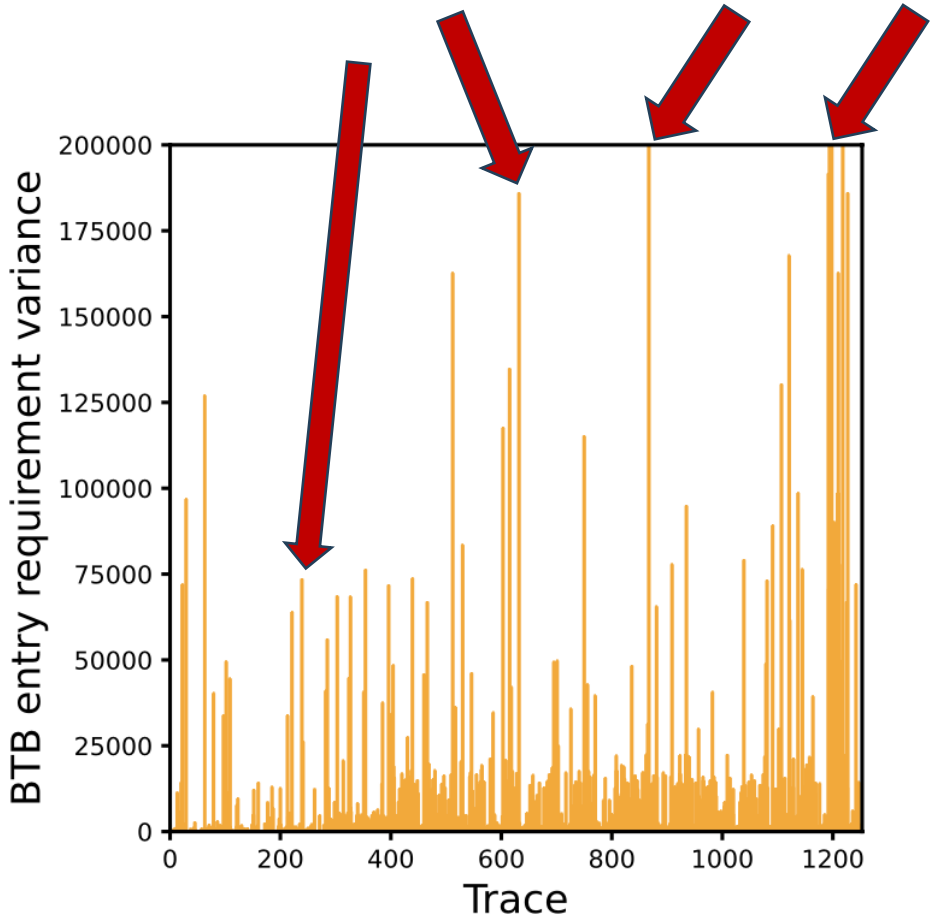
Observation 1: BTB requirements vary significantly among different applications and even different running stages of the same application. A larger BTB does not yield significant performance benefits in half of the traced scenarios.

Observation 1: BTB Requirement

Observation 1: BTB requirements vary significantly among different applications and even different running stages of the same application. A larger BTB does not yield significant performance benefits in half of the traced scenarios.



