BINGO PREFETCHER

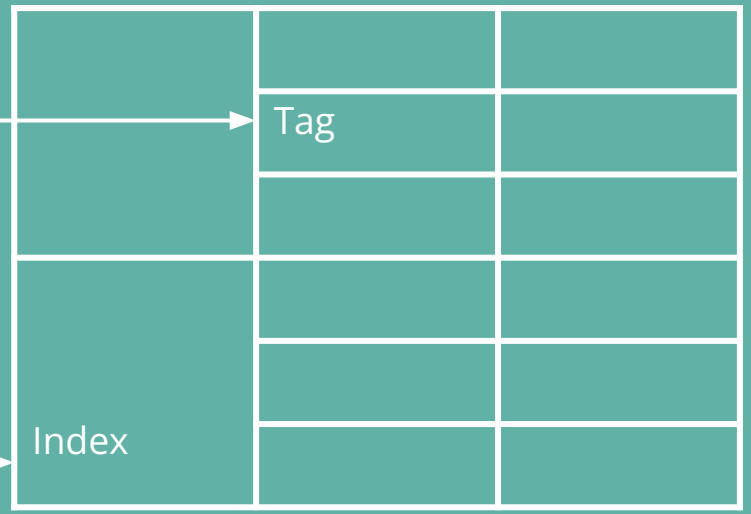
- Team Pipeliners

Insights: What and Why?

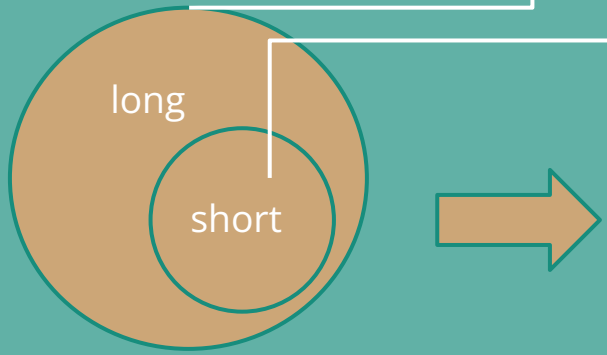
- Exploits the similarity of access patterns among memory pages to prefetch
- Associates them with both longer and shorter events

What and Why? (Slide 2)

