Victim Cache and Skewed Associative Cache

31.10.2019

Group Members

Ayush Mangal (18114016)
Harshit Verma (18114029)
Hemant (18114030)
Hemil Panchiwala (18114031)
Kunal Mohan (18116042)
Nalin Prabhath (18116015)
Piyush Gupta (18116059)

Abstract

In our project, we have compared two new types of cache - the Skewed Associative Cache and Victim Cache.
Skewed Associative Cache has better behaviour than Set-associative caches. Typically a two-way skewed-associative cache has the hardware complexity same as that of two-way set-associative cache, yet simulations show that it exhibits approximately the same hit ratio as a four-way set-associative cache of the same size.
Victim Cache is basically a fully associative cache, comparatively smaller in size than L1 cache and placed behind the L1 cache. It holds the evicted block because of conflict miss from L1 cache. So rather than evicting the memory block, we store it in another cache, thus increasing the hit ratio and increasing the speed.

Importance

As hardware architecture and technology advanced, processor performance grew much faster than memory cycle times, leading to a large gap in performance. The problem of increasing memory latency, relative to processor speed, has been dealt with by adding high-speed cache memory.

Performance of associative caches depends on the associativity of the cache. There is a continuous trade-off between the access time and the hit ratio depending on the associativity. Skewed associative caches are better than set-associative cache because they have access time equivalent to that of a 2-way set-associative cache but the hit ratio is similar to that of a 4-way set-associative cache.

Direct Mapped caches have faster access time than set-associative caches. However, for a direct-mapped cache if multiple cache blocks in the memory map to the same cache-line they end up evicting each other when any one of them is accessed. This is known as the cache conflict problem. This problem is resolved by increasing the associativity of the cache. But there is a limit to which associativity can be increased owing to the complexity in its implementation. Thus, for solving the cache conflict problem for a cache with limited associativity, victim cache is employed.

Description

Victim Cache

A victim cache is a small, usually fully associative cache placed in the refill path to CPU cache, and it stores all the blocks evicted from that level of cache.

Victim cache is a hardware technique to improve performance of caches. It decreases conflict misses and improve hit latency. It is employed at the refill path of Level 1 cache, such that any cache line gets evicted from the cache is cached in the victim cache. Thus, victim cache gets populated only when data is pushed out of L1 cache.

In case of a miss in L1 cache, the victim cache is searched for the data. If the resulting access is a hit, the contents of L1 cache-line and the matching victim cache are swapped.

Implementation

The behavior of a victim cache in its respective interaction with the corresponding level cache is given below:

Cache Hit: No action

Cache Miss, Victim Hit: Block in victim cache and the one in L1 cache are replaced. This new entry in L1 cache becomes the most recently accessed block.

Cache Miss, Victim Miss: The block is brought from next level (L2 cache in our implementation). The block evicted from L1 cache is stored in victim cache.

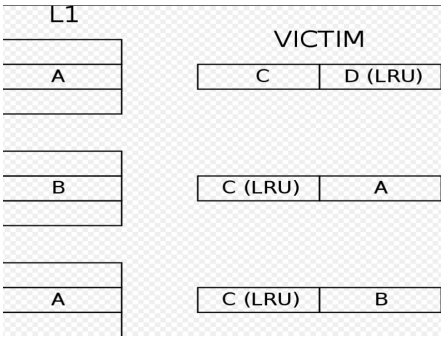
Example-

Suppose a direct-mapped L1 cache with blocks A, B pointing to the same set. It is linked to a 2 entry fully associative victim cache with blocks C, D in it.

The trace to be followed: A, B, A, B...

From the screenshot, we can see that, in the case of victim cache (VC) hit, blocks A and B are swapped. The least recently used block of VC remains as it is. Hence, it gives an illusion of associativity to the direct-mapped L1 cache, in turn reducing the conflict misses.

In the case of two caches, L1 and L2 with exclusive policy (L2 does not cache the same memory locations as L1), L2 acts as the victim cache for L1.