Requiring fewer than 2k-entry

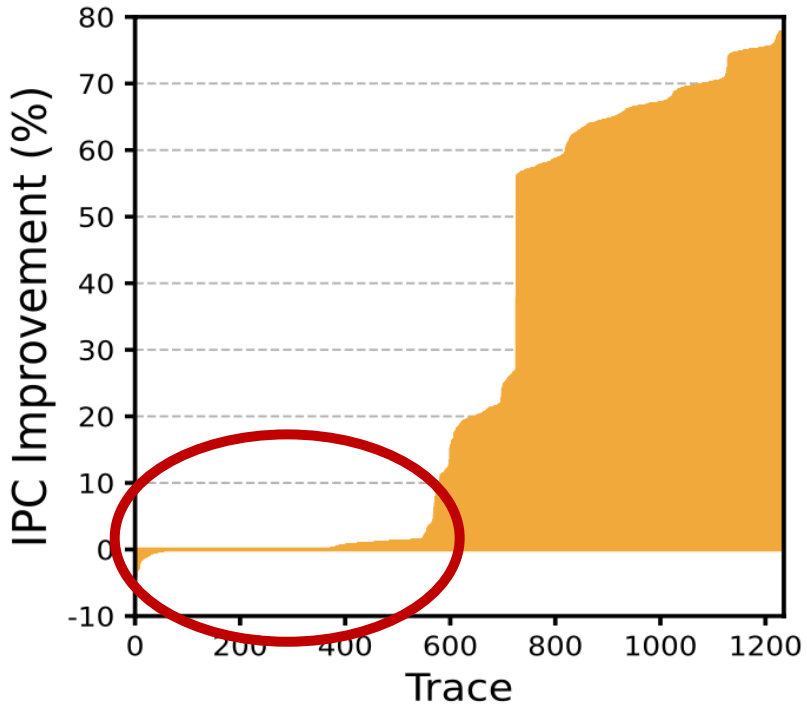


Requiring more than 20k-entry

Observation 1: BTB Requirement

Observation 1: BTB requirements vary significantly among different applications and even different running stages of the same application. A larger BTB does not yield significant performance benefits in half of the traced scenarios.

Insight 1: Static BTB design makes it difficult to meet the variable requirements of applications. The ability to dynamically adjust the BTB capacity is preferable.



Observation 2: ICache versus BTB

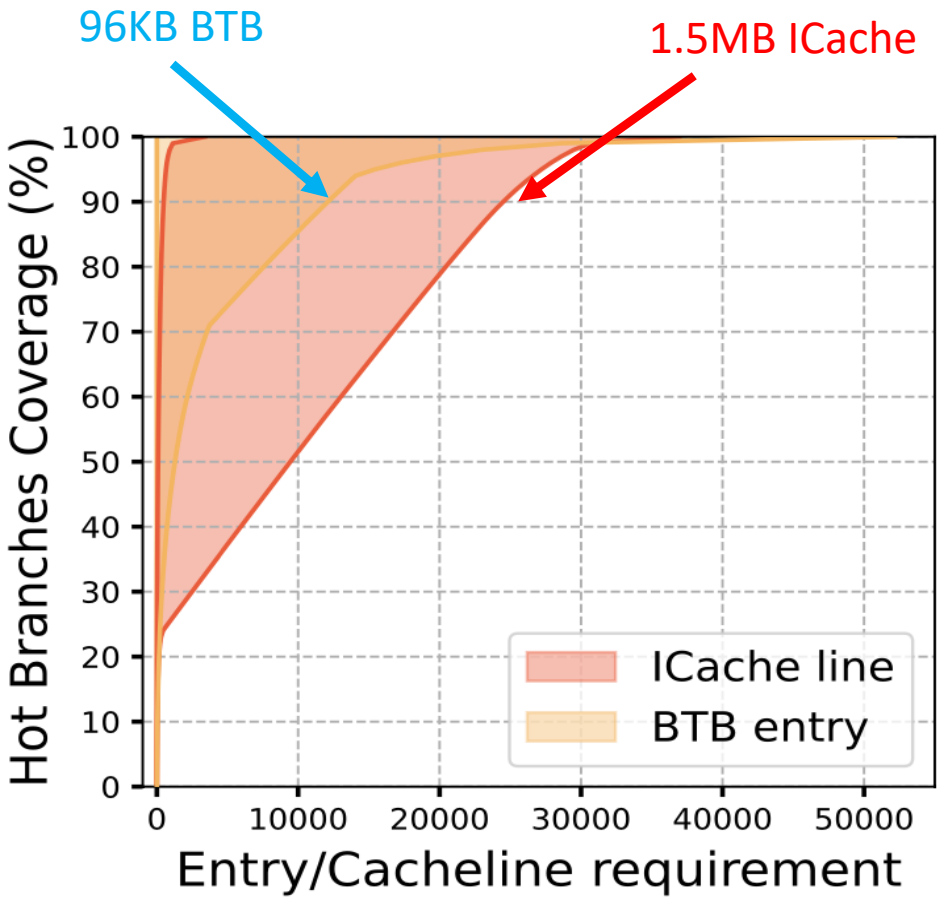
Observation 2 : In frontend overloaded scenarios, the performance of BTB is more critical than that of ICache. Moreover, improving the hit rate of BTB by increasing capacity incurs less overhead compared to enlarging ICache.