- One history table for both events? … redundancy

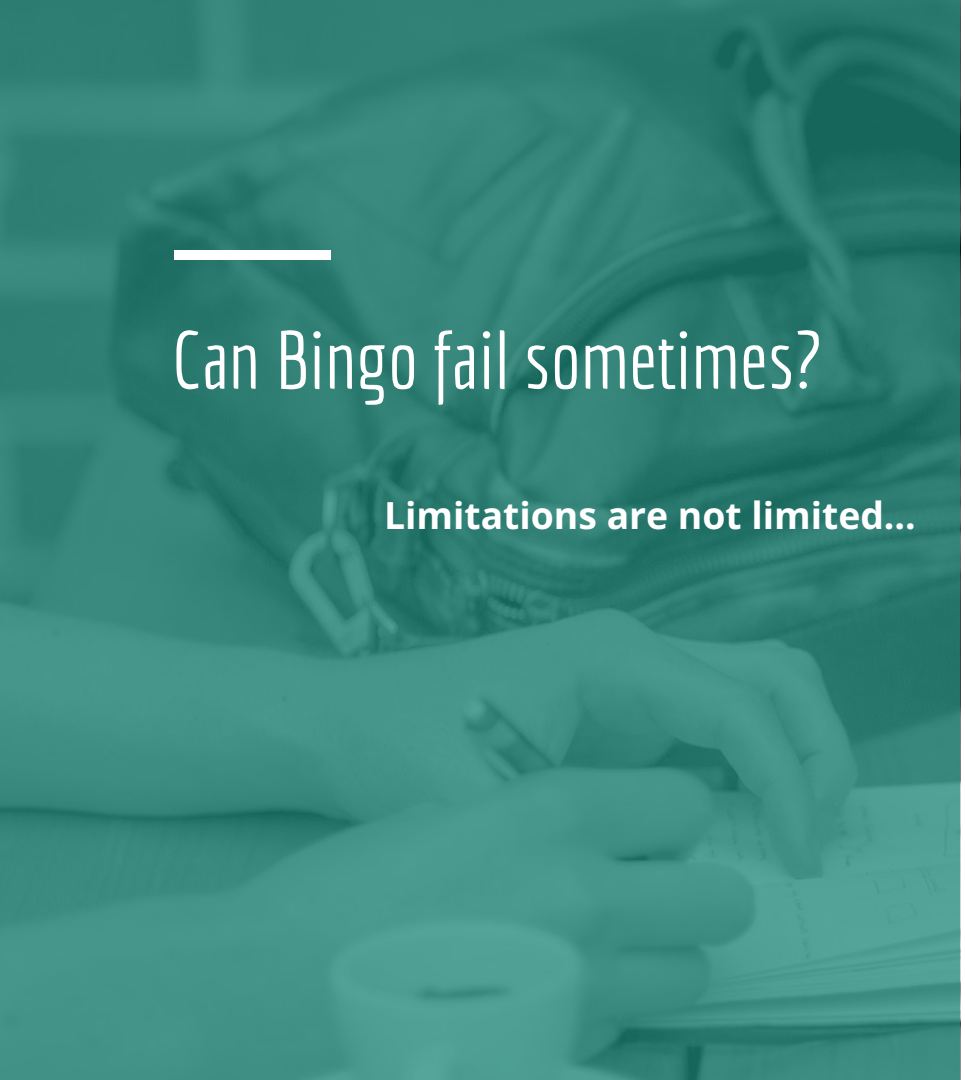


"Short events being carried out in long events"



Redundant information

Less execution time and reducing storage overhead - **BETTER RESULTS!**



Can Bingo fail sometimes?

Limitations are not limited...

- Is the hash function used to encode “PC + offset” efficient?
- Is the heuristic used for handling multiple footprints of same "PC + offset" fair?

JAL experiments...

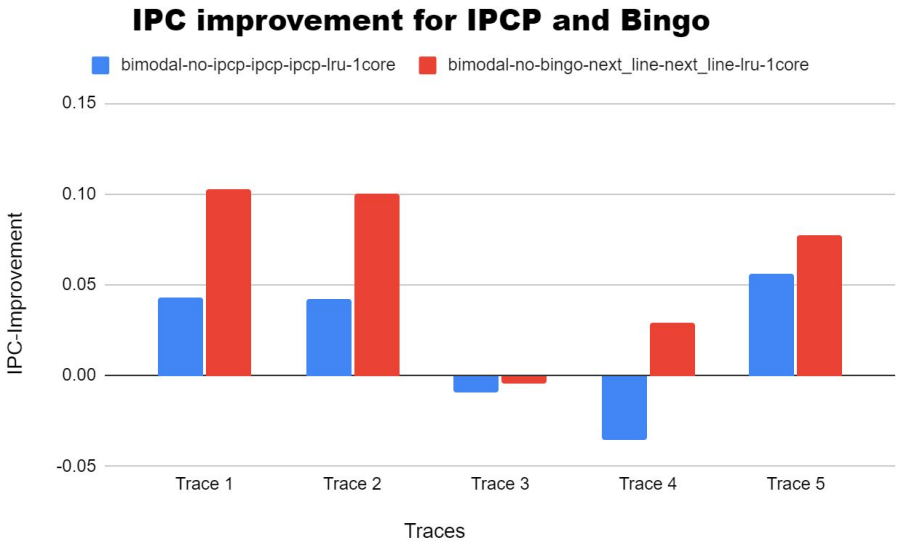
Low five

Traces with least improvement for IPCP

1. 620.omnetpp_s-141B.champsimtrace.xz
2. 620.omnetpp_s-874B.champsimtrace.xz
3. 657.xz_s-2302B.champsimtrace.xz
4. 605.mcf_s-994B.champsimtrace.xz
5. 600.perlbench_s-570B.champsimtrace.xz

