COMPUTER ARCHITECTURE
PROJECT 2 REPORT

By
Himanshu Garg : 50292195
Shreyas Narasimha: 50289736

PROBLEM STATEMENT

To run different benchmarks on Gem5 and analyse the variations of different parameters like cache size and associativity on values like hit rates and miss rates etc. Plot results and show them in a tabular format.

IMPLEMENTATION

The benchmarks were run on the metallica server where we have the Gem5 installed. Following benchmarks have been tested:

1. 401.bzip2
2. 429.mcf
3. 456.hmmer
4. 458.sjeng
5. 470.lbm

And the following parameters varied for each of the above benchmarks.

1. L1 D Cache size
2. L1 I Cache size
3. L2 Cache size
4. L1 D Associativity
5. L1 I Associativity
6. L2 Associativity
7. Block size

Executing Instructions

1. First an output directory was created in the home folder “ca_lab2_out” which contains two subdirectories:

a. Instance : This directory is to hold the stats file of the current instruction running
b. Final : This directory is to hold the final stats file of all the instructions.

2. Then we go to the gem5 directory at the location "/util/gem5/ and for each benchmark we find its argument files and how to run it by looking at run.sh script for each of them.

```
dict = {
  "401.bzip2" : ["input.program"],
  "429.mcf" : ["inp.in", "mcf.out"],
  "456.hmmer" : ["bombesin.hmm", "bombesin.hmm.new"],
  "458.sjeng" : ["test.txt"],
  "470.lbm" : ["100_100_130_cf_a.of","lbm.in"]
}
```

3. Also, we create the mappings of the values of each parameter we need to change.

```
l1d_sizes = ["8kB", "16kB" ,"64kB", "128kB", "256kB"]
l1i_sizes = ["4kB", "8kB", "16kB", "64kB", "128kB"]
l2_sizes = ["256kB", "512kB", "1MB", "2MB", "4MB"]
l1d_assoc = ["1","2","4","8","16"]
l1i_assoc = ["1","2","4","8","16"]
l2_assoc = ["1","2","4","8","16"]
block_size = ["8","16","32","64","128"]
benchmarks = ["401.bzip2", "429.mcf", "456.hmmer", "458.sjeng", "470.lbm"]
```

4. The command run for each instruction follows the following syntax:

/util/gem5/build/X86/gem5.opt	
-d /home/csgrad/hgarg/ca_lab2_out/instance/	#output directory
/util/gem5/configs/example/se.py	#system script
-c /util/gem5/benchmark/401.bzip2/src/benchmark	#benchmark file to run
-o /util/gem5/benchmark/401.bzip2/data/input.program	#arguments to the benchmark
-I 100000000	#max instructions to run
--l1d_size=128kB	#L1-D Cache size
--l1i_size=64kB	#L1-I Cache size
--l2_size=1MB	#L2 Cache size
--caches --l2cache	#Initialise caches
--l1d_assoc=16	#L1-D Associativity
--l1i_assoc=2	#L1-I Associativity
--l2_assoc=1	#L2 Associativity
--cacheline_size=64	#L1-D Cache Block size

The instruction count is taken as 100000000 to reduce the time of execution.

5. We first set the environment variables PATH and LD_LIBRARY_PATH using the following two commands:
    a. setenv PATH ${PATH}:"/util/gcc/bin"
    b. setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:"/util/gcc/lib64"

6. We then write a script with the mappings discussed above to automate the process of running each instruction using the main command given above.

7. To test initially that we are getting value of hit in the generated stats file we use the following command:
    a. cat ~/ca_lab2_out/stats.txt | grep "hits"

8. For each each benchmark the 7 parameters are varied by keeping the others constant as can be seen below:

```
base_l1d = "128kB"
base_l1i = "64kB"
base_l2 = "1MB"
base_l1dassoc = "2"
base_l1iassoc = "2"
base_l2assoc = "1"
base_block = "64"
```

8. The above main command is executed by passing the different parameter values. This takes the form like shown below when written in the python script:

```
benchmarks = ["401.bzip2", "429.mcf", "456.hmmer", "458.sjeng", "470.lbm"]
final_out_dir = "/home/csgrad/hgarg/ca_lab2_out/final/"
gem5_script_file = "/util/gem5/configs/example/se.py"
out_dir = "/home/csgrad/hgarg/ca_lab2_out/instance/"
ex_com = "/util/gem5/build/X86/gem5.opt"
gem5benchdir = "/util/gem5/benchmark/"
max_instr = "100000000"

subprocess.call(["time", ex_com,
"-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+l1d_assoc[i], "--l1i_assoc="+base_l1iassoc,
"--l2_assoc="+base_l2assoc, "--cacheline_size="+base_block])
```

9. The generated stats file for each instruction execution is renamed and moved to the final directory. The command to do that is as follows:

```
subprocess.call(["mv", out_dir+"stats.txt", final_file])
```

Here we rename the file according to the parameter being varied, its size and the benchmark for which it ran.

10. After each instruction for all the cases are run we obtain 175 stats files containing the information for each instruction like hit rate,miss rate etc. The following are the values that we are interested in.
    a. Hit rate of L1 D cache
    b. Miss rate of L1 D cache
    c. Hit rate of L1 I cache
    d. Miss rate of L1 I cache
    e. Hit rate of L2 cache
    f. Miss rate of L2 cache
    g. CPI

Note: The final script file: **runCALab2.py** has been provided in the scripts folder.

Instructions:
The **commands** run to execute the instructions are given at the **end** of the document. (script code)

Creating Graphs and Tables from the data

1. After we have run all the instructions using the script file (provided in the "scripts" folder) : runCALab2.py we write another script in python to process the data collected and find the values given above like hit rate for each case.

2. We created a list with with the mappings we used for running the instructions.

3. Then all the stats files were opened for a particular benchmark by varying parameter at a time like L1-D Cache size.

4. The results were then tabulated in python using the 'tabulate' library.

5. The values in the tables were written onto a .csv file and then converted into graphs using Microsoft Excel. Some graphs are also generated by the python file which tabulates results.

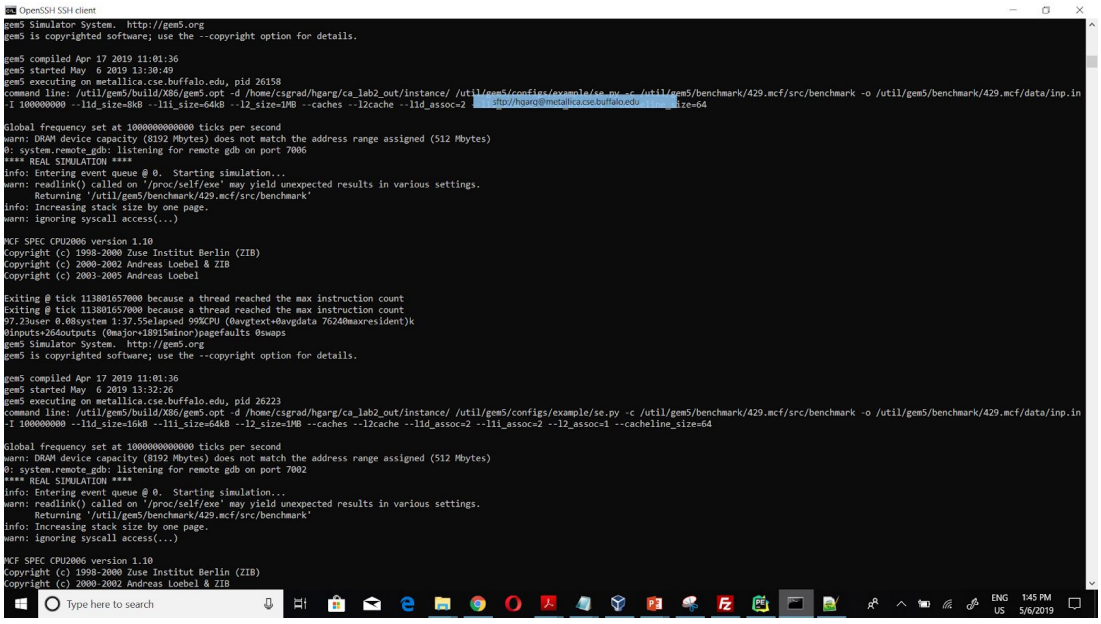
6. The CPI is calculated using the following formula:

$$CPI = 1 + \frac{(IL1.miss\_num + DL1.miss\_num) \times 6 + L2.miss\_num \times 50}{Total\_Inst\_num}$$

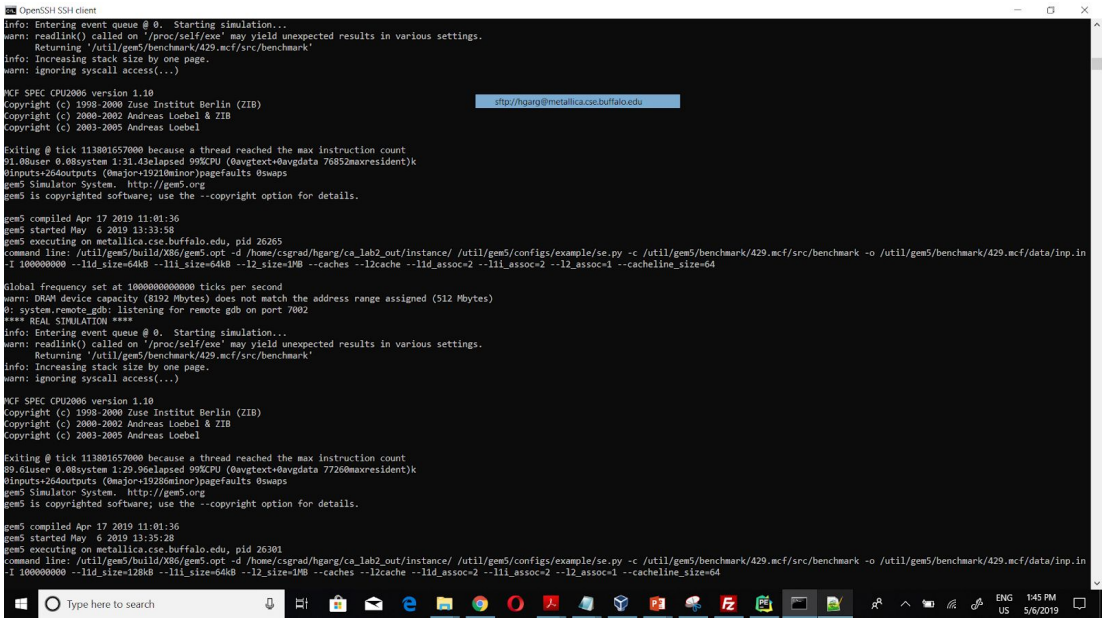
Instructions:
The commands and the code for this part is provided at the end of the document.

