Project Report

ARDHA RAGHAVENDRA REDDY(18114011)
GOVIND JAIN(18114025)
LAKSHAY MADAAN(18116044)
MADHUKAR(18116047)
PRANAV SINGHAL(18116062)
RONIT CHAKRABORTY(18112069)
SATYA VAMSI GADIKOYILA(18114024)

October 2019

1 Description

Here we are going to discuss a new way of Cache Management. This new way is SKEWED ASSOCIATIVE CACHE.

Previous studies have shown that, for cache sizes larger than 64 Kb, direct-mapped caches exhibit hit ratios nearly as good as set-associative caches at a lower hardware cost. Moreover, the cache hit time on a direct-mapped cache may be quite smaller than the cache hit time on a set-associative cache because the optimistic use of data flowing out from the cache is quite natural. But now, microprocessors are designed with small on-chip caches. The performance of low-end microprocessor systems highly depends on cache behavior. Simulations show that using some associativity in on-chip caches allows boosting the performance of these low-end systems. When considering the optimistic use of data (or instruction) flowing out from the cache, the cache hit the time of a two-way skewed-associative (or set-associative) cache is very close to the cache hit a time of a direct-mapped cache. Therefore two-way skewed associative caches represent the best trade-off for today microprocessors with on-chip caches whose sizes are in the range of 4-8K bytes. The motive is to implement and simulate a k way skewed set-associative cache using a MIPS simulator and studying the effects of changing associativity on hit ratio.

2 Importance of statement

A 2-way skewed associative cache has the same hardware complexity as 2 way set associative cache, yet it typically exhibits the same hit ratio as 4 way set associative cache. So skewed associative cache is preferred over the set-associative cache.

Programming languages and Test bench have a heavy influence on hit ratio. Different type of parameters is used in different contexts.

So,Hit ratio is a very important factor for the software industry by increasing the throughput with the same price is desirable.

3 Related works and References

Performance improvement and reduced conflict misses have been addressed the majority in it by limiting the size of working set of applications and blocking, which enhances locality in applications. Blocking due to a high level of reuse copies data much to control data and conflict placements in the cache. This is avoided using a complex hashing function for set selection for obtaining better behavior. Method still lacks for high associativity and complex hardwares.

References

1. A case for two way skewed associative cache. – Andre Seznec
   `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.30.1042&rep=rep1&type=pdf`

2. Skewed-Associative Caches – Prof. Sohi
   `https://pdfs.semanticscholar.org/ae67/dad82181b47dd2680ca07fe994a3912f6c35.pdf`

3. Skewed-Associative Caches – Andre Seznec, Francois Bodin
   `https://www.researchgate.net/publication/220758754_Skewed-associative_Caches`

4. `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=589219`

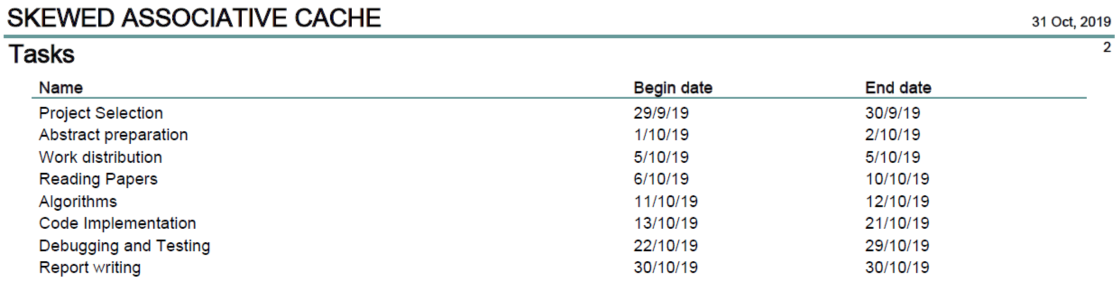
5. `https://hal.inria.fr/inria-00073481/document`

4 Methods used

We included a library (cLibrary.h) and we are taking some inputs regarding cache such as cache size, line size, input file, different mapping functions, test benches and simulations to measure hit ratio in different environments and conditions. We found that the hit-ratio is better in skew associative cache. Mainly, our testing was in C language. We used Github as our version control system so that all of us can collaborate our work together. Also we used GANTT charts to maintain our timeline and to keep a check on the work of members.