Bingo & IPCP VS Baseline

Warmup - 50M instructions
Simulation - 200M instructions



Bingo VS IPCP

- Bingo > IPCP for all traces
- Why 🤔🤔💬
  - IPCP uses fixed access patterns - constant & complex stride or streaming
  - IPCP uses fixed range of prefetch degree
  - Bingo uses both longer and shorter events
  - Bingo - 16 PC bits in index (>IPCP) with help of hashing

Bingo VS Baseline

IPC increases in general

...Is that always the case??

Bingo VS Baseline[2]

- Trace 3 - Bingo performs worse than baseline
- Why 🤔🤔💬
  - unfavourable evictions
  - unfavourable prefetches
  - benchmark might not be reusing the same data


BINGO
Let’s play with some numbers!!
B
I
N
G
O

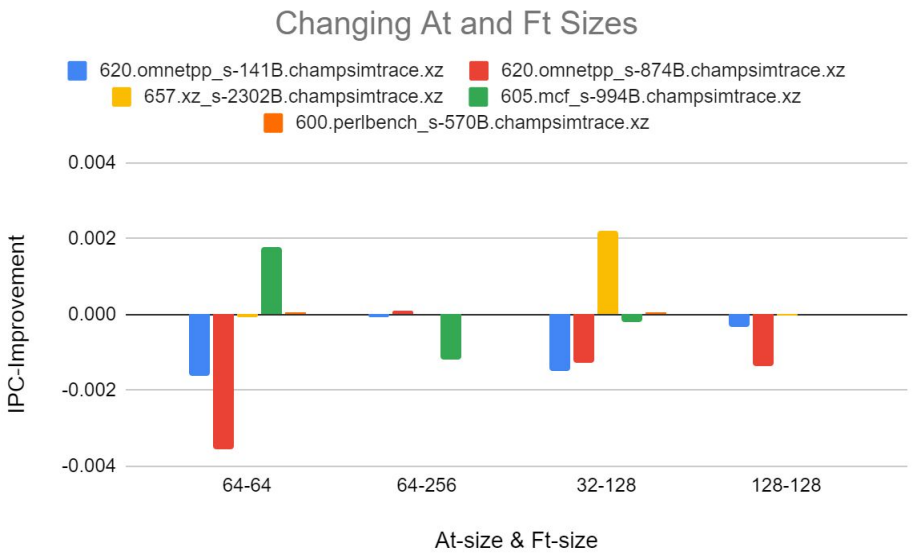
Experiments

- 10M Warm-up instructions, 10M Simulation instructions
- For each modification, we plot IPC improvement (= $\frac{\text{IPC with modified Bingo}}{\text{IPC with default Bingo}}$ - 1)