Screenshots

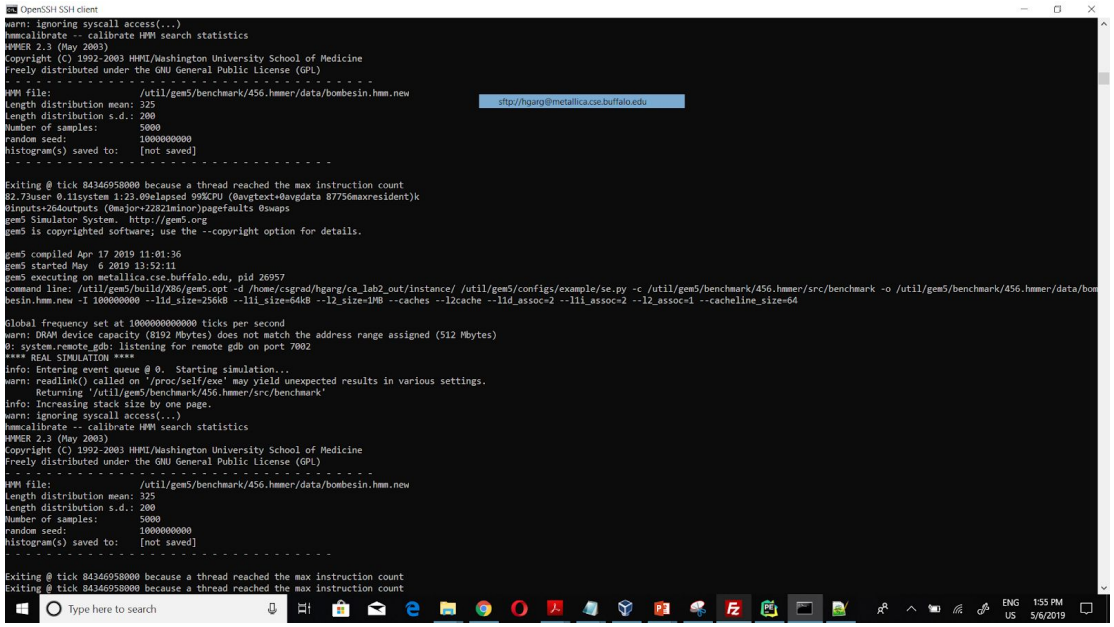
1. Running for 401.bzip2 benchmark



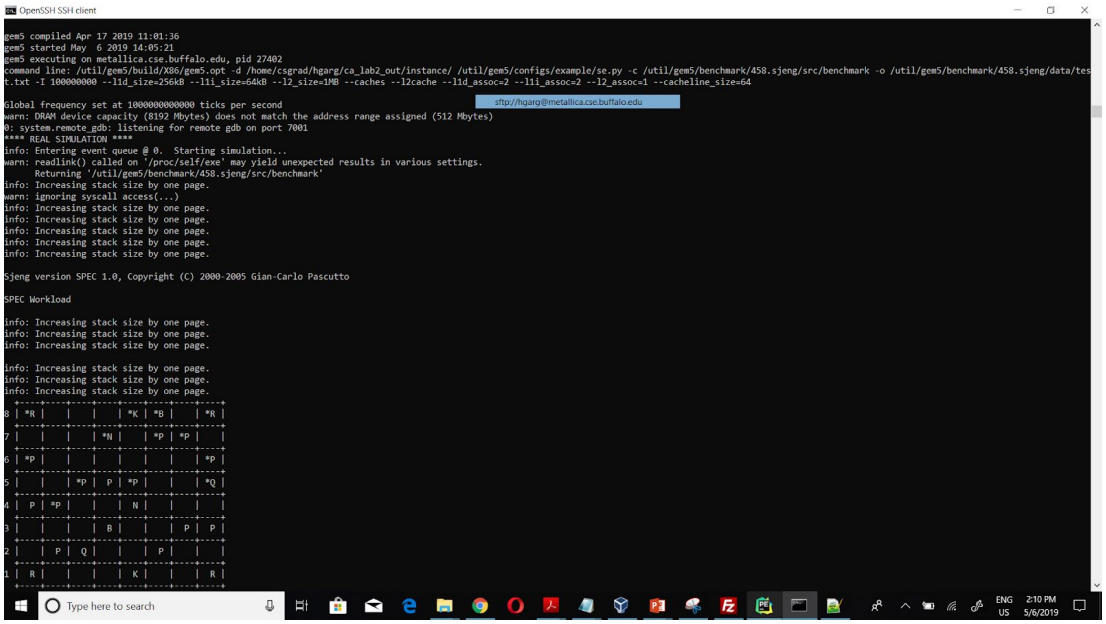
2. Running for 429.mcf benchmark



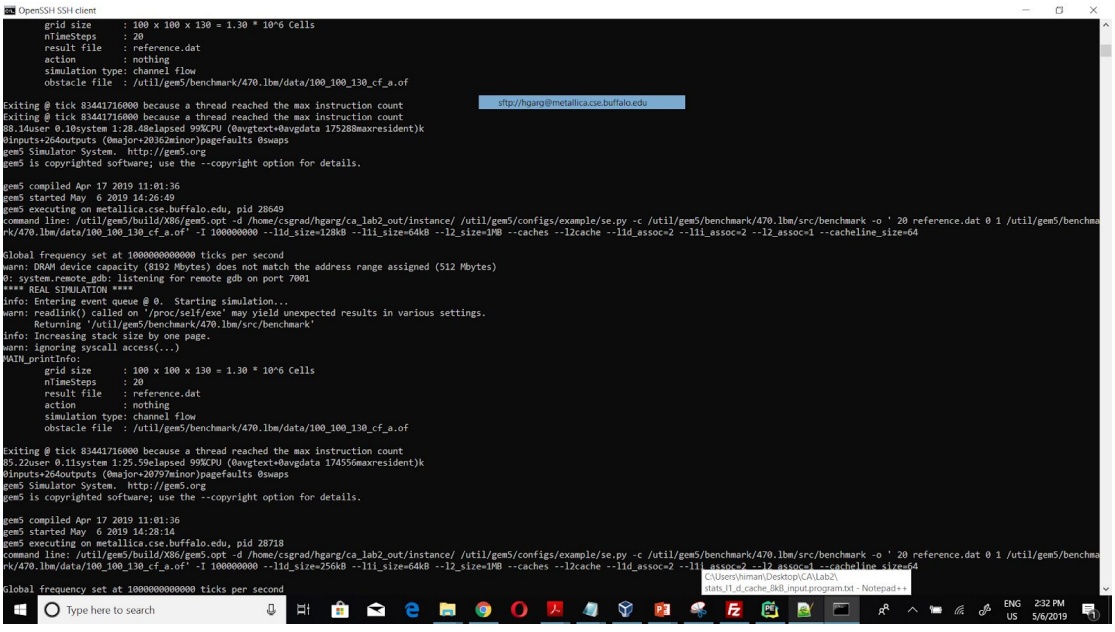
3. Running for 456.hmmer benchmark



4. Running for 458.sjeng benchmark



5. Running for 470.lbm benchmark



RESULTS

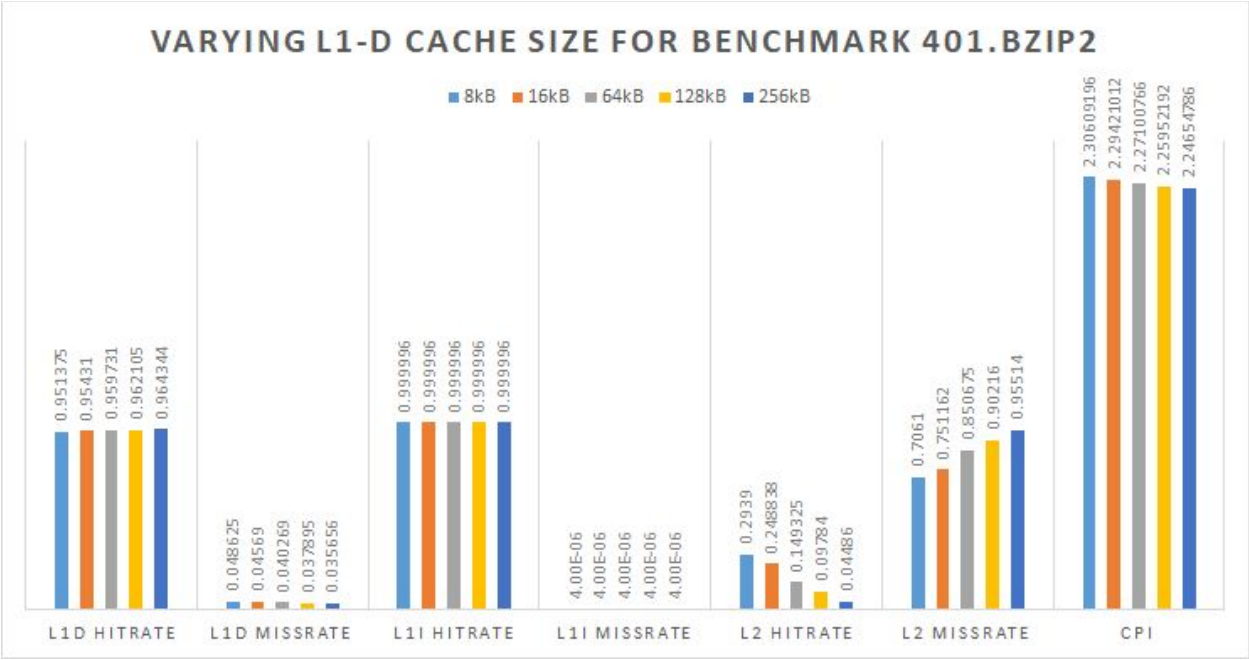
Graphs and Tables

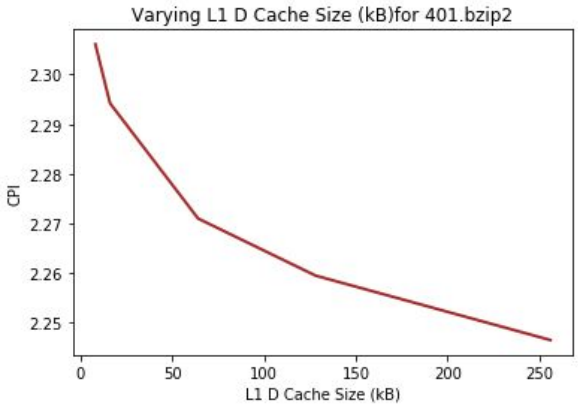
Benchmark : 401.bzip2

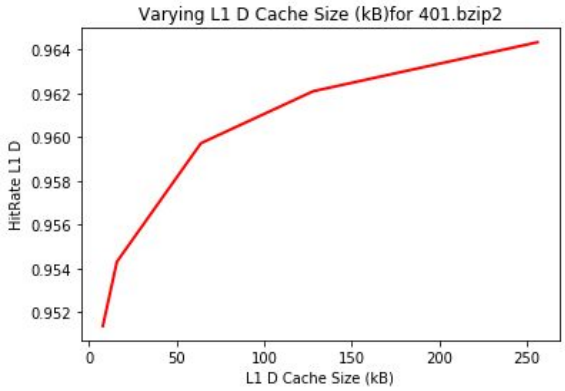
1. Varying L1-D cache size

Varying L1-D Cache Size for 401.bzip2

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8kB	0.951375	0.048625	0.999996	4e-06	0.2939	0.7061	2.30609
16kB	0.95431	0.04569	0.999996	4e-06	0.248838	0.751162	2.29421
64kB	0.959731	0.040269	0.999996	4e-06	0.149325	0.850675	2.27101
128kB	0.962105	0.037895	0.999996	4e-06	0.09784	0.90216	2.25952
256kB	0.964344	0.035656	0.999996	4e-06	0.04486	0.95514	2.24655







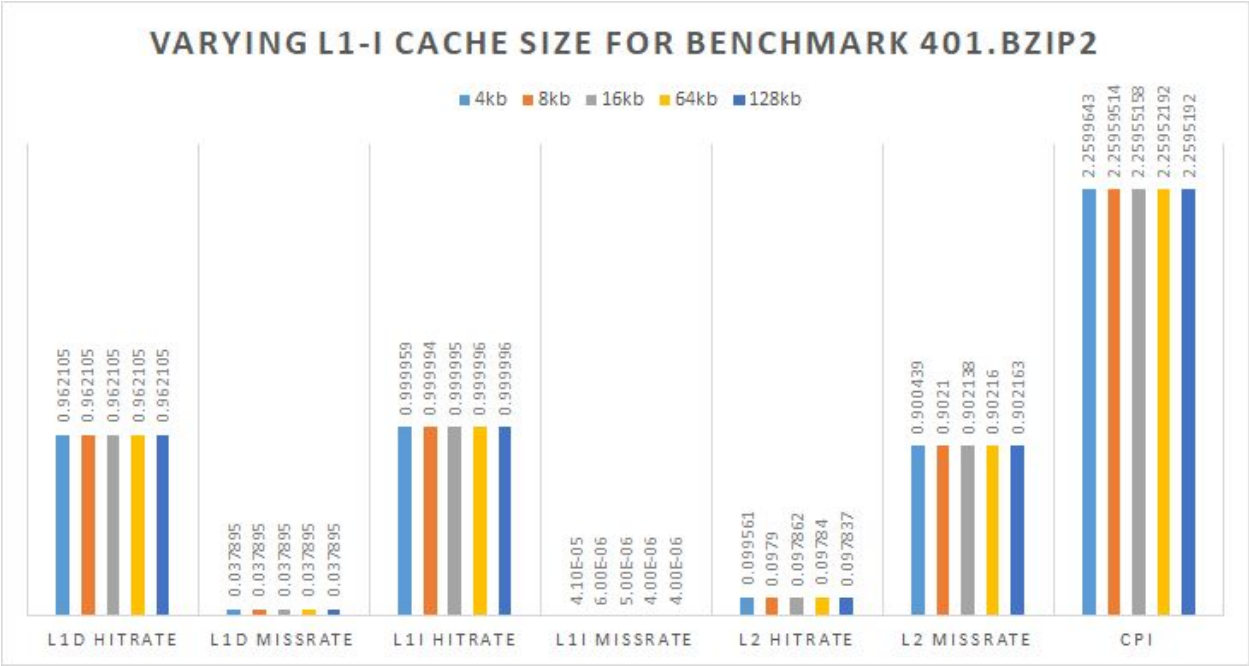
Explanation

- It can be seen here that as the L1-D cache size is increased there is a decrease in L2 hit-rate i.e an increase in the L2 cache miss-rate. This can be attributed to the fact that as L1 D becomes larger with L2 being kept constant, the requests that the L2 cache gets are references that probably have never been loaded onto the cache and are lower in number.
- The cycles per instruction(CPI) decreases due to the fact that there is less miss penalty due to a higher hit-rate.(It can be seen that the L1D hit-rate increases as the L1D cache size increases.)
- The difference in the L1 hit-rate was negligible as can be seen in the bar-plot.