Selective Victim Cache (an improved version of victim cache)-

Another improvement in victim cache can be made by turning it into a selective victim cache. In the simple victim cache configuration, victim cache holds only those blocks evicted from the main cache as a result of replacements. In the selective victim cache configuration, some of the incoming blocks can be placed in addition to blocks resulting from replacements. When a new block is fetched to the first level cache, a prediction algorithm(not discussed in this project) is applied to determine whether the incoming block is to be placed in the main cache or the victim cache. Thus owing to this, the access time further reduces and hit rate increases, as expected.

Skewed Associative Cache

A Set Associative Cache and a Skewed Associative cache both have almost the same hardware complexity. The way in which Skewed and Set associative cache differ from each other is in the way of allocation of a block in their banks. The set-associative cache uses the same function (say f) to map a given memory address to a block in different banks as shown in figure1. The block chosen finally depends upon the replacement policy used. On the other hand, an N-way Skewed Associative cache also has N banks, but the block assignment in these banks is done by different functions (say f1 and f2) as shown in figure2.

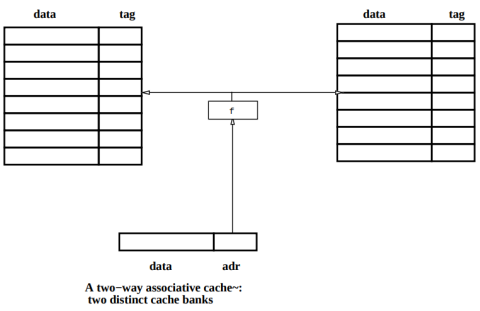


Figure 1.

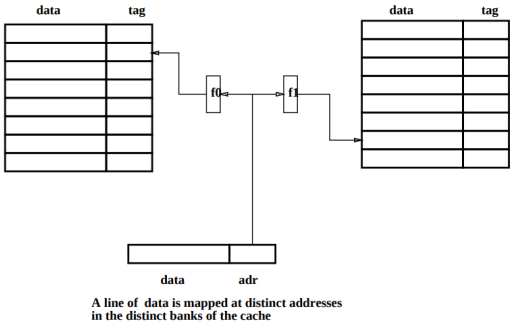


Figure 2.

Now although associativity reduces the conflict of misses, there still can be many of them even in the set-associative cache due to temporal and spatial locality. According to the principle of Skewed Associative cache, addresses in different sets should be mapped in a different way. With this the probability of conflict in all the sets at the same time reduces.

Description of Related Work

A. Introduced a new multi-bank cache organization: the skewed-associative cache A two-way skewed-associative cache has the same hardware complexity as a two-way set-associative cache, yet simulations show that it typically exhibits the same hit ratio as a four-way set-associative cache with the same size. Then skewed-associative caches must be preferred to set-associative caches.

B. In this paper, we show that the recently proposed four-way skewed associative cache yields very stable execution times and good average miss ratios on blocked algorithms. As a result, execution time is faster and much more predictable than with conventional caches. As a result of its better comportment, it is possible to use larger blocks sizes with blocked algorithms, which will furthermore reduce blocking overhead costs.

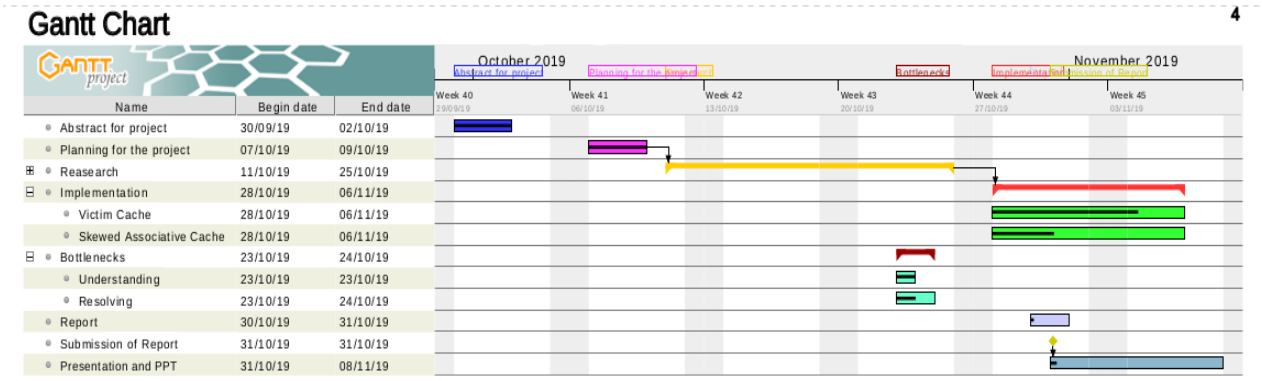
C. This review paper proposed the technique this technique is very useful to reduce the miss rate in cache memory. In the proposed technique the direct-mapped cache is added with a small associative cache known as a victim cache. A line removed from the direct-mapped is temporarily stored in the victim cache.