- Modified Parameters
  - AT-size & FT-size
  - L1D_Thresh & L2C_Thresh

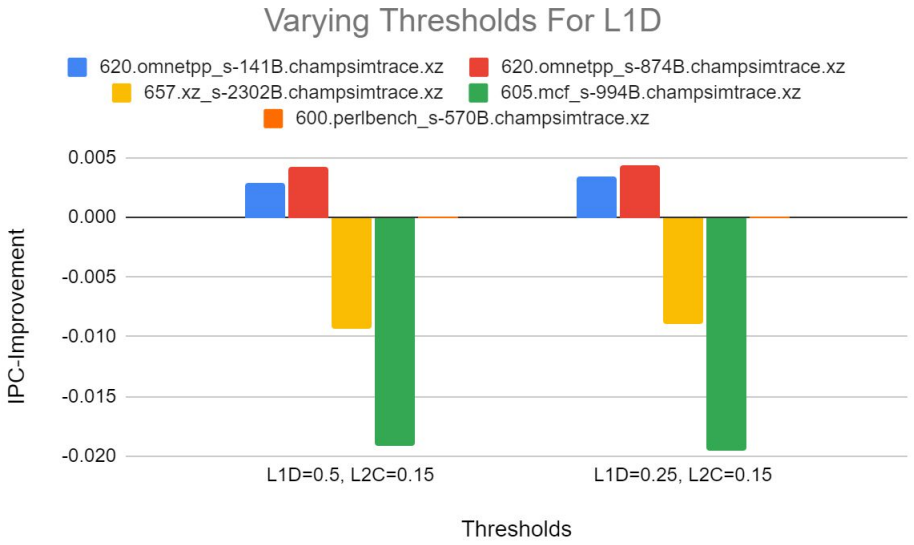
AT-size and FT-size

Default Accumulation table size - 64
Default Filter Table size - 128



L1D & L2C Thresh

Default L1D threshold - 0.75
Default L2C threshold - 0.25



Decrease in threshold →
Prefetches increase but relevance decreases: opposing effects

We tried few more...

None of them actually worked :(

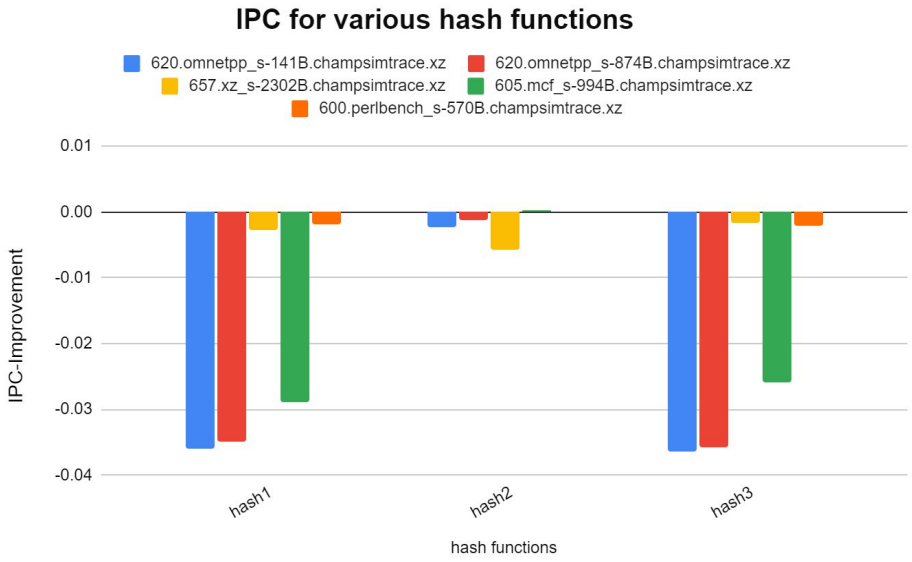
Let us try something new.....

What if we change the Hash function?

hash 1 - https://stackoverflow.com/a/12996028

hash 2 - https://www.modernescpp.com/index.php/hash-functions

hash 3 - https://stackoverflow.com/a/57556517



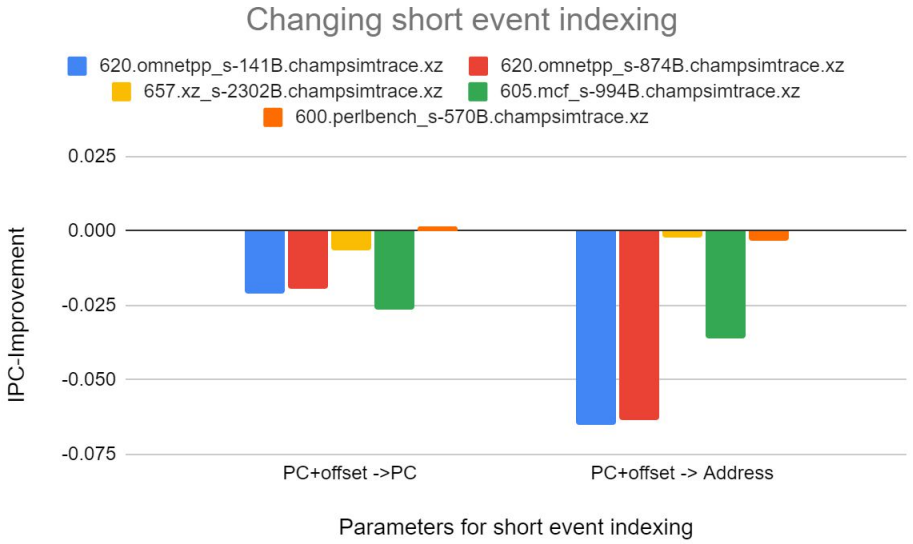
Default hash function outperforms
- ***condenses info from entire key***