2. Varying L1-I cache size

Varying L1-I Cache Size for 401.bzip2

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
4kb	0.962105	0.037895	0.999959	4.1e-05	0.099561	0.900439	2.25996
8kb	0.962105	0.037895	0.999994	6e-06	0.0979	0.9021	2.2596
16kb	0.962105	0.037895	0.999995	5e-06	0.097862	0.902138	2.25955
64kb	0.962105	0.037895	0.999996	4e-06	0.09784	0.90216	2.25952
128kb	0.962105	0.037895	0.999996	4e-06	0.097837	0.902163	2.25952



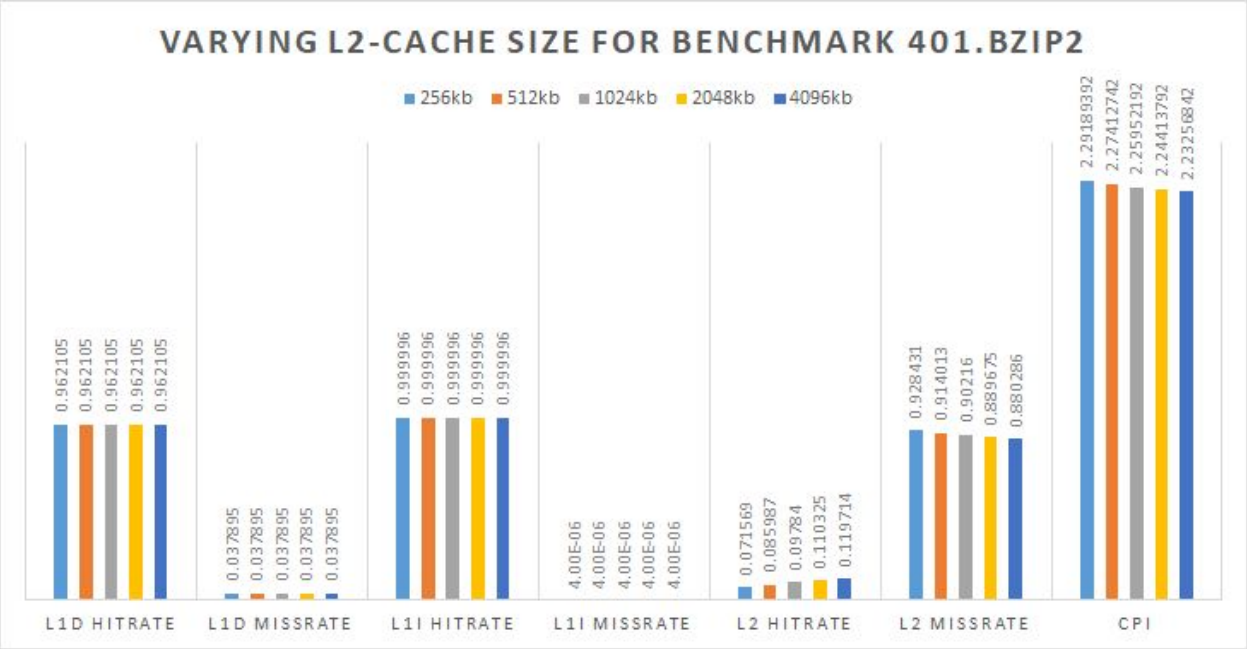
Explanation

- The bar-plot shows us that there almost to change in the parameters when L1-I is varied. This can be attributed to the fact the Instruction cache doesn't affect the hit and miss rate as much as the data cache.
- The trend of decreasing CPI is as cache size is reduced is followed by this test run.
- The L1 I hit-rate increases as the cache-size increases.

3. Varying L2 cache size

Varying L2 Cache Size for 401.bzip2

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
256kb	0.962105	0.037895	0.999996	4e-06	0.071569	0.928431	2.29189
512kb	0.962105	0.037895	0.999996	4e-06	0.085987	0.914013	2.27413
1024kb	0.962105	0.037895	0.999996	4e-06	0.09784	0.90216	2.25952
2048kb	0.962105	0.037895	0.999996	4e-06	0.110325	0.889675	2.24414
4096kb	0.962105	0.037895	0.999996	4e-06	0.119714	0.880286	2.23257



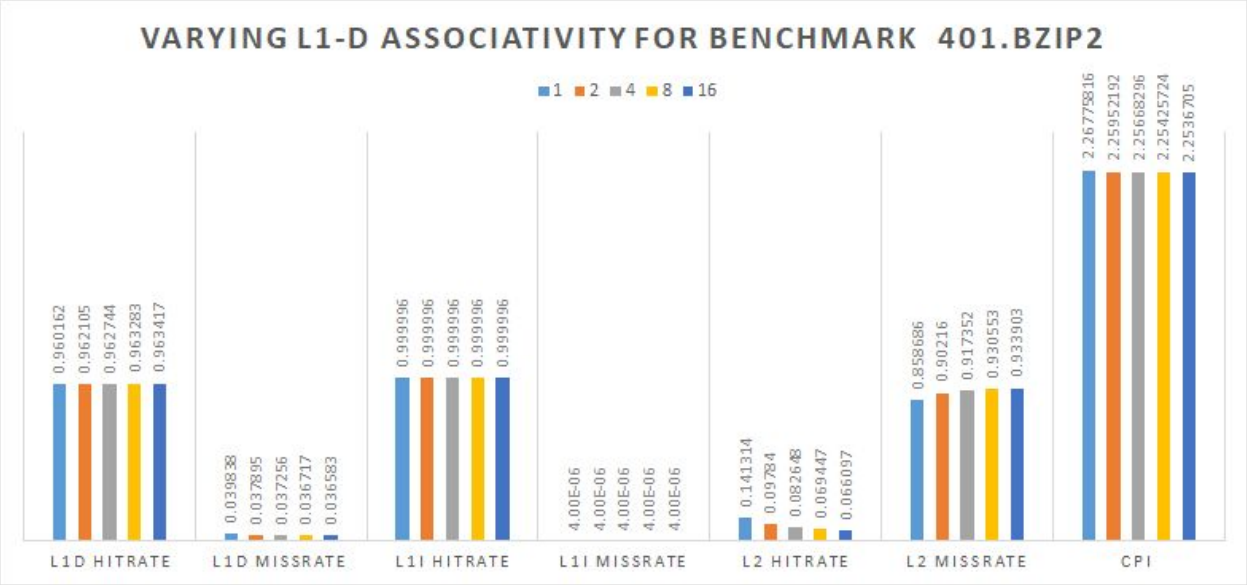
Explanation

- As L2-cache is varied, CPI decreases, and the hit-rate of the L2 cache increases.
- There is negligible change in the L1 rates.

4. Varying L1-D Associativity

Varying L1 D Associative Cache Size for 401.bzip2

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.960162	0.039838	0.999996	4e-06	0.141314	0.858686	2.26776
2	0.962105	0.037895	0.999996	4e-06	0.09784	0.90216	2.25952
4	0.962744	0.037256	0.999996	4e-06	0.082648	0.917352	2.25668
8	0.963283	0.036717	0.999996	4e-06	0.069447	0.930553	2.25426
16	0.963417	0.036583	0.999996	4e-06	0.066097	0.933903	2.25367



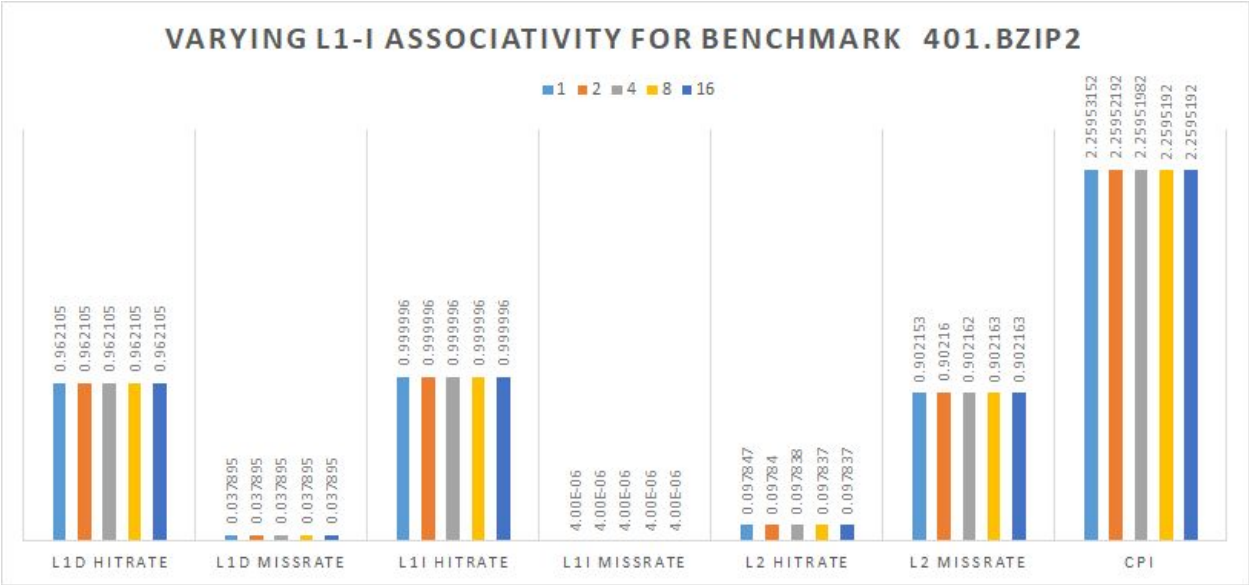
Explanation

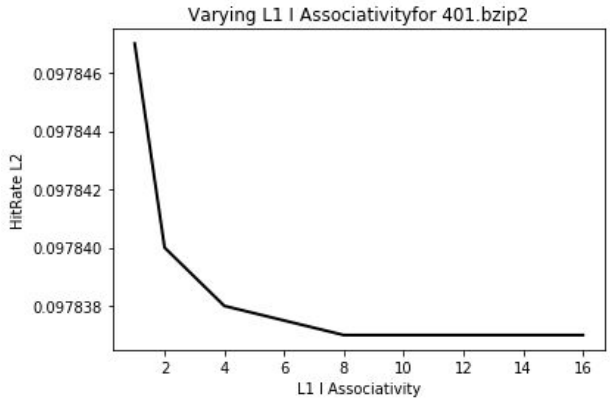
- As associativity increases L2 hit-rate is seen to improve with a slight improvement in CPI.

5. Varying L1-I Associativity

Varying L1 I Associative Cache Size for 401.bzip2

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.962105	0.037895	0.999996	4e-06	0.097847	0.902153	2.25953
2	0.962105	0.037895	0.999996	4e-06	0.09784	0.90216	2.25952
4	0.962105	0.037895	0.999996	4e-06	0.097838	0.902162	2.25952
8	0.962105	0.037895	0.999996	4e-06	0.097837	0.902163	2.25952
16	0.962105	0.037895	0.999996	4e-06	0.097837	0.902163	2.25952

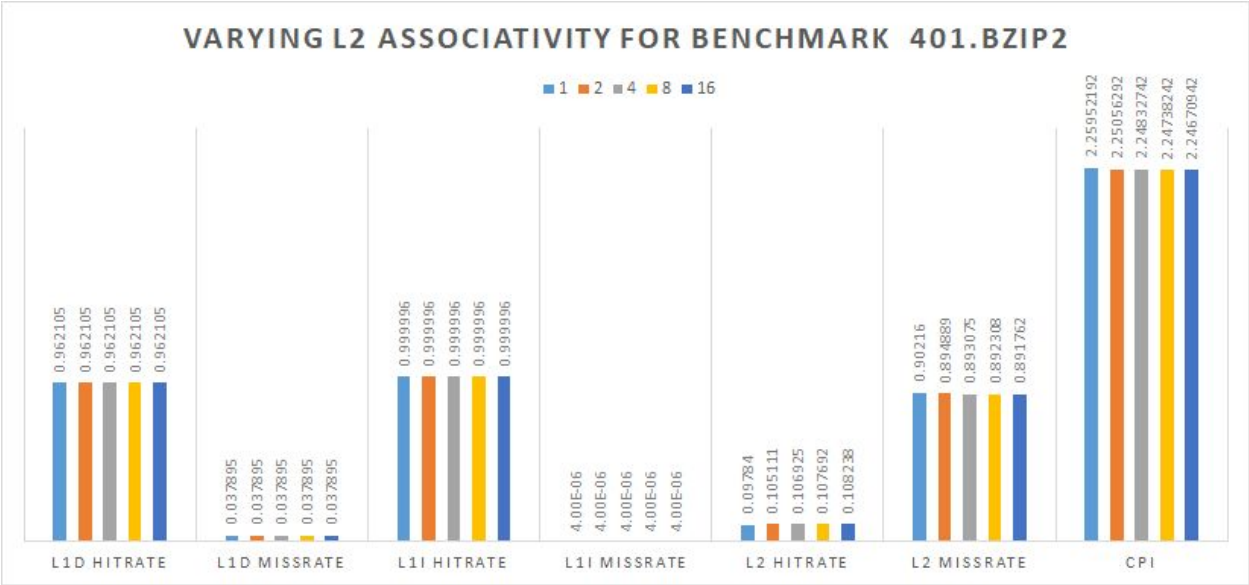


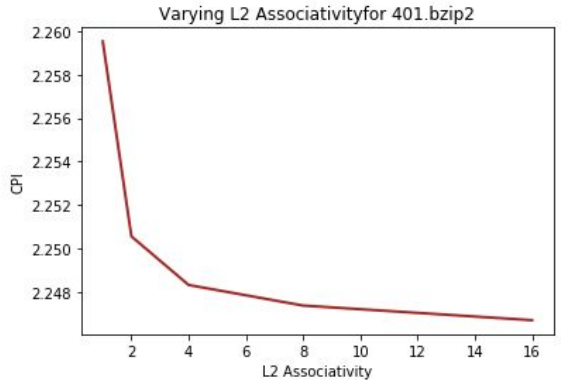


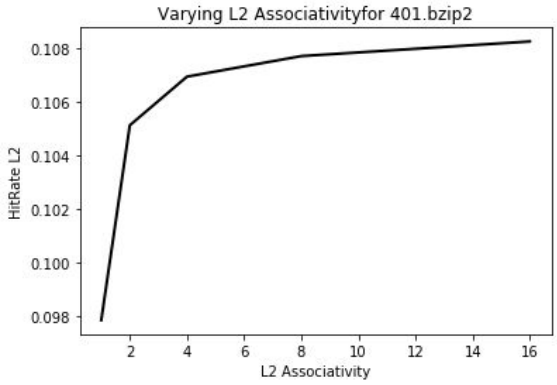
6. Varying L2 Associativity

Varying L2 Associative Cache Size for 401.bzip2

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.962105	0.037895	0.999996	4e-06	0.09784	0.90216	2.25952
2	0.962105	0.037895	0.999996	4e-06	0.105111	0.894889	2.25056
4	0.962105	0.037895	0.999996	4e-06	0.106925	0.893075	2.24833
8	0.962105	0.037895	0.999996	4e-06	0.107692	0.892308	2.24738
16	0.962105	0.037895	0.999996	4e-06	0.108238	0.891762	2.24671