Observation 2: ICache versus BTB

$$N_{BTBentry} = N_{instruction} \times f_{branch} \times P_{branchTaken}$$

18.83% 49.53%

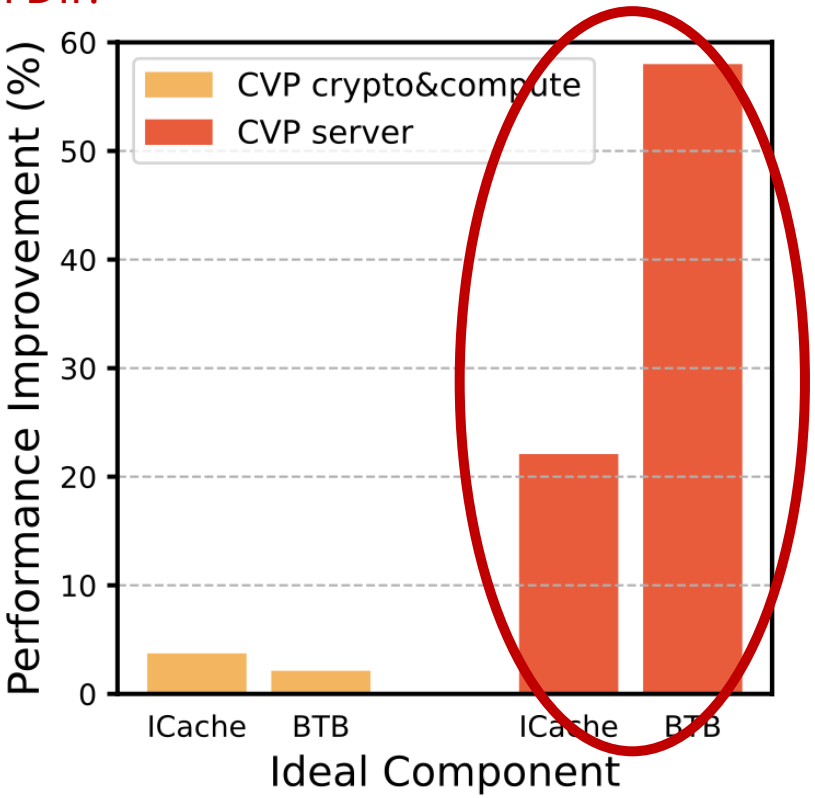
9.3%



Observation 2: ICache versus BTB

Observation 2 : In frontend overloaded scenarios, the performance of BTB is more critical than that of ICache. Moreover, improving the hit rate of BTB by increasing capacity incurs less overhead compared to enlarging ICache.

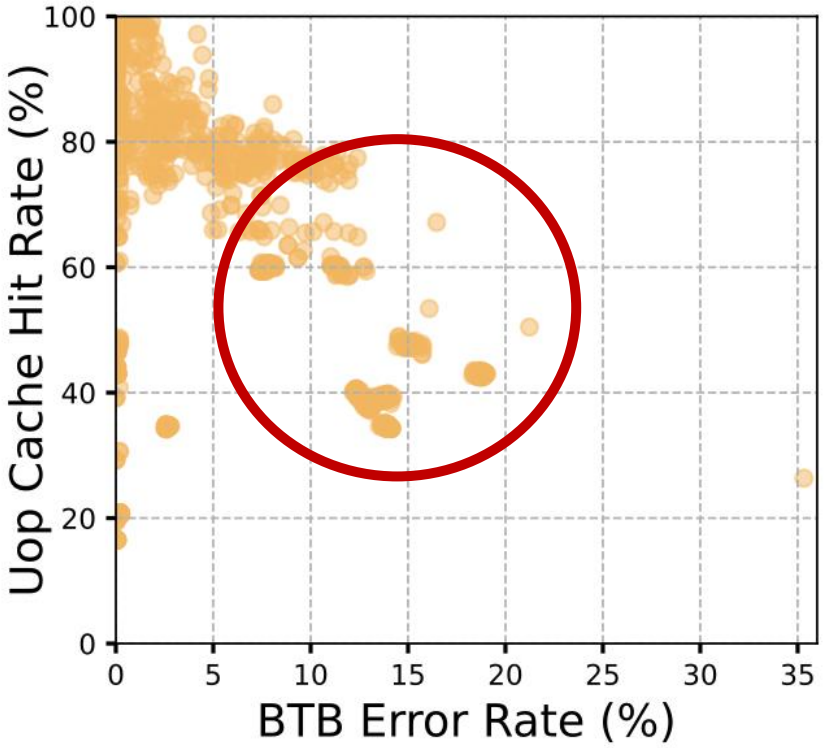
Insight 2 : Finding a dynamic capacity trade-off between the BTB and ICache holds the potential to guarantee a higher BTB hit rate, consequently minimizing branch prediction errors while maximizing the prefetch benefits of FDIP.