Possible Short events

Default indexing - PC+offset for short events

Replaced with

- PC
- Address



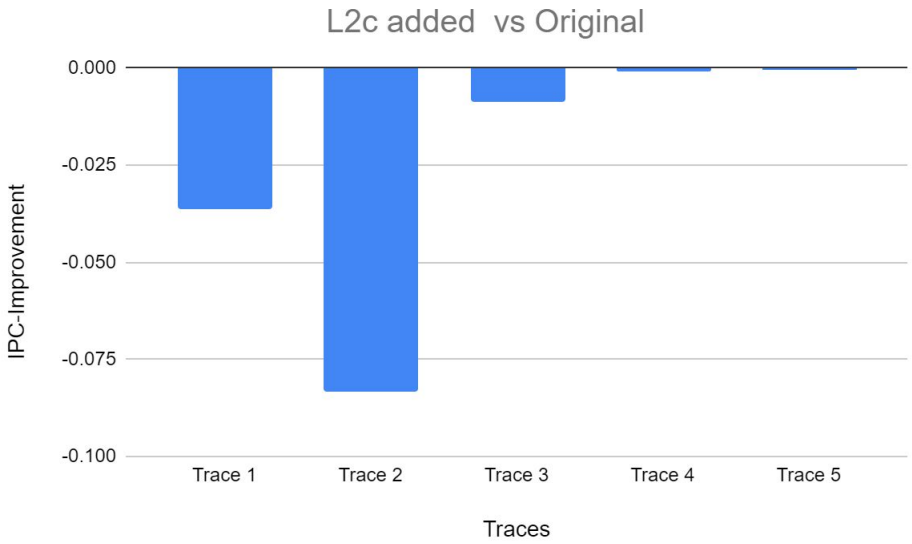
PC indexing decreases accuracy, increases match probability
Address indexing decreases accuracy and match probability