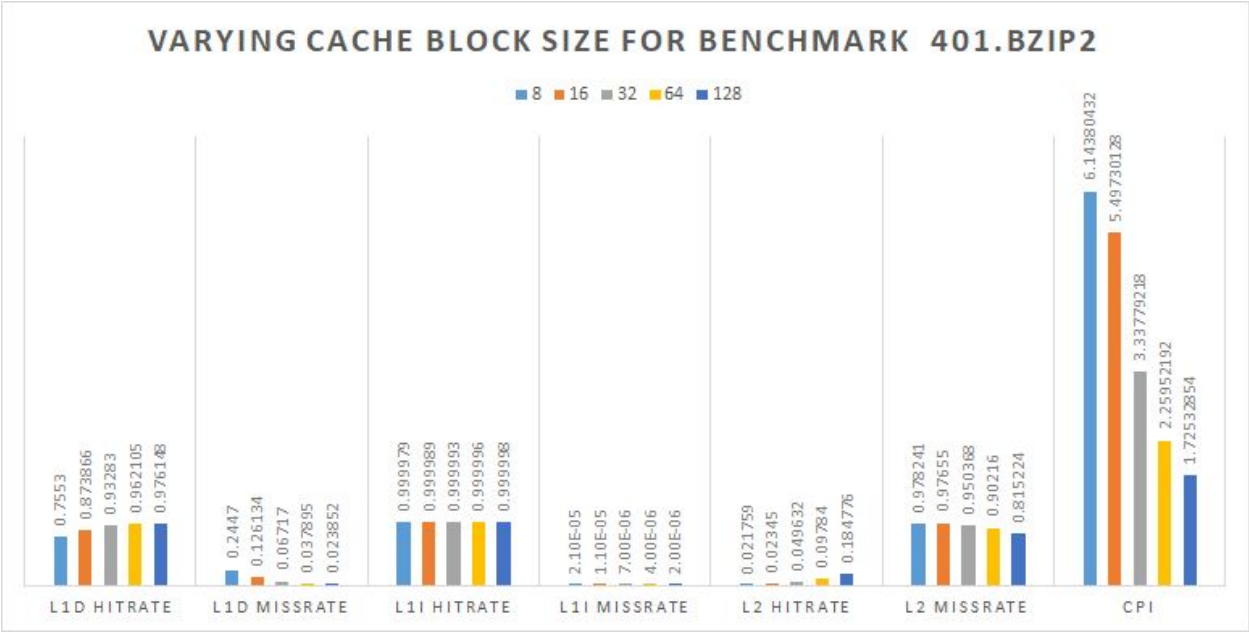


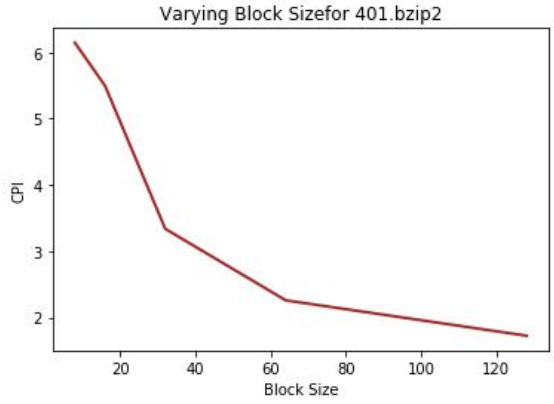


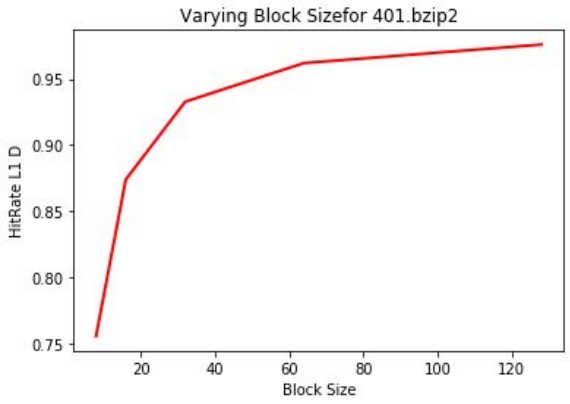
7. Varying Cache Block size

Varying Cache Block Size for 401.bzip2

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8	0.7553	0.2447	0.999979	2.1e-05	0.021759	0.978241	6.1438
16	0.873866	0.126134	0.999989	1.1e-05	0.02345	0.97655	5.4973
32	0.93283	0.06717	0.999993	7e-06	0.049632	0.950368	3.33779
64	0.962105	0.037895	0.999996	4e-06	0.09784	0.90216	2.25952
128	0.976148	0.023852	0.999998	2e-06	0.184776	0.815224	1.72533







Explanation

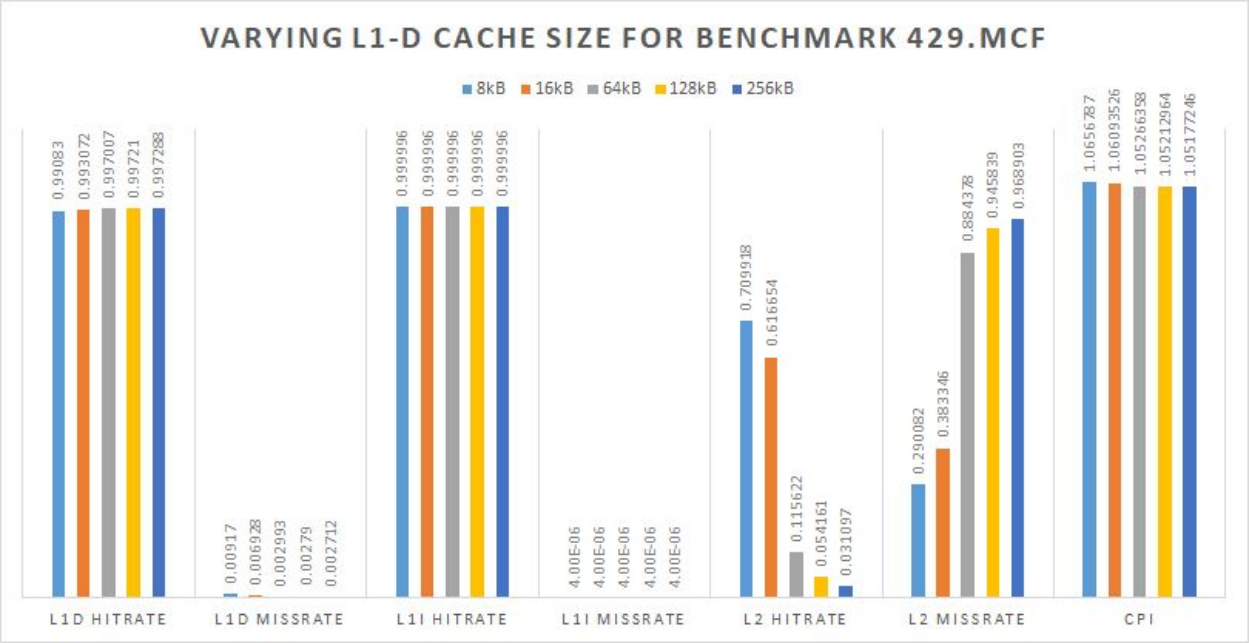
- Block size significantly affects performance.
- As block size increases, CPI reduces significantly due to the fact that more data can be read in one go.
- There is also an increase in the L1 D hit-rate which is expected due to the fact that increased block size improves locality of data.

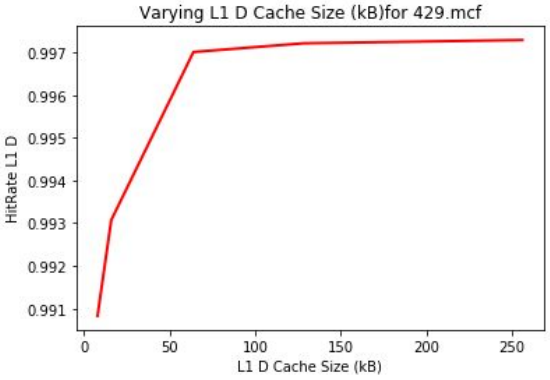
Benchmark : 429.mcf

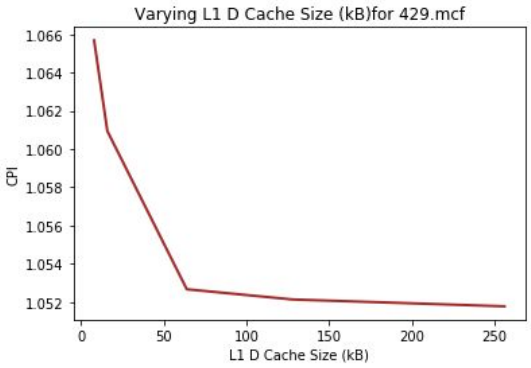
1. Varying L1-D cache size

Varying L1-D Cache Size for 429.mcf

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8kB	0.99083	0.00917	0.999996	4e-06	0.709918	0.290082	1.06568
16kB	0.993072	0.006928	0.999996	4e-06	0.616654	0.383346	1.06094
64kB	0.997007	0.002993	0.999996	4e-06	0.115622	0.884378	1.05266
128kB	0.99721	0.00279	0.999996	4e-06	0.054161	0.945839	1.05213
256kB	0.997288	0.002712	0.999996	4e-06	0.031097	0.968903	1.05177







Explanation

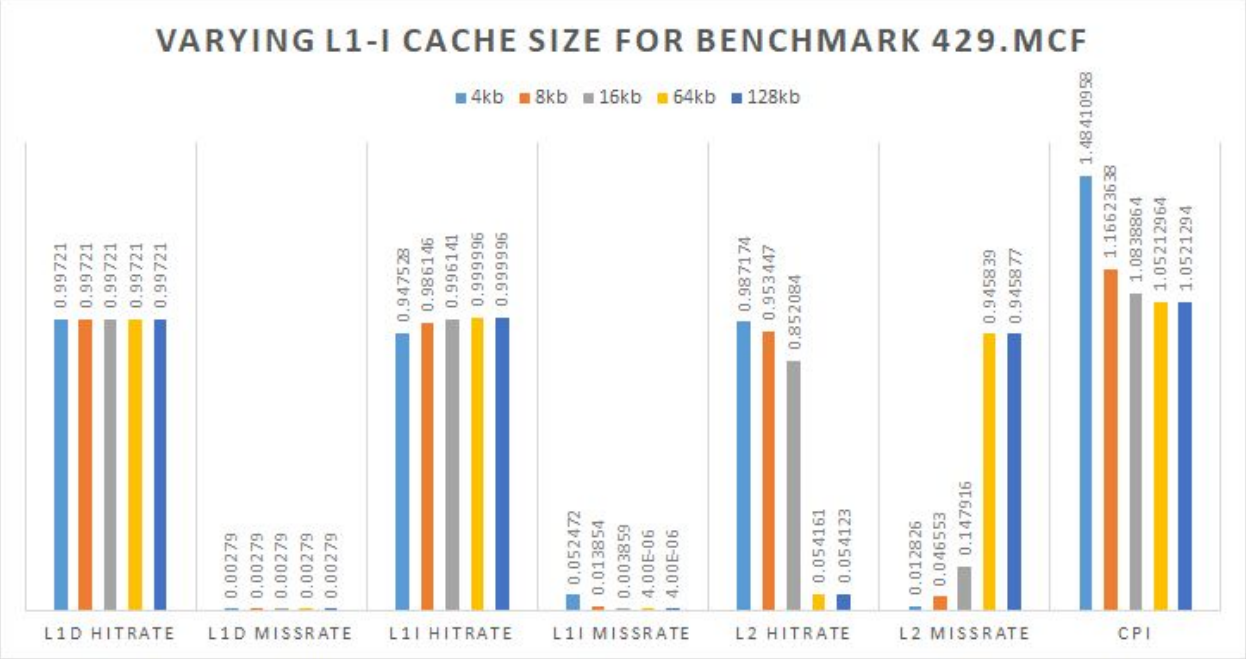
- The CPI can be seen to decrease as the L1D cache size increases since this reduces the miss-rate.