5 GANTT chart with the proposed timeline

(a) Proposed timeline of project



SKEWED ASSOCIATIVE CACHE

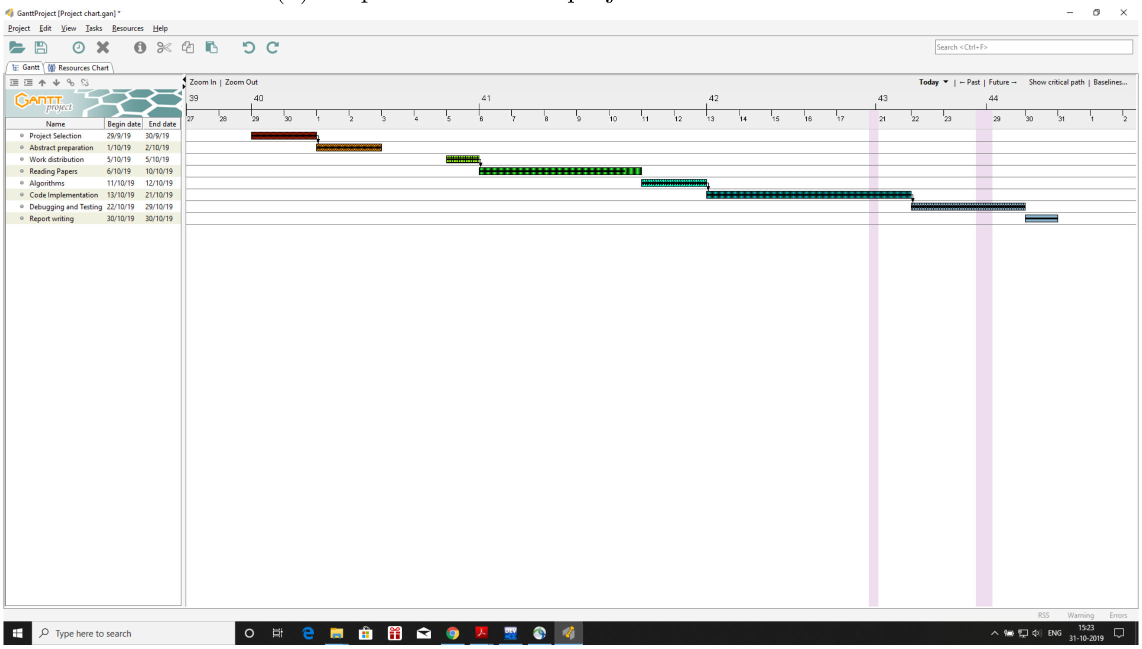
31 Oct, 2019

Tasks

2

Name	Begin date	End date
Project Selection	29/9/19	30/9/19
Abstract preparation	1/10/19	2/10/19
Work distribution	5/10/19	5/10/19
Reading Papers	6/10/19	10/10/19
Algorithms	11/10/19	12/10/19
Code Implementation	13/10/19	21/10/19
Debugging and Testing	22/10/19	29/10/19
Report writing	30/10/19	30/10/19

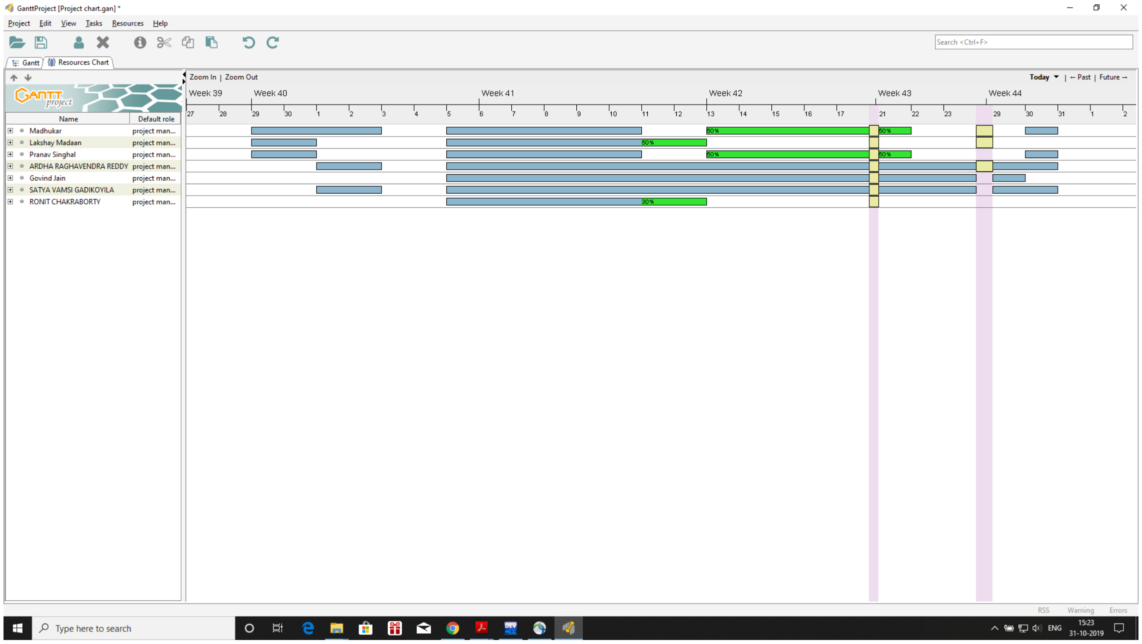
(b) Proposed timeline of project



GanttProject [Project chart.gan] *

Name	Begin date	End date
Project Selection	29/9/19	30/9/19
Abstract preparation	1/10/19	2/10/19
Work distribution	5/10/19	5/10/19
Reading Papers	6/10/19	10/10/19
Algorithms	11/10/19	12/10/19
Code Implementation	13/10/19	21/10/19
Debugging and Testing	22/10/19	29/10/19
Report writing	30/10/19	30/10/19