D. This paper gave an improvement of the victim cache scheme, called selective victim caching. In this scheme, incoming blocks into the first level cache are placed selectively in the main cache or a small victim cache by the use of a prediction scheme based on their past history of use. In addition, interchanges of blocks between the main cache and the victim cache are also performed selectively.

Description of the methods used to evaluate the proposed topic

For evaluating the caches which we had implemented, we randomly generated some addresses present in our main memory (which we had made using arrays and other data structures) and accessed the data corresponding to them. In doing so, the caches are used to store chunks of data (the data stored depends upon the cache type). We noted the hit rate, miss rate, number of accesses in main memory etc, and by using these parameters, we compared all the caches.

Gantt Chart



Results And Discussion

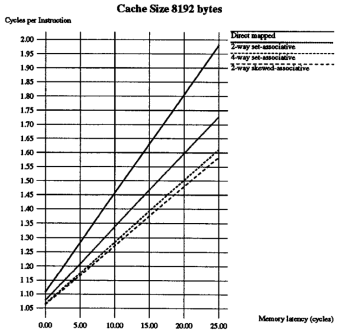
The two caches discussed above are two completely different ways of improving the performance of existing cache types. One type increases the physical memory space for improving the hit rate and reducing access time whereas the other cache implements an optimized method of mapping the data onto the existing physical cache memory to reduce the latency and increase the access speed of access. Therefore it would not be completely wise to compare two such different types of caches, as, for different types of input, the performance of either of the cache might be better than the other one. Nevertheless, here we have tried to compare them using a few input samples.

Victim cache output

```
Main memory size = 64 KB
L1 cache size = 1 KB
Victim cache size = 512 Bytes
Block size = 8 Bytes

Number of write miss in L1 cache = 62
Number of write miss in Victim cache = 53
Number of write hit in L1 cache = 115
Number of write hit in Victim cache = 94
Number of read miss in L1 cache = 47
Number of read miss in Victim cache = 53
Number of read hit in L1 cache = 112
Number of read hit in Victim cache = 95

Miss Ratio = 0.2699
Hit Ratio = 0.7301
```

Skewed Associative cache output

```
#instructions : 4  || #data : 12
    nbinst: number of instructions
    dataMiss: number of miss of the data cache
    instMiss:number of miss of the instruction cache


# 1024
strategy    NbBank  nbinst  dataMiss    instMiss    100*nbMiss/nbinst

SkewLRU     2   4   2       2       100.000000


# 2048
strategy    NbBank  nbinst  dataMiss    instMiss    100*nbMiss/nbinst

SkewLRU     2   4   2       2       100.000000


# 4096
strategy    NbBank  nbinst  dataMiss    instMiss    100*nbMiss/nbinst

SkewLRU     2   4   2       2       100.000000
```



Skewed cache performance graph

References

A. Andre Seznec IRISA-INRIA, Campus de Beaulieu 35042 Rennes Cedex, FRANCE "A case for two-way skewed-associative caches" .

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.30.1042& rep=rep1& type=pdf

B. "Skewed associativity enhances performance predictability"

https://hal.inria.fr/file/index/docid/74177/filename/RR-2499.pdf

C. "High Performance Cache Architecture Using Victim Cache"

https://www.ijedr.org/papers/IJEDR1703068.pdf

D. "Selective Victim Caching: A method to improve the performance of Direct-Mapped caches"

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=& arnumber=589235