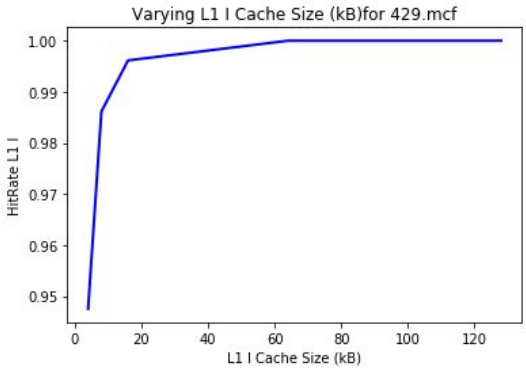
- The hit-rate increases with increase in L1-D cache size since more data can now be stored on the cache thereby reducing miss penalty.
- This benchmark shows a significant increase in L2 miss-rate as size of L1 D increases.
- There is also an increase in the hit-rate of L1D cache.

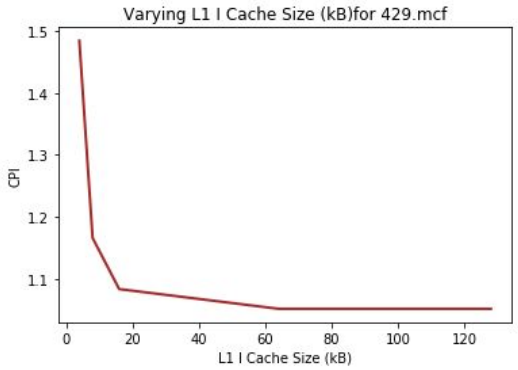
2. Varying L1-I cache size

Varying L1-I Cache Size for 429.mcf

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
4kb	0.99721	0.00279	0.947528	0.052472	0.987174	0.012826	1.48411
8kb	0.99721	0.00279	0.986146	0.013854	0.953447	0.046553	1.16624
16kb	0.99721	0.00279	0.996141	0.003859	0.852084	0.147916	1.08389
64kb	0.99721	0.00279	0.999996	4e-06	0.054161	0.945839	1.05213
128kb	0.99721	0.00279	0.999996	4e-06	0.054123	0.945877	1.05213







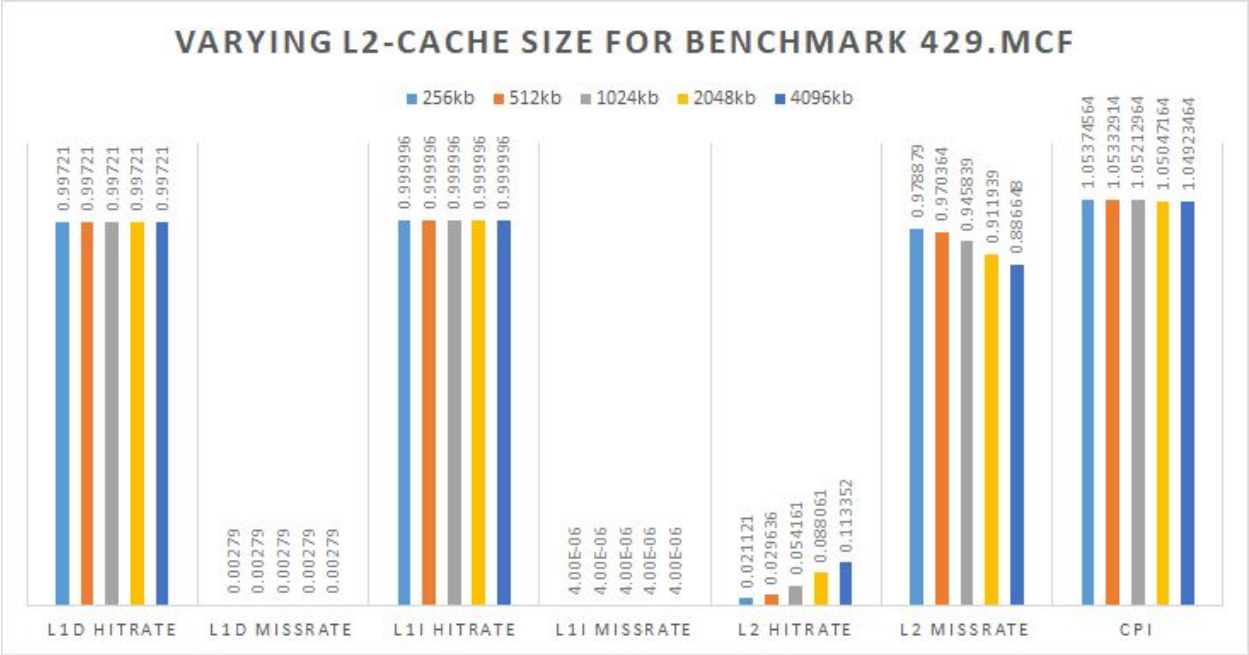
Explanation

- The L1-I miss-rate reduces here significantly as the cache size is doubled.
- The CPI is also seen to reduce slightly. This benchmark works quite well, in that, the CPI is very close to 1.
- The L2 miss-rate increases significantly as the L1 hit-rate increases.

3. Varying L2 cache size

Varying L2 Cache Size for 429.mcf

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
256kb	0.99721	0.00279	0.999996	4e-06	0.021121	0.978879	1.05375
512kb	0.99721	0.00279	0.999996	4e-06	0.029636	0.970364	1.05333
1024kb	0.99721	0.00279	0.999996	4e-06	0.054161	0.945839	1.05213
2048kb	0.99721	0.00279	0.999996	4e-06	0.088061	0.911939	1.05047
4096kb	0.99721	0.00279	0.999996	4e-06	0.113352	0.886648	1.04923



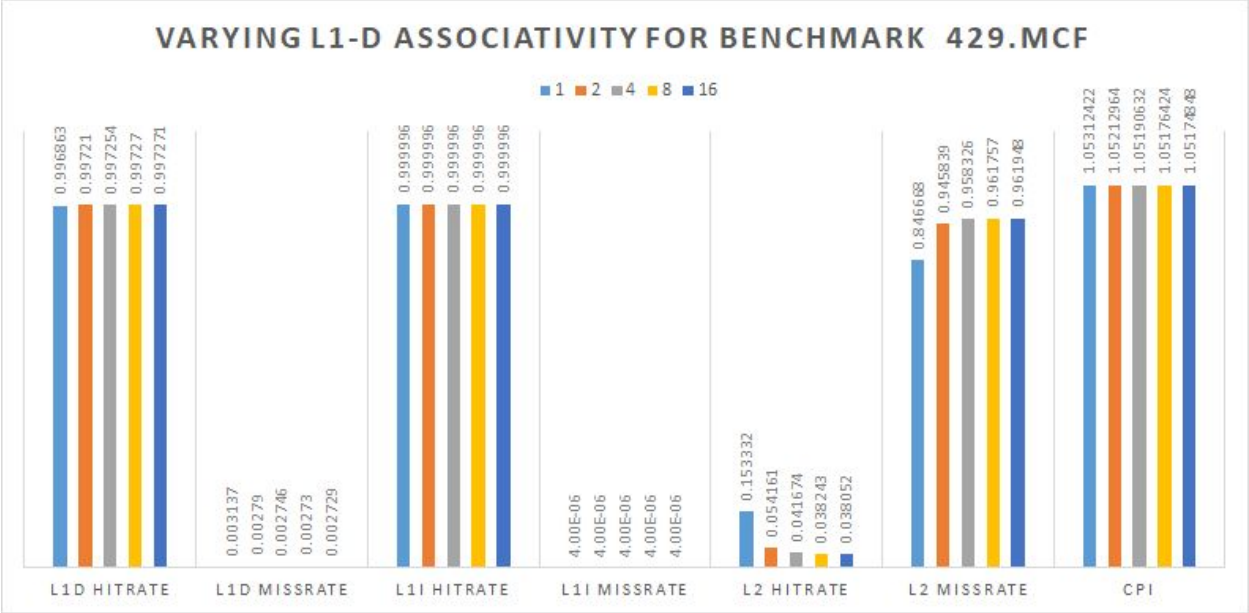
Explanation

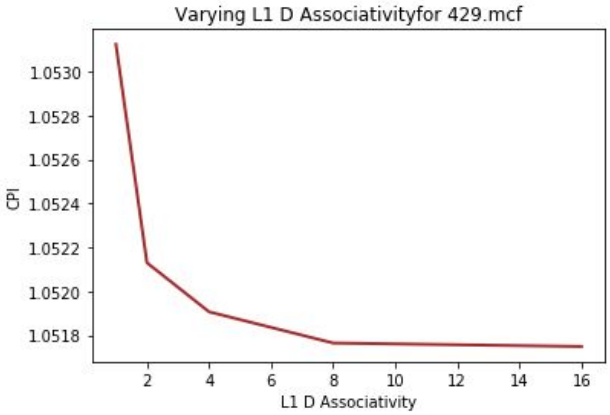
- L2 hit-rate increases with increase in L2 cache size.
- Everything else remains more or less constant with a low CPI.

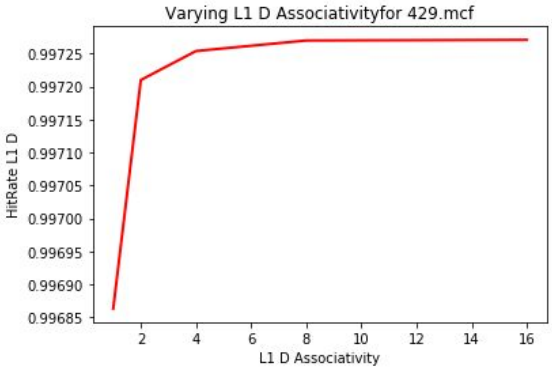
4. Varying L1-D Associativity

Varying L1 D Associative Cache Size for 429.mcf

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.996863	0.003137	0.999996	4e-06	0.153332	0.846668	1.05312
2	0.99721	0.00279	0.999996	4e-06	0.054161	0.945839	1.05213
4	0.997254	0.002746	0.999996	4e-06	0.041674	0.958326	1.05191
8	0.99727	0.00273	0.999996	4e-06	0.038243	0.961757	1.05176
16	0.997271	0.002729	0.999996	4e-06	0.038052	0.961948	1.05175







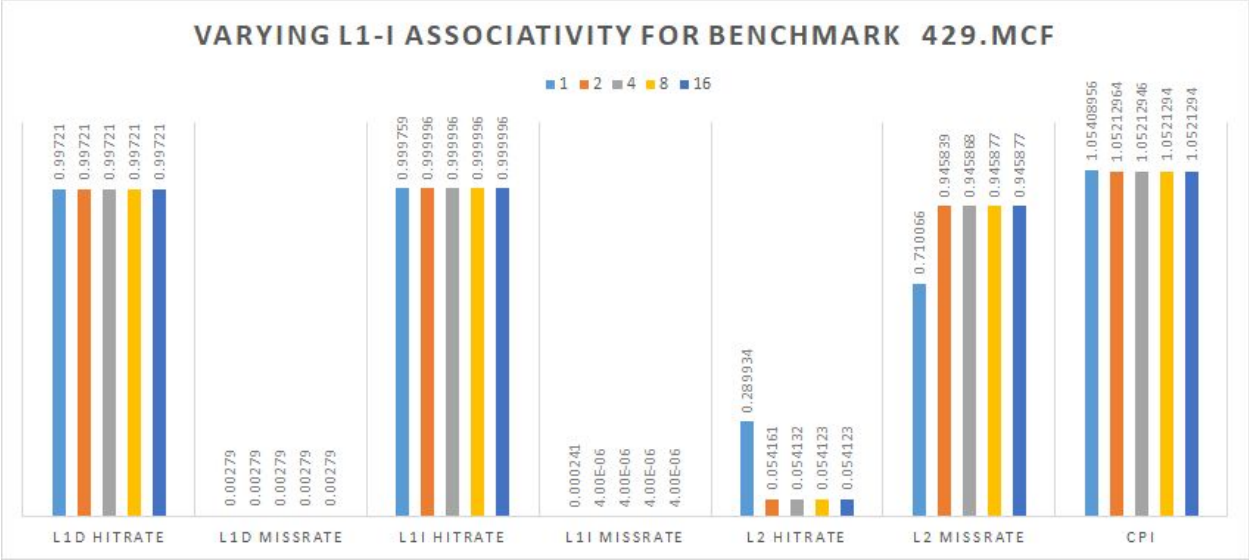
Explanation

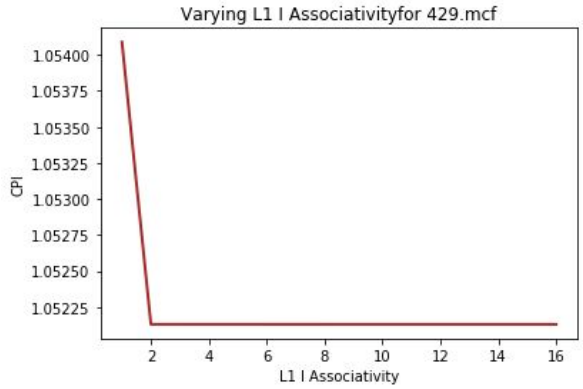
- CPI reduces with increase in associativity.
- Hit-rate of L1-D increases as well
- L2 miss-rate increases.