Observation 3: Uop Cache versus BTB

Observation 3 : Uop Cache exhibits inefficiency when confronted with large instruction footprints.

Insight 3 : In scenarios where Uop Cache demonstrates a low hit rate, repurposing it should be considered as a means to enhance the overall performance.



Observation 4: BTB metrics

Observation 4 : A high BTB error rate indicates that there is insufficient BTB capacity. Significantly fewer effective accesses to the lower-level BTB compared with the upper-level BTB indicates the satisfaction of BTB demands.

Insight 4 : BTB metrics reflect application requirements and can be used to guide dynamic adjustment in BTB capacity.

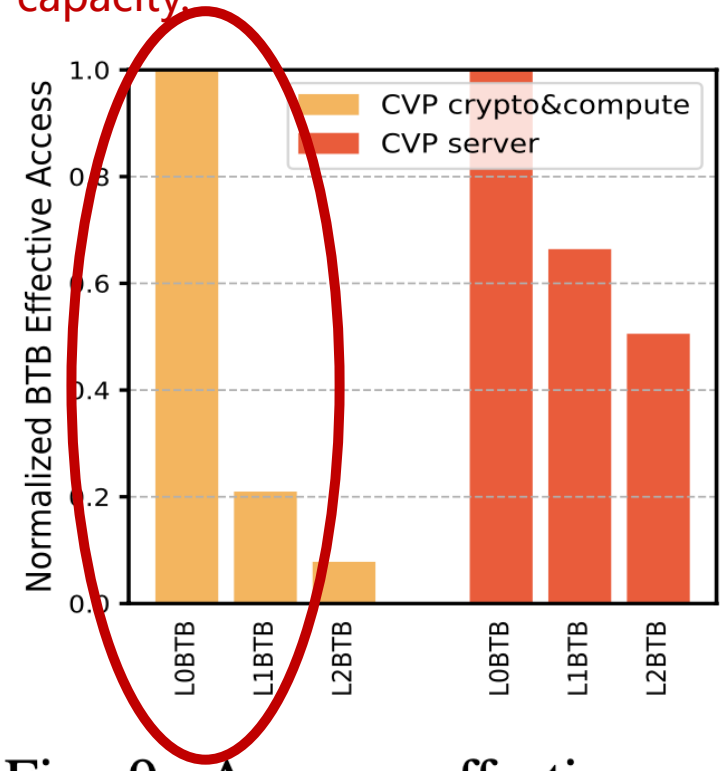
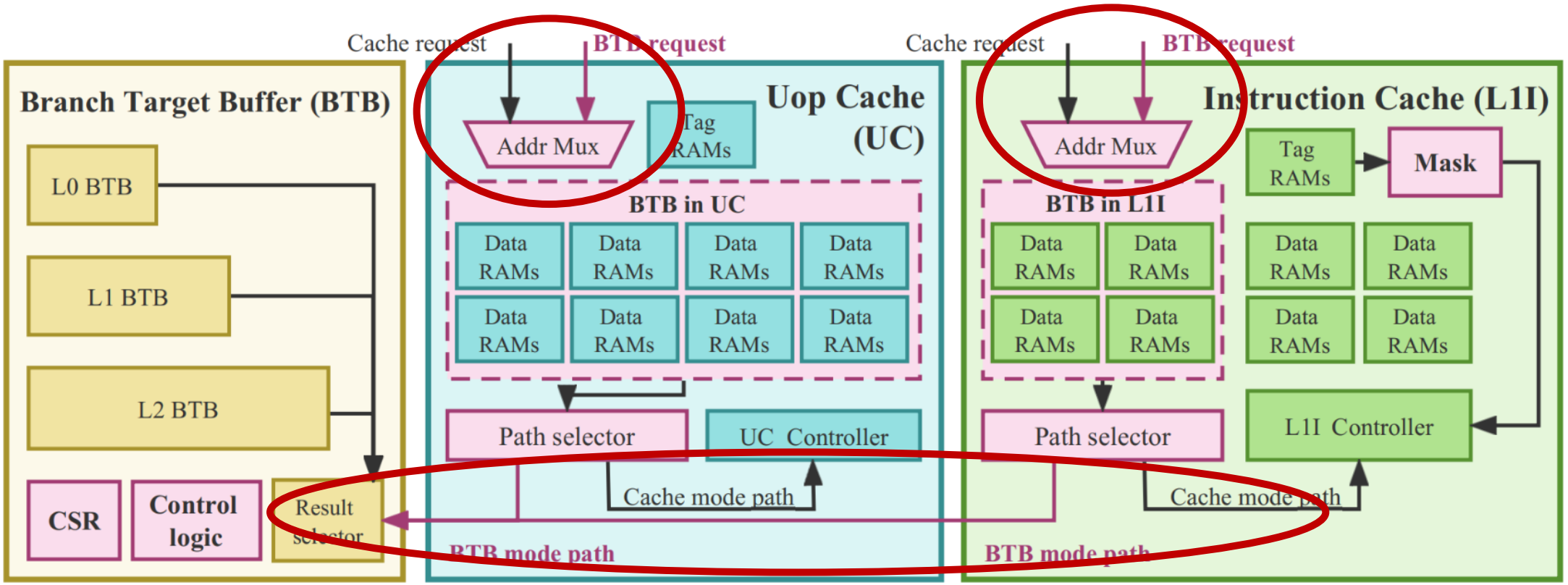