Failed Again……. :(

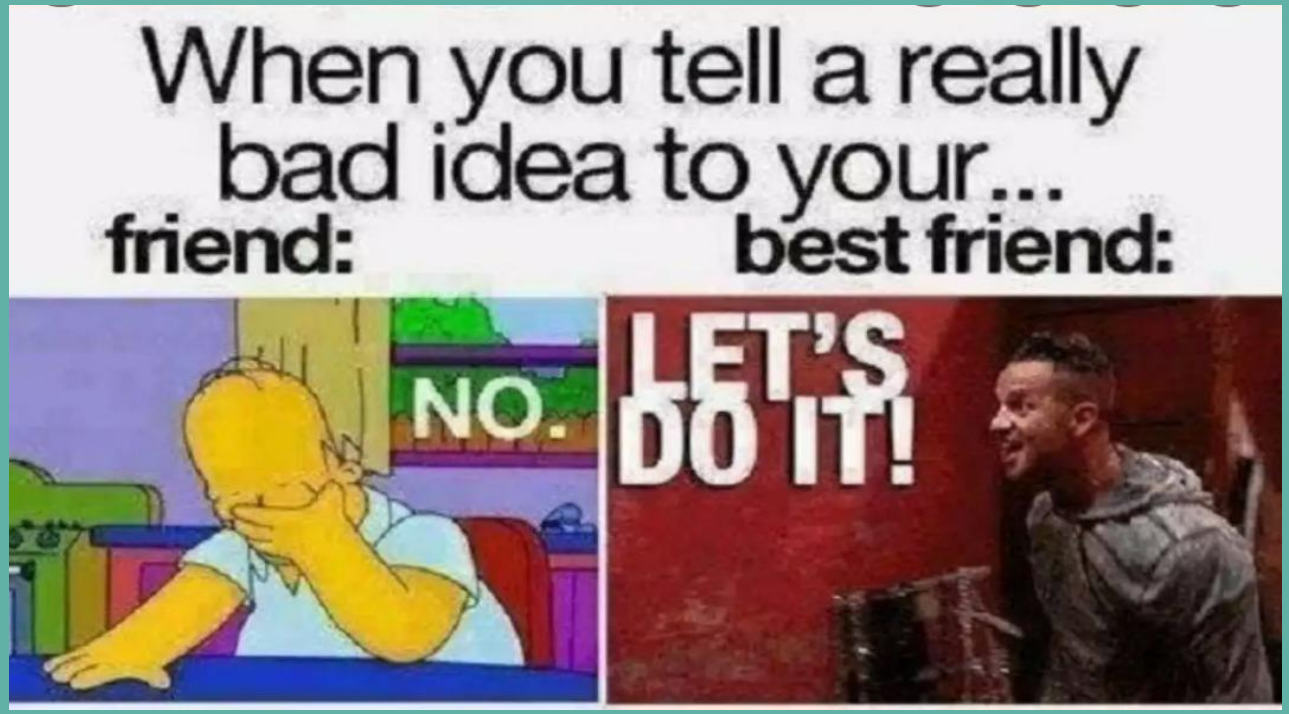
But we won’t give up :)

Bingo for L2C.....

L2C with Bingo parameters same as L1D

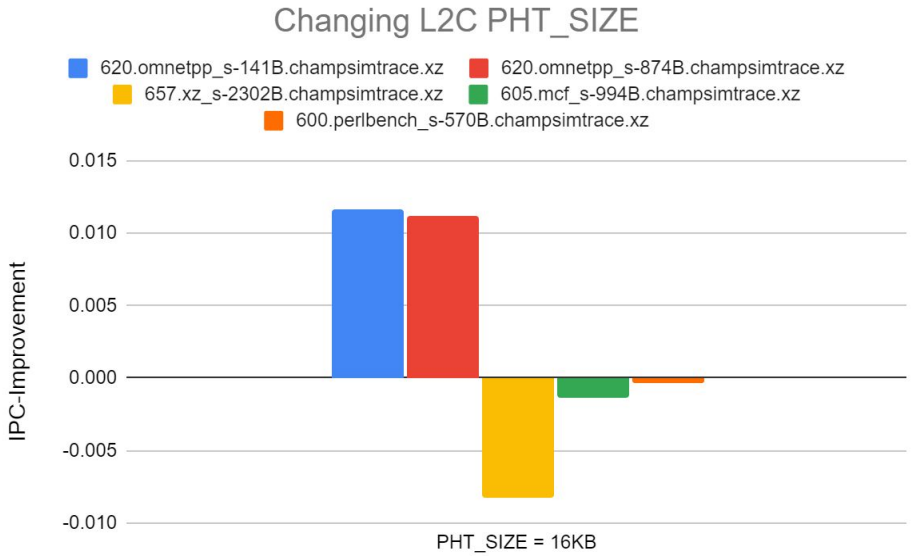


Restart the number-game....