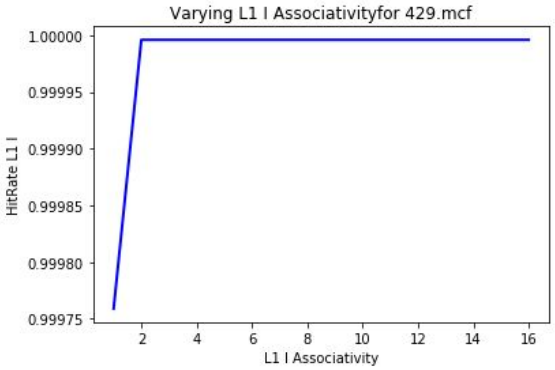
5. Varying L1-I Associativity

Varying L1 I Associative Cache Size for 429.mcf

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.99721	0.00279	0.999759	0.000241	0.289934	0.710066	1.05409
2	0.99721	0.00279	0.999996	4e-06	0.054161	0.945839	1.05213
4	0.99721	0.00279	0.999996	4e-06	0.054132	0.945868	1.05213
8	0.99721	0.00279	0.999996	4e-06	0.054123	0.945877	1.05213
16	0.99721	0.00279	0.999996	4e-06	0.054123	0.945877	1.05213







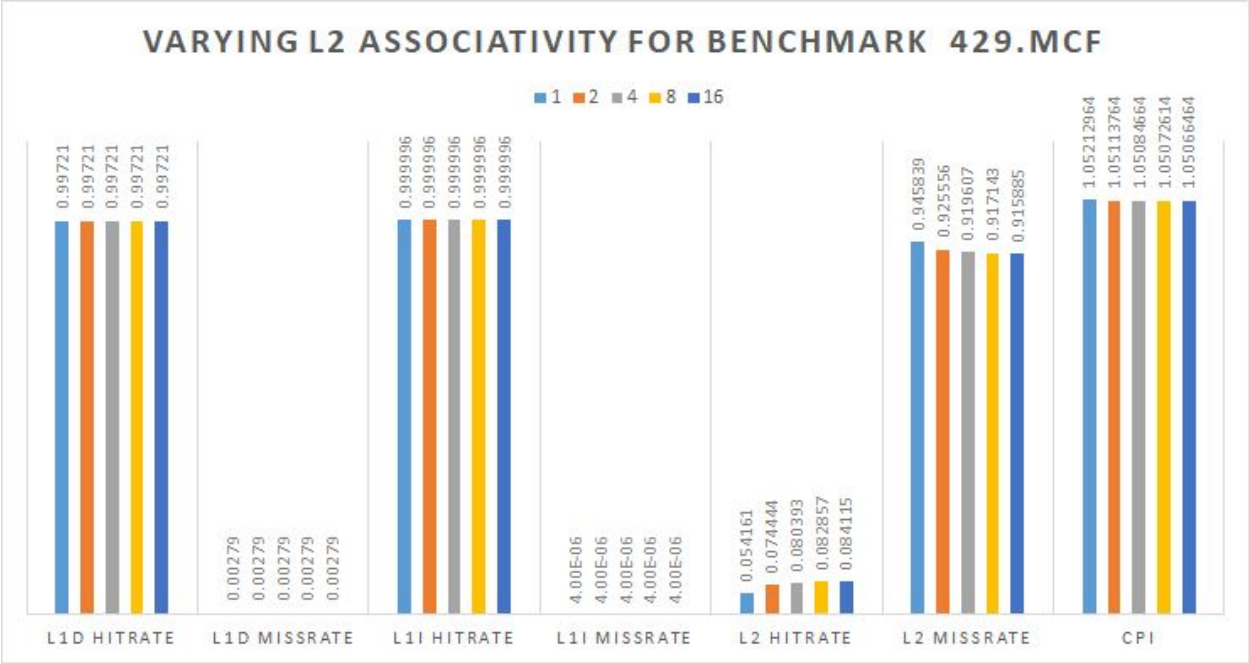
Explanation

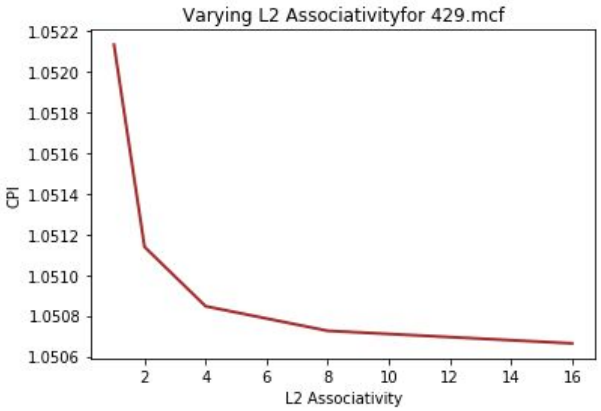
- An increase is seen in the L2 miss-rate which then becomes constant. This is because, as associativity increases, L1 hit-rate increases which intern reduces the L2 hit-rate.

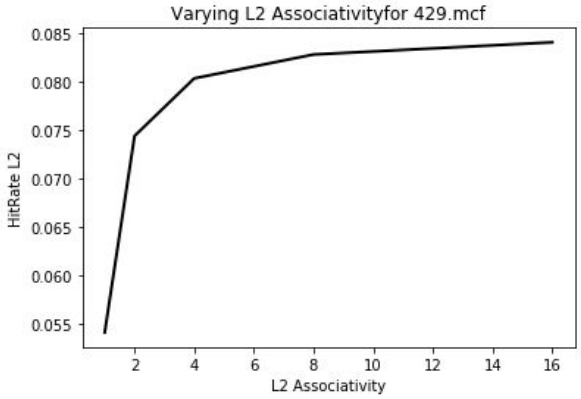
6. Varying L2 Associativity

Varying L2 Associative Cache Size for 429.mcf

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.99721	0.00279	0.999996	4e-06	0.054161	0.945839	1.05213
2	0.99721	0.00279	0.999996	4e-06	0.074444	0.925556	1.05114
4	0.99721	0.00279	0.999996	4e-06	0.080393	0.919607	1.05085
8	0.99721	0.00279	0.999996	4e-06	0.082857	0.917143	1.05073
16	0.99721	0.00279	0.999996	4e-06	0.084115	0.915885	1.05066







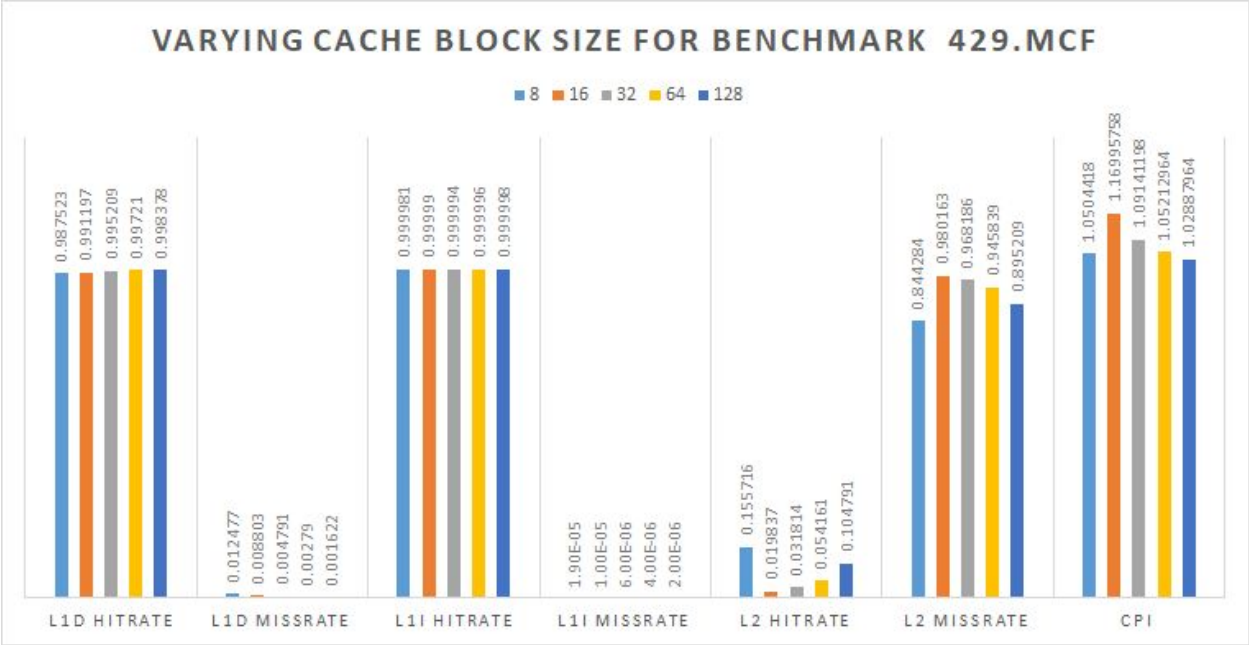
Explanation

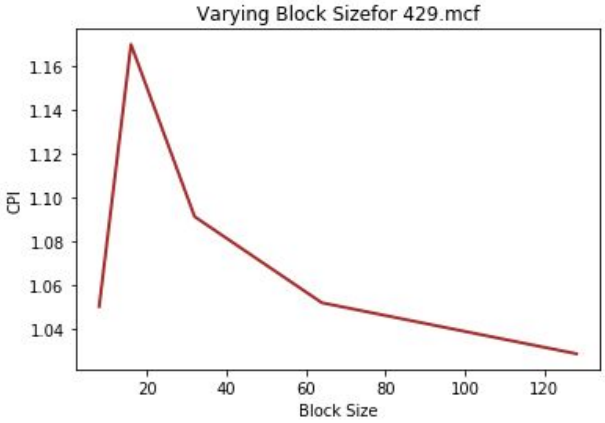
- A slight decrease in L1 hit rates occur due to which L2 miss rate increases.

7. Varying Cache Block size

Varying Cache Block Size for 429.mcf

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8	0.987523	0.012477	0.999981	1.9e-05	0.155716	0.844284	1.05044
16	0.991197	0.008803	0.99999	1e-05	0.019837	0.980163	1.16996
32	0.995209	0.004791	0.999994	6e-06	0.031814	0.968186	1.09141
64	0.99721	0.00279	0.999996	4e-06	0.054161	0.945839	1.05213
128	0.998378	0.001622	0.999998	2e-06	0.104791	0.895209	1.02888





Explanation

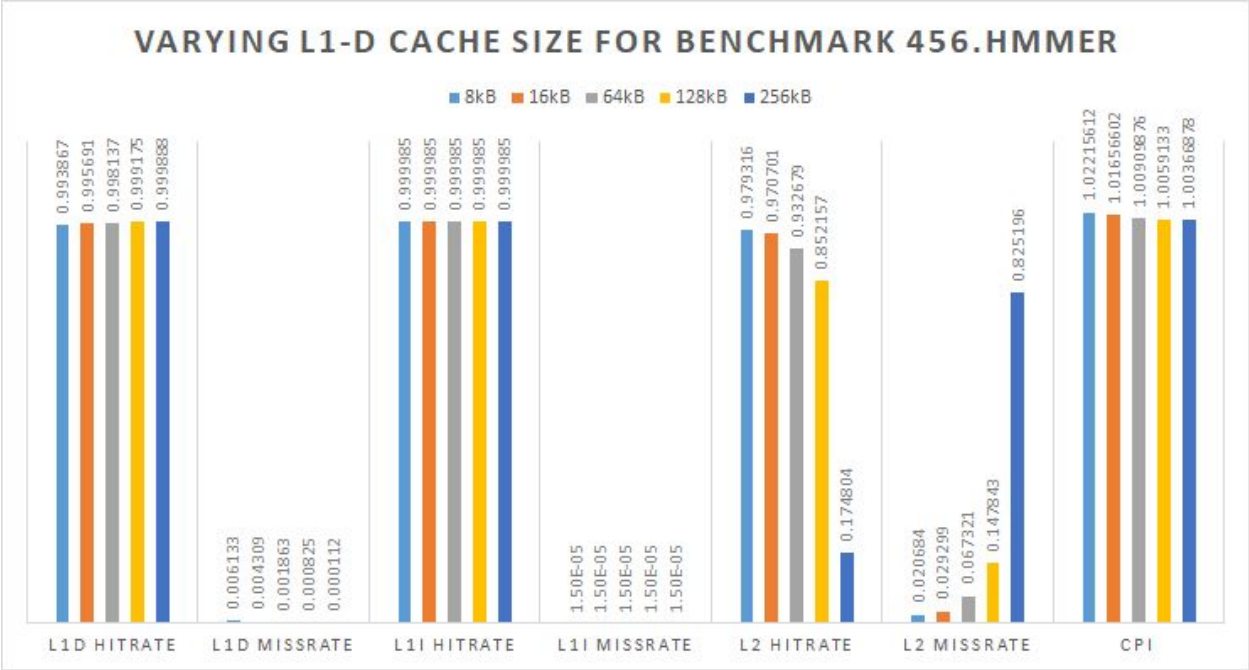
- The CPI spikes and reduces with increase in block size due to the spatial locality.
- The L2 miss-rate also reduces after a while if block-size is continually increased. The larger the block size, the less the number of entries in the cache, and the more the competition between program data for these entries, hence the higher miss-rate after a while.

