

Combining Local and Global History for High Performance Data Prefetching

Martin Dimitrov and Huiyang Zhou



School of Electrical Engineering and Computer Science
University of Central Florida



Our Contributions

- New localities in the local and global address stream
- A high performance prefetcher design
- Mechanisms for eliminating redundant prefetches
- Advocating for L1-cache data prefetchers



Presentation Outline

- Contributions
- Novel data localities in the address stream
- Proposed data prefetcher
- Filtering of redundant prefetches
- Design Space Exploration
- Experimental Results
- Conclusions



Novel Data Localities: Global Stride

- **Global Stride** exists when there is a constant stride between addresses of two *different* instructions.

global address stream →

Load A:	X	Y	Z
Load B:	X+d	Y+d	Z+d

- When does it occur
 - Load/store instructions access adjacent elements of a data structure
 - Address-Value Delta [MICRO-38] is also a form of global stride



Novel Data Localities: Most Common Stride

- **Most Common Stride** exists when a constant pattern is disrupted from time to time.

local address delta stream →

Store A: D X D Y D Z D ...

- When does it occur

```
for (j = lll = 0; j < ll; ++j){
  x = psv->value(j);
  if (isNotZero(x, eps)){
    k = psv->index(j);
    kk = u.row.start[k] + (u.row.len[k]++);
    u.col.idx[m++] = k;
    u.row.idx[kk] = i;
    u.row.val[kk] = x;
    ++lll;
  }
  ...
}
```

Code example from *Soplex*

68
47316
68
47212
68
47236
68
47068
68
47164
68
47132
68
47356
68

Local address delta in bytes



Novel Data Localities: Scalar Stride

- **Scalar Stride** exists when the address is multiplied or divided by a constant

local address stream →

Load A: 32D 16D 8D 4D 2D D ...

- When does it occur

```

long cmp;

while ( ... ){
    ...
    cmp *= 2;
    if( cmp + 1 <= net->max_residual_new_m )
        if( new[cmp-1].flow < new[cmp].flow )
            cmp++;
}