Fig. 9: Average effective access to each level of BTB.

AVM-BTB Design: BTB in Cache

AVM-BTB dynamically borrows on-chip SRAM from UC and L1I as temporary BTB and returns these SRAM when BTB demand decreases.

Hardware metrics collected include the BTB error rate (errRate) and effective access (efAccss).



AVM-BTB Design: Mode Switch Strategy

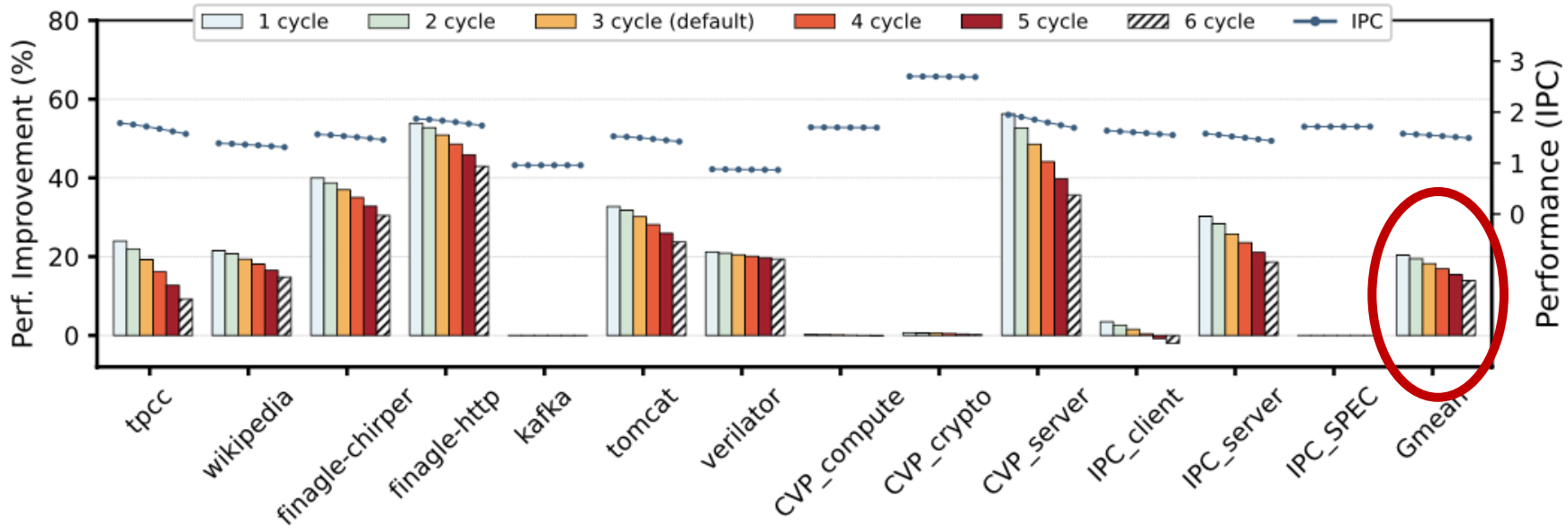
Algorithm 1: BTB switch strategy.

```
1  if errRate > 1% then
2      if saturation_bit_for_activation == 0 then
3          saturation_bit_for_activation = 1;
4          saturation_bit_for_deactivation = 0;
5      else
6          if BTB in UC is activated then
7              activate_BTB_in_L1I();
8          else
9              activate_BTB_in_UC();
10         end
11     end
12 else
13     if saturation_bit_for_deactivation == 0 then
14         saturation_bit_for_deactivation = 1;
15         saturation_bit_for_activation = 0;
16     else
17         if BTB in L1I is activated then
18             if efAccssBTBinUC > 6 * efAccssBTBinL1I
                 then deactivate_BTB_in_L1I();
19         else
20             if efAccssL2BTB > 6 * efAccssBTBinUC then
                 deactivate_BTB_in_UC();
21         end
22     end
23 end
```