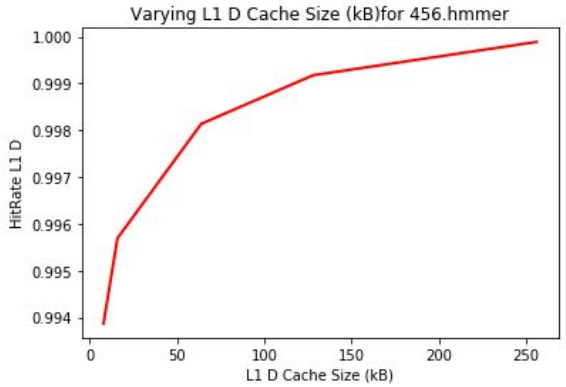
Benchmark : 456.hmmer

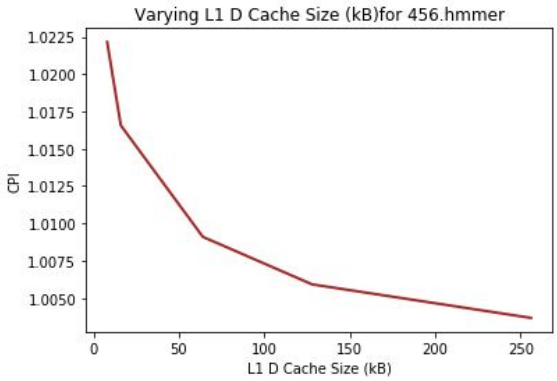
1. Varying L1-D cache size

Varying L1-D Cache Size for 456.hmmer

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8kB	0.993867	0.006133	0.999985	1.5e-05	0.979316	0.020684	1.02216
16kB	0.995691	0.004309	0.999985	1.5e-05	0.970701	0.029299	1.01657
64kB	0.998137	0.001863	0.999985	1.5e-05	0.932679	0.067321	1.0091
128kB	0.999175	0.000825	0.999985	1.5e-05	0.852157	0.147843	1.00591
256kB	0.999888	0.000112	0.999985	1.5e-05	0.174804	0.825196	1.00367







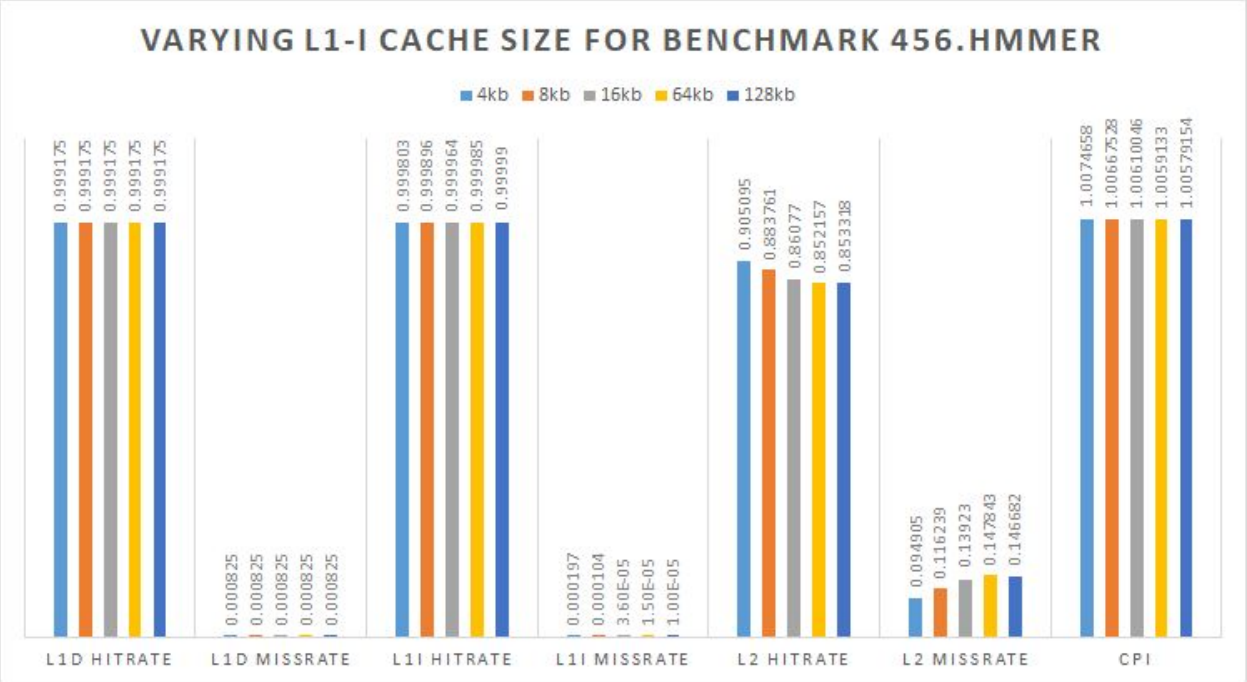
Explanation

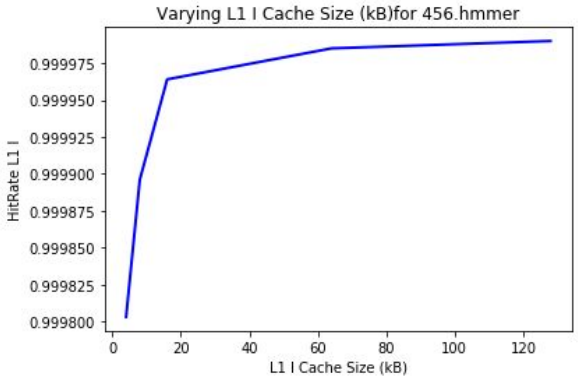
- The CPI decreases with increase in cache size and similarly, the hit-rte of L1D increases.
- The bar-plot shows a significant decrease in l2 hit-rate when the L1-D cache becomes larger. This is again to initial misses which will become larger as the L1 D cache size is increases.
- This appears to be a well performing benchmark since L1 D and I hit-rates are high completes instructions in less number of cycles.

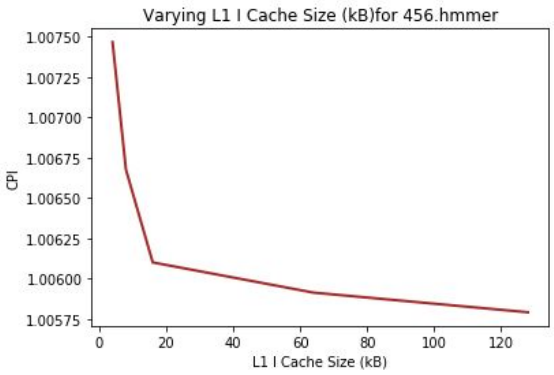
2. Varying L1-I cache size

Varying L1-I Cache Size for 456.hmmer

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
4kb	0.999175	0.000825	0.999803	0.000197	0.905095	0.094905	1.00747
8kb	0.999175	0.000825	0.999896	0.000104	0.883761	0.116239	1.00668
16kb	0.999175	0.000825	0.999964	3.6e-05	0.86077	0.13923	1.0061
64kb	0.999175	0.000825	0.999985	1.5e-05	0.852157	0.147843	1.00591
128kb	0.999175	0.000825	0.99999	1e-05	0.853318	0.146682	1.00579







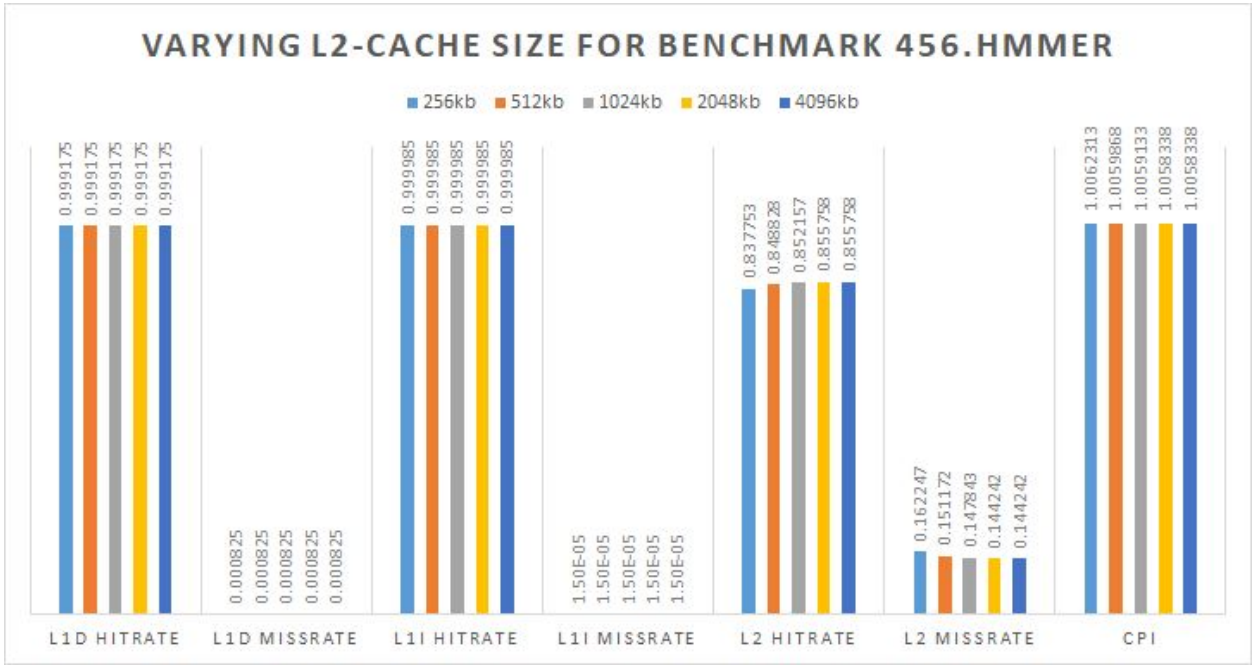
Explanation

- Follows similar trends as the other runs with increasing L1-I cache.
- This benchmark performs very well with high L1 hit-rates and increasing L2 miss-rate.

3. Varying L2 cache size

Varying L2 Cache Size for 456.hmmer

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
256kb	0.999175	0.000825	0.999985	1.5e-05	0.837753	0.162247	1.00623
512kb	0.999175	0.000825	0.999985	1.5e-05	0.848828	0.151172	1.00599
1024kb	0.999175	0.000825	0.999985	1.5e-05	0.852157	0.147843	1.00591
2048kb	0.999175	0.000825	0.999985	1.5e-05	0.855758	0.144242	1.00583
4096kb	0.999175	0.000825	0.999985	1.5e-05	0.855758	0.144242	1.00583



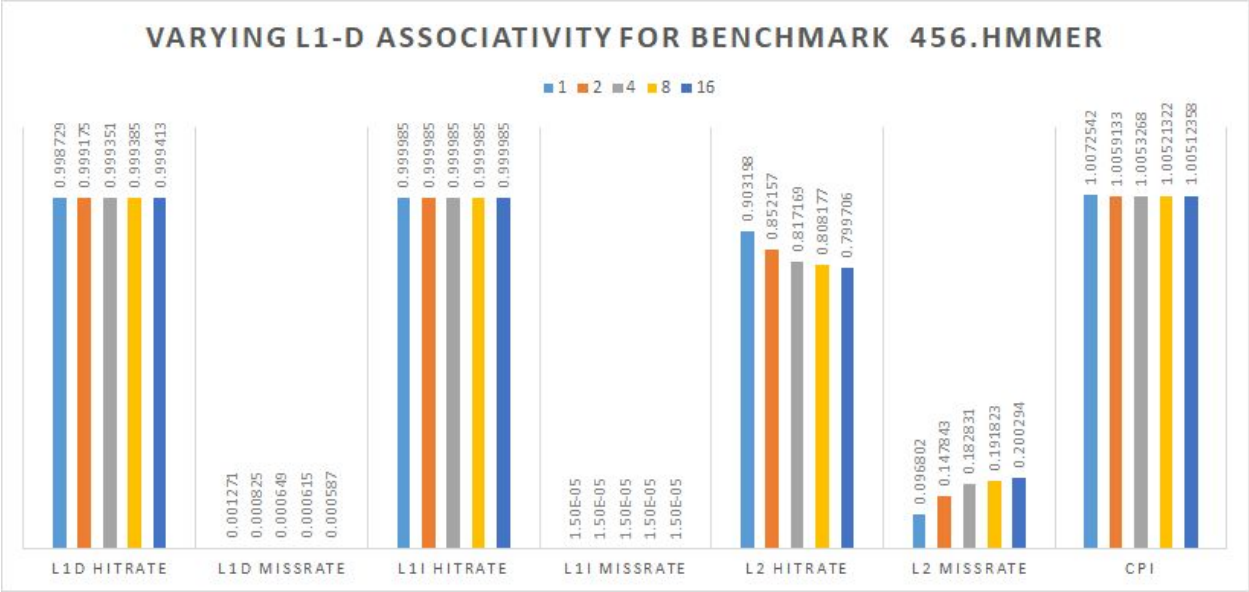
Explanation

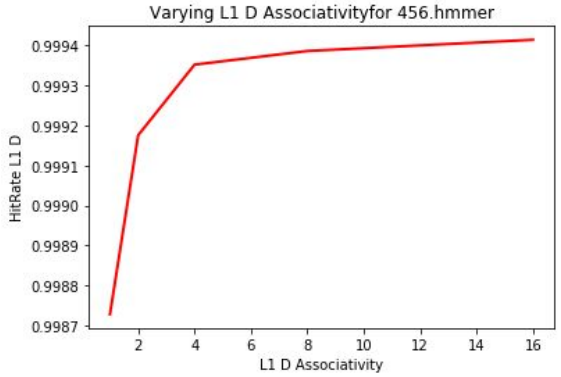
- Instructions perform very well on this benchmark with a very low CPI of around 1 and a constantly high L1 I and D miss-rate.