L2C PHT Size

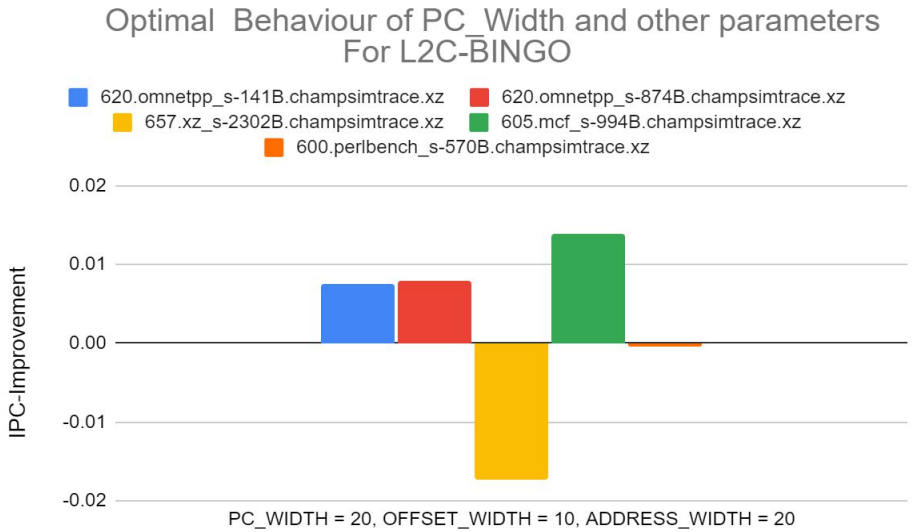
Greater Pattern History Table(PHT) size for greater capacity