Evaluation

Area & timing

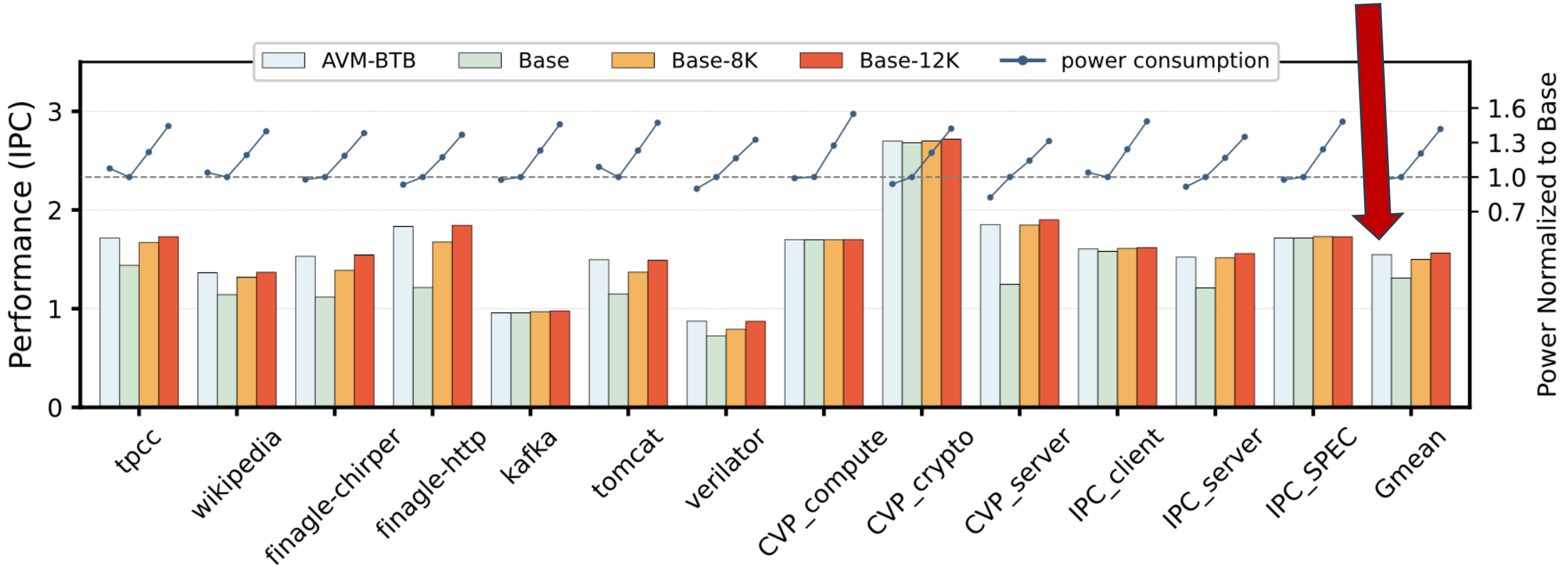
	Area compared with Base
AVM-BTB	+ > 0.1%
8K-entry L2BTB	+ 88.9%
12K-entry L2BTB	+ 186.46%



Evaluation

Performance: exceeds respective 18.22% and 3.26% over Base (4K-entry L2BTB) and Base-8K (8K-entry L2BTB) and achieves 99.12% of the performance of Base-12K (12K-entry L2BTB)