(c) The work done

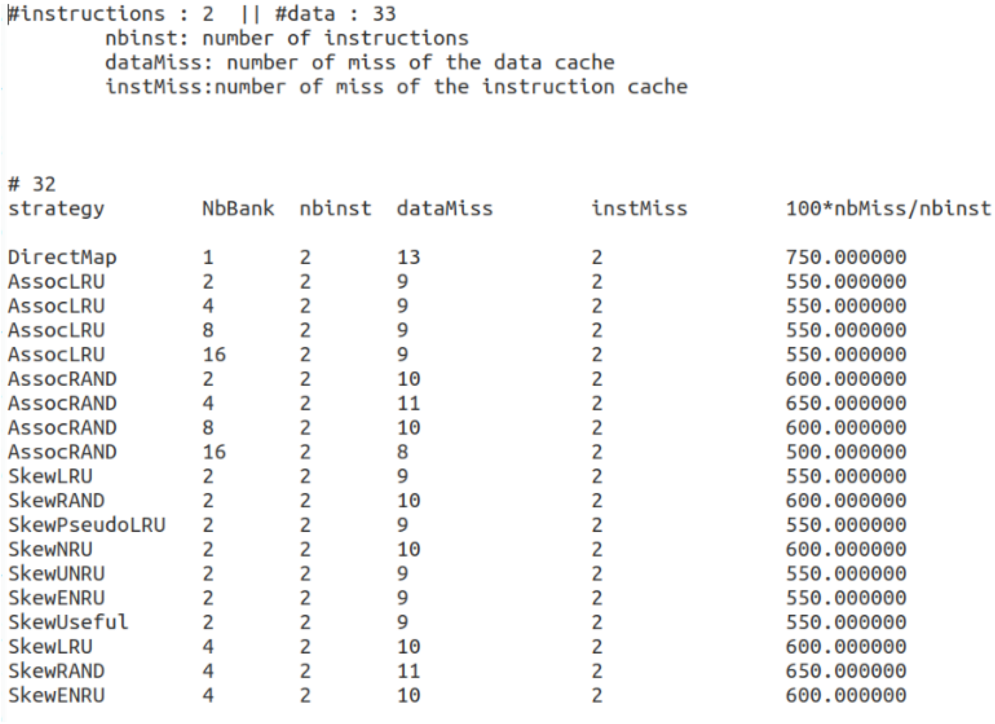


Name	Default role
Madhukar	project man...
Lakshay Madaan	project man...
Pranav Singhal	project man...
ARDHA RAGHAVENDRA REDDY	project man...
Govind Jain	project man...
SATYA VAMSI GADIKOYILA	project man...
RONIT CHAKRABORTY	project man...

6 Results and Discussion

- Cache size heavily influences the hit ratio. Larger the size lower the miss ratio.
- The skewed associative cache is not good or even worse for some test benches over normal associative.
- Even Hit ratio is higher but there is a comparatively large number of calling for and returning of small mapping functions. This increases latency.

This is the result of a particular test case



```
#instructions : 2  || #data : 33
        nbinst: number of instructions
        dataMiss: number of miss of the data cache
        instMiss:number of miss of the instruction cache



# 32
strategy        NbBank  nbinst  dataMiss        instMiss        100*nbMiss/nbinst

DirectMap       1       2       13              2               750.000000
AssocLRU        2       2       9               2               550.000000
AssocLRU        4       2       9               2               550.000000
AssocLRU        8       2       9               2               550.000000
AssocLRU        16      2       9               2               550.000000
AssocRAND       2       2       10              2               600.000000
AssocRAND       4       2       11              2               650.000000
AssocRAND       8       2       10              2               600.000000
AssocRAND       16      2       8               2               500.000000
SkewLRU         2       2       9               2               550.000000
SkewRAND        2       2       10              2               600.000000
SkewPseudoLRU   2       2       9               2               550.000000
SkewNRU         2       2       10              2               600.000000
SkewUNRU        2       2       9               2               550.000000
SkewENRU        2       2       9               2               550.000000
SkewUseful      2       2       9               2               550.000000
SkewLRU         4       2       10              2               600.000000
SkewRAND        4       2       11              2               650.000000
SkewENRU        4       2       10              2               600.000000
```