Able to store more history - might increase IPC

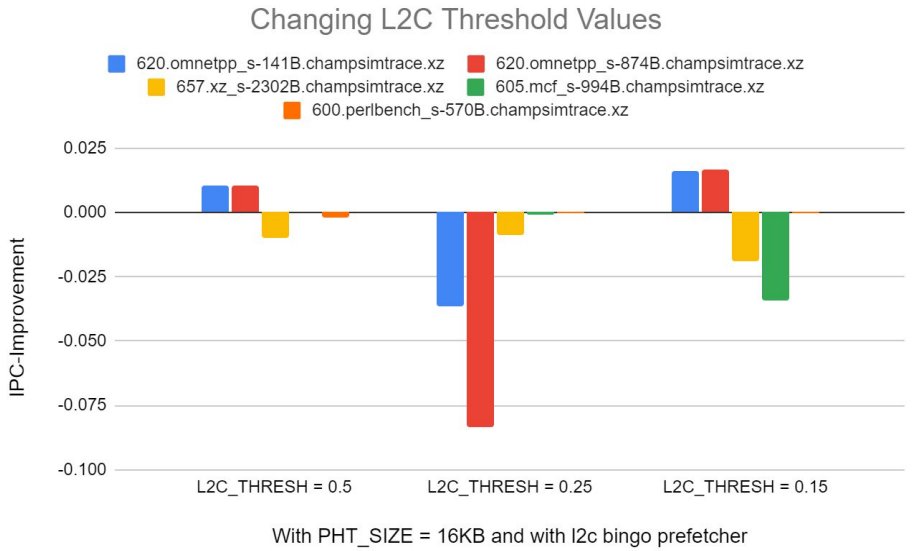
L2C Index and tag bits

Greater number of bits for extracting more information



More information → better prefetches

L2C Threshold in L2C prefetcher

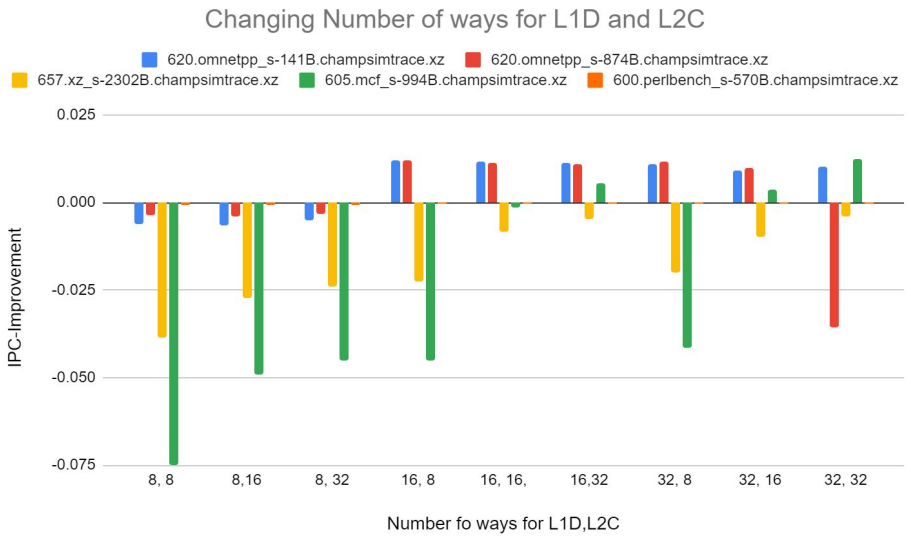


Decrease in threshold →

Prefetches increase but relevance decreases: opposing effects

L1D and L2C ways

Default PHT ways in L1D - 16

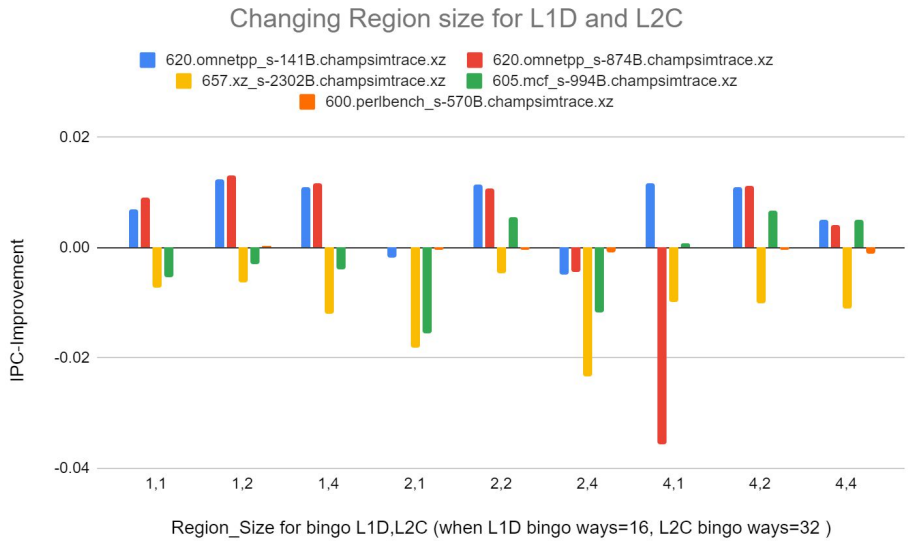


Increase in ways →
Match probability for longer events increase

L1D and L2C Region Size

Default Region size for L1D - 2KB

Large region size contributes to large pattern length



Increasing pattern length →
prefetching time increases and number of useful prefetches increases - opposing factors

Conclusions

Tuning table sizes - AT, FT, PHT

No improvement on either increasing or decreasing table sizes

... default sizes are already optimal

Tuning L1D and L2C threshold

Improves for some traces and decreases for others

... increased prefetches, decreased relevance opposing each other

Exploring different short events

No improvement

... PC + offset might be optimal

Trying out different hash functions

No improvement

... default hashing might be optimal

Conclusions[2] ⟶ Extending Bingo to L2C

Played with PHT size, Index and Tag bits, number of ways and region size

On an average, best improvement obtained for

- PHT size = 16K, Num ways = 32 & remaining same as L1D

Finally improvement! ...by adding L2C prefetcher

Division of Labor

Name	Roll Number	
Ilindra Lakshmi Sai Shreya	190050050	L2C, Reasons(20%)
Palti Ramyasri	190050078	L2C, Reasons(20%)
Pokala Mohith	190050084	L1D, Reasons(20%)
Poluparthi Preetham	190050085	L1D, Results and Plots(20%)
Punna Hitesh Kumar	190050093	L1D, Results and Plots(20%)


Arigatōgozaimas