```

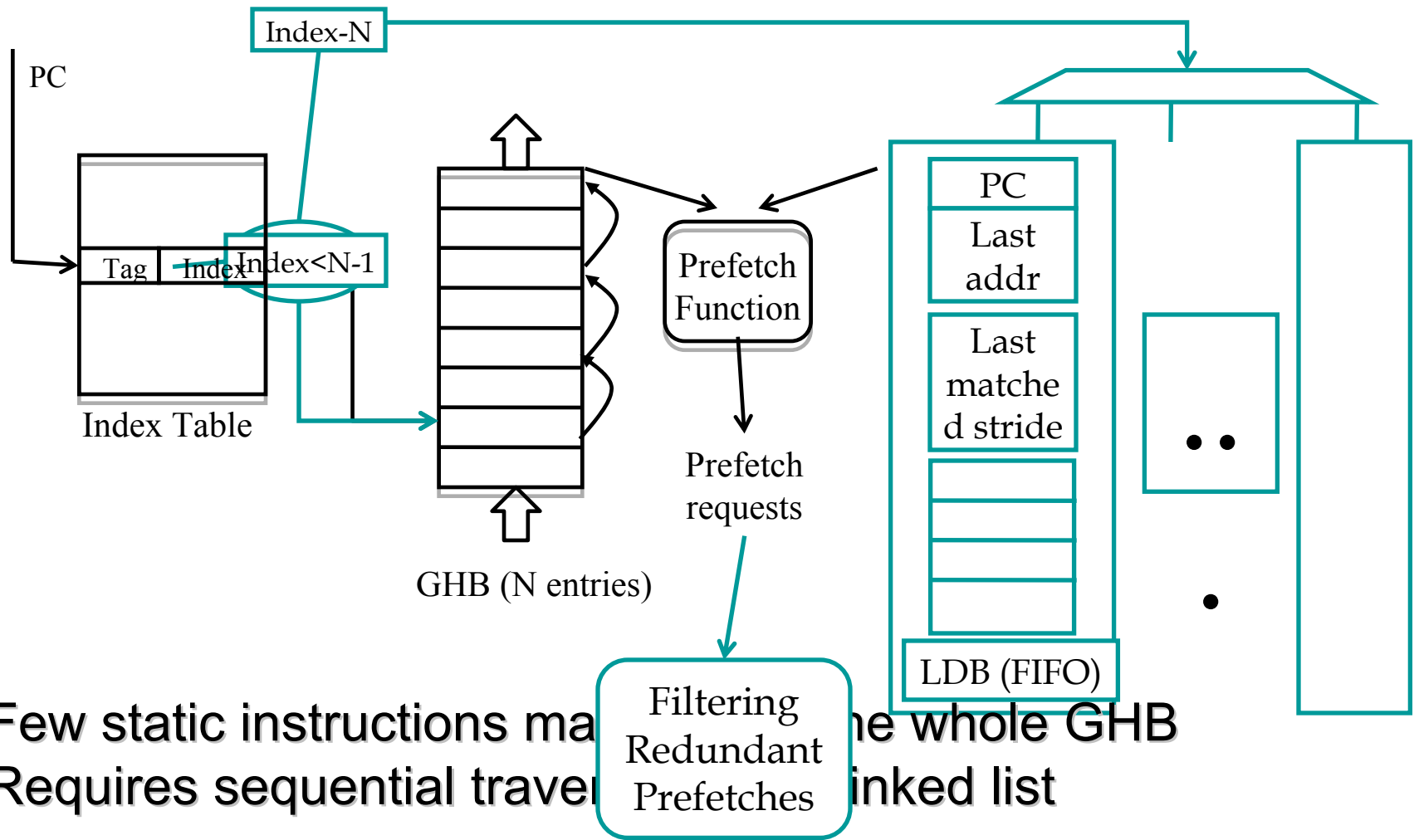
Code example from *mcf*

576
768
1600
3200
6336
12672
25344
50688
101440
202880
405696
811392
1622784
3245632
6491200
12982464
25964864
51929728
103859456
207718976
415437888

Local address delta in bytes



Global History Buffer (GHB) Prefetcher



- Few static instructions make up the whole GHB
- Requires sequential traversal of linked list

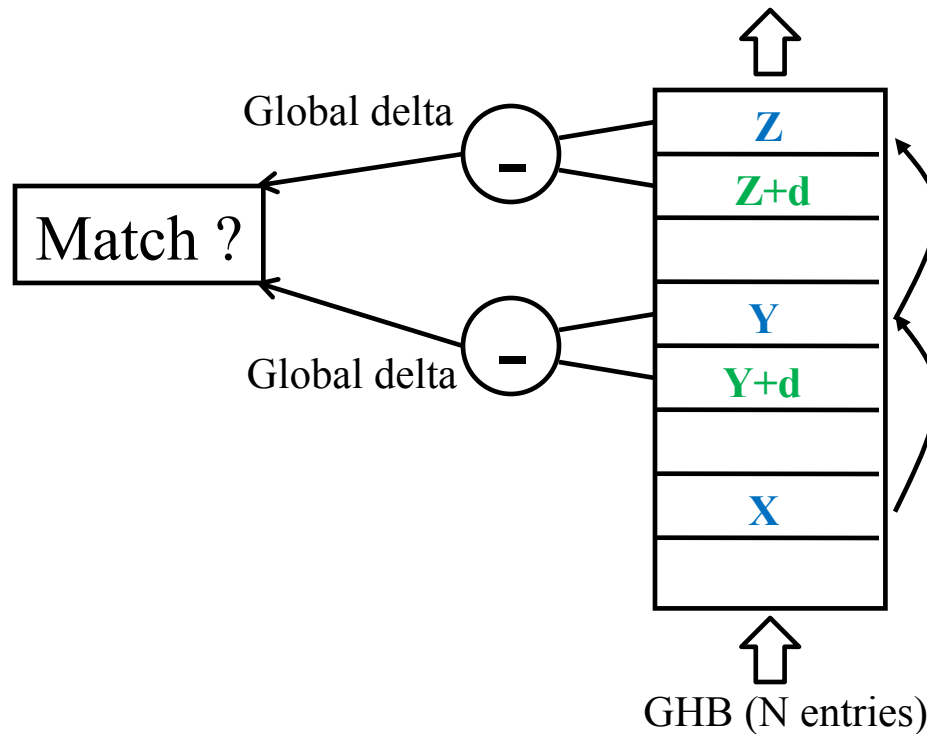


Prefetch Function Detecting Global Stride

global address stream \rightarrow

Load A: X Y Z

Load B: X+d Y+d Z+d





Prefetch Function

Detecting Delta Correlation

local delta stream →

Load A: a b

c	d
a	b

 a b c d a b c d ...

Match !

a b c d ← generate prefetches



Prefetch Function

Detecting Single Delta Match

local delta stream →

Load A: a

x
a

 c d a z c d a y c d ...

Match !

a x c d ← generate prefetches



Prefetch Function

- If no delta correlation is detected, generate 2 prefetches
 - Prefetch last matched stride to approximate most common stride.
 - Next line prefetch
- The output of the prefetch function is a buffer (up to max prefetch degree) filled with potential prefetch addresses.



Filtering of Redundant Prefetches

- Local redundant prefetches

Load A address stream

time 1: miss: **a** prefetch: **b, c, d, e**

time 2: hit (pref bit ON): **b** prefetch: ~~**c, d, e, f**~~

time 3: hit (pref bit ON): **c** prefetch: ~~**d, e, f, g**~~

- Global redundant prefetches

Load B prefetches: ~~**a+8**~~, x, y, etc.

Load C prefetches: ~~**b+16**~~, w, z, etc.

Other loads/stores use data in the same cache line as Load A.