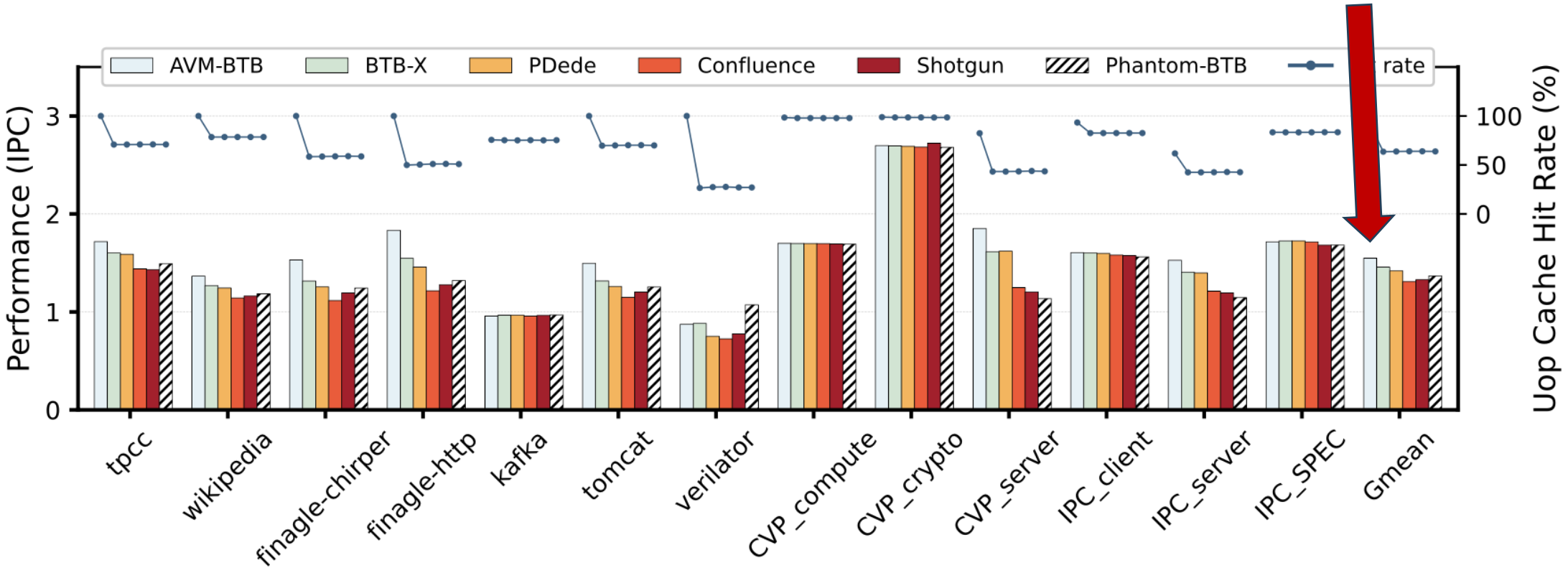
Power: achieves a power reduction of 2.77%, 19.28% and 31.40%, respectively, over Base, Base-8K and Base-12K



Evaluation

Performance: surpasses BTB-X, PDede , Confluence, Shotgun, and Phantom-BTB (PBTB) by 6.22%, 9.11%, 18.26%, 16.53% and 13.19%, respectively.

BTB-X, PBTB, and AVM-BTB demonstrate performance enhancements of 11.29%, 4.47%, and 18.22%, respectively. When BTB-X and PBTB are combined with AVM-BTB, they can achieve improvements of 23.34% and 19.66%, respectively.



Conclusion

- Multi-dimensional characterization of the modern applications' frontend requirements.
- An adaptive and virtualized multi-level BTB design, that caters to diverse and evolving computing environments .
  - 18.22% performance boost
  - 2.77% power consumption reduction
  - Limited area overhead

AVM-BTB: Adaptive and Virtualized Multi-level Branch Target Buffer

Yunzhe Liu[12], Xinyu Li[12], Tingting Zhang[13], Tianyi Liu[4], Qi Guo[1], Fuxin Zhang[12], Jian Wang[12]

1.State Key Lab of Processors, ICT, CAS
2.University of Chinese Academy of Sciences
3.Loongson Technology Co. Ltd.
4.The University of Texas at San Antonio