Filtering of Redundant Prefetches

- Filtering local redundant prefetches
 - Add a confidence bit to each LDB to indicate that we have already prefetched the full prefetch degree
 - If conf bit is set, make only 1 prefetch

Load A address stream

time 1: miss: **a** prefetch: **b, c, d, e** conf: **ON**

time 2: hit (pref bit ON): **b** prefetch: **f** conf: **ON**

- Filtering global redundant prefetches
 - Use a MSHR
 - Use a Bloom filter. On a Bloom filter hit, drop the prefetch. Reset the Bloom filter periodically.



Design Space Exploration

Prefetch into the L1 or L2 Cache ?

- We advocate for prefetching into the L1 cache
 - + L1-cache hits are better than L2-cache hits
 - + More accurate address stream
 - + Access to the program counter (PC)
- Latency is more critical



Design Space Exploration

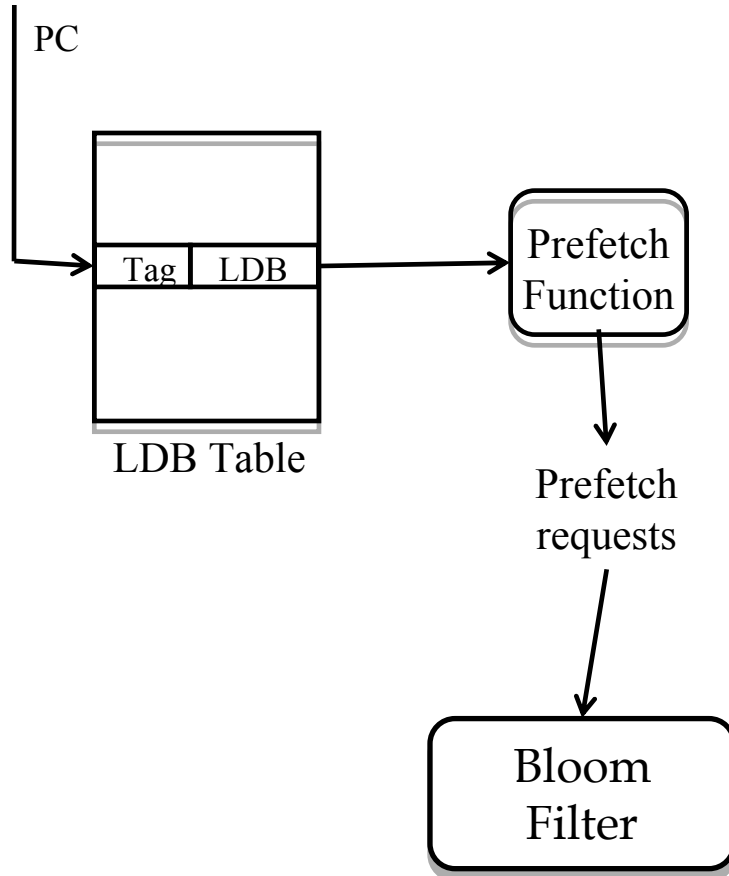
Three Prefetcher Design Points

- GHB-LDB-v1: Highest performance design, using MSHRs to remove redundant prefetches.
- GHB-LDB-v2: Scaled down design, using Bloom filter to remove redundant prefetches.
- LDB-only: Very complexity and latency efficient design.



Design Space Exploration

LDB-only Design



- Each entry in the table is an LDB. (a FIFO of last several deltas, last address and a confidence bit)
- Can detect all the stride patterns, except global stride
- Latency efficient: no linked list traversal, quick Bloom filter access



Storage Cost

Storage Cost	GHB-LDB-1	GHB-LDB-2	LDB-only
Index Table	256-entry 8-way 9728 bits	256-entry 8-way 9728 bits	64-entry 8-way
GHB	192 entry $192 * (32+8)$ = 7680 bits	128 entry $128 * (32+7)$ = 4992 bits	N/A
Prefetch Func.	1120 bits	1120 bits	1120 bits
Prefetch MSHR	256-entry 8-way $256*(21+3)=6144$ bits	N/A	N/A
Bloom filter	N/A	2048 + 8-bit reset counter	4096 + 9-bit reset counter
LDBs	16 LDBs $16*(7*32+32+32+32+5)$ =5200 bits	16 LDBs $16*(7*32+32+32+32+4+1)$ =5200 bits	64 LDBs $64*(7*24+32+32+3+1)$ =15104 bits
Counters	100 bits	100 bits	N/A
Total	29972 bits (3.7kB)	23196 bits (2.9kB)	20329 bits (2kB)



Experimental Results

Speedup for best performing design point GHB-LDB-v1

Speedup	bzip2	lbm	mcf	milc	omnetpp	soplex	xalan	Gmean
Conf1	1.07	2.89	2.65	1.97	1.13	1.54	0.99	1.61
Conf2	1.08	2.98	1.90	2.83	1.10	1.46	0.97	1.60
Conf3	1.02	2.98	1.88	2.83	1.11	1.48	1.37	1.67

Avg. speedup for other two designs: 1.60X and 1.56X



Conclusions

- We introduce a high performance prefetcher design for prefetching into the L1 cache.
- Discover and utilize novel localities in the global and local address streams
- Emphasize the importance of filtering redundant prefetches and proposing mechanisms to accomplish the task



Questions?