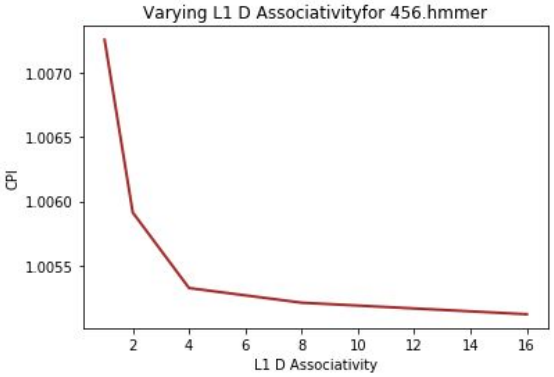
4. Varying L1-D Associativity

Varying L1 D Associative Cache Size for 456.hmmer

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.998729	0.001271	0.999985	1.5e-05	0.903198	0.096802	1.00725
2	0.999175	0.000825	0.999985	1.5e-05	0.852157	0.147843	1.00591
4	0.999351	0.000649	0.999985	1.5e-05	0.817169	0.182831	1.00533
8	0.999385	0.000615	0.999985	1.5e-05	0.808177	0.191823	1.00521
16	0.999413	0.000587	0.999985	1.5e-05	0.799706	0.200294	1.00512







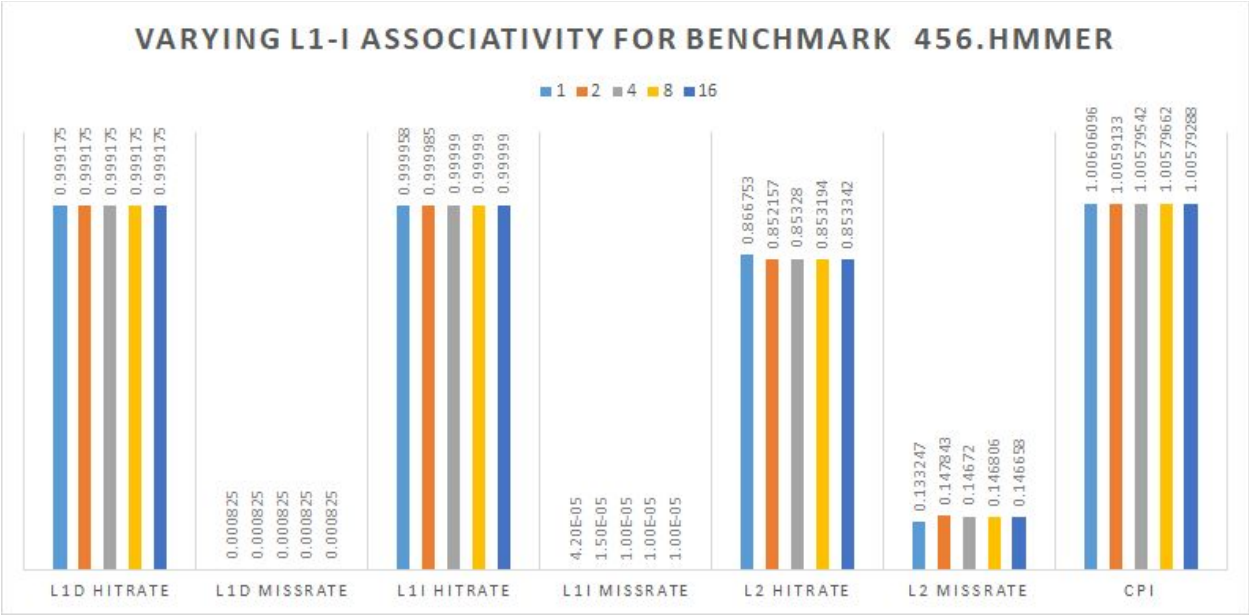
Explanation

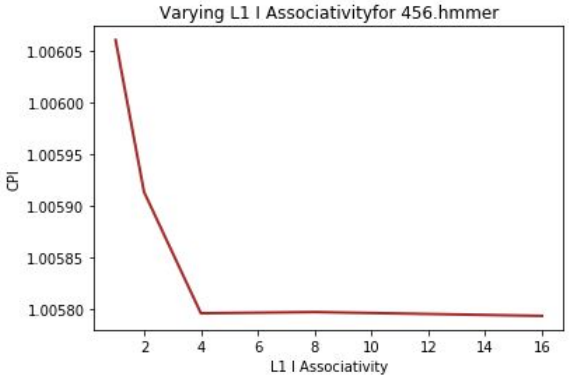
- L2 miss-rate increases with associativity.

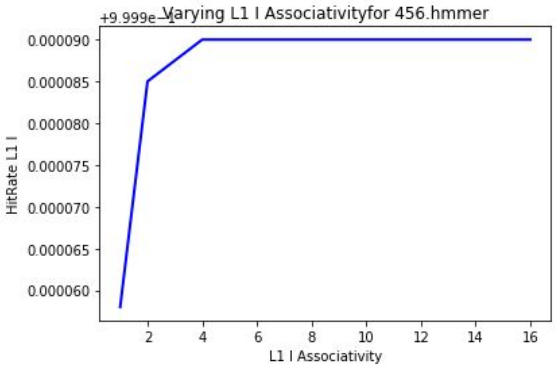
5. Varying L1-I Associativity

Varying L1 I Associative Cache Size for 456.hmmer

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.999175	0.000825	0.999958	4.2e-05	0.866753	0.133247	1.00606
2	0.999175	0.000825	0.999985	1.5e-05	0.852157	0.147843	1.00591
4	0.999175	0.000825	0.99999	1e-05	0.85328	0.14672	1.0058
8	0.999175	0.000825	0.99999	1e-05	0.853194	0.146806	1.0058
16	0.999175	0.000825	0.99999	1e-05	0.853342	0.146658	1.00579







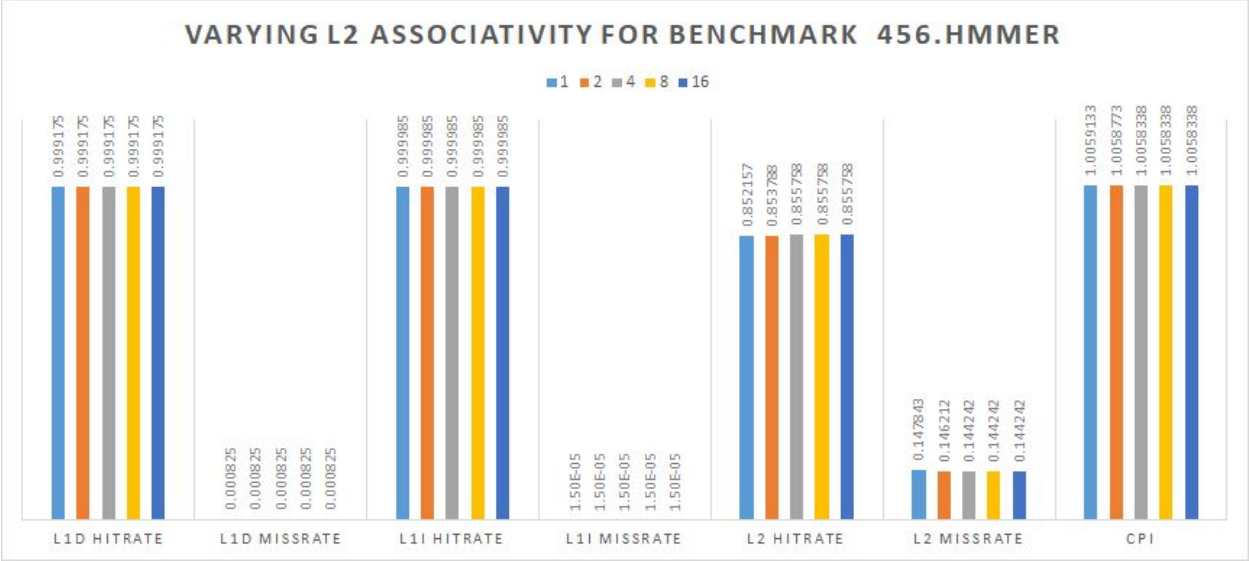
Explanation

- Performance on this benchmark is quite good and there is very slight changes due to change in associativity.

6. Varying L2 Associativity

Varying L2 Associative Cache Size for 456.hmmer

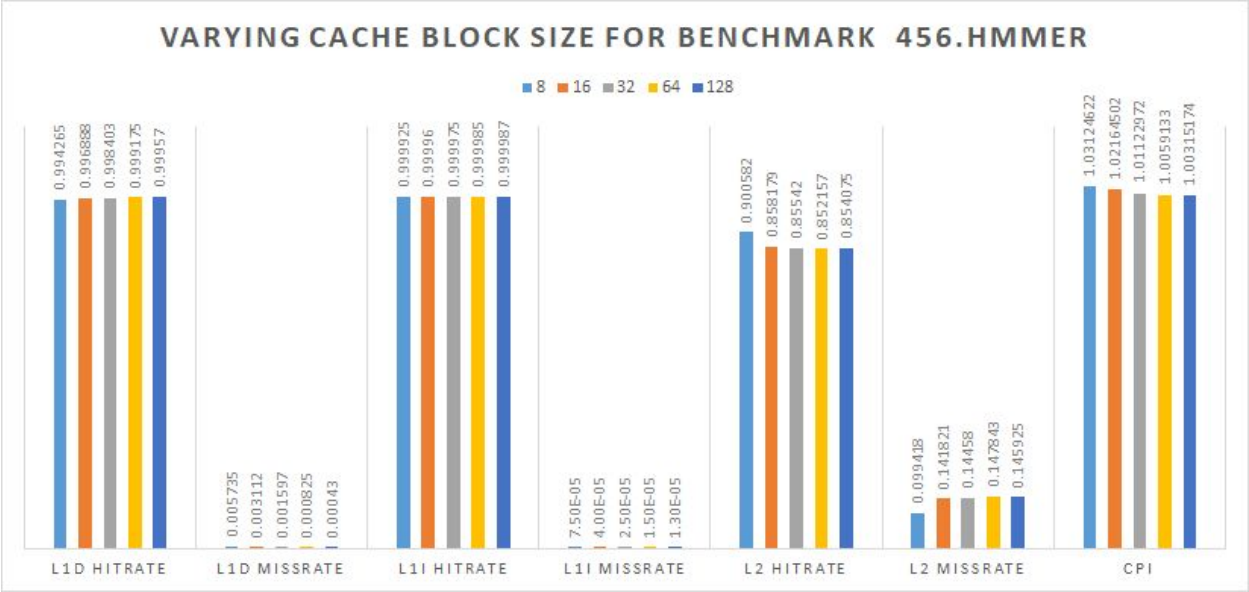
Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.999175	0.000825	0.999985	1.5e-05	0.852157	0.147843	1.00591
2	0.999175	0.000825	0.999985	1.5e-05	0.853788	0.146212	1.00588
4	0.999175	0.000825	0.999985	1.5e-05	0.855758	0.144242	1.00583
8	0.999175	0.000825	0.999985	1.5e-05	0.855758	0.144242	1.00583
16	0.999175	0.000825	0.999985	1.5e-05	0.855758	0.144242	1.00583

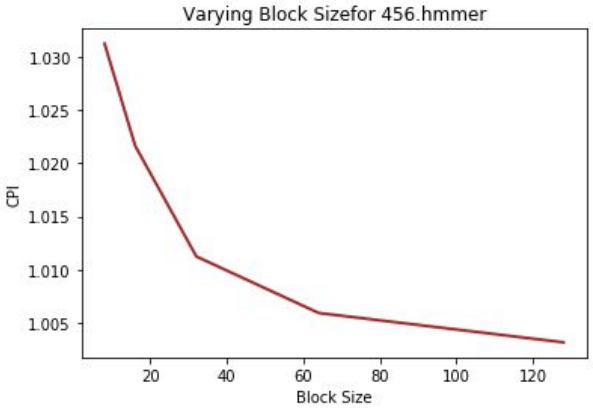


7. Varying Cache Block size

Varying Cache Block Size for 456.hmmer

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8	0.994265	0.005735	0.999925	7.5e-05	0.900582	0.099418	1.03125
16	0.996888	0.003112	0.99996	4e-05	0.858179	0.141821	1.02165
32	0.998403	0.001597	0.999975	2.5e-05	0.85542	0.14458	1.01123
64	0.999175	0.000825	0.999985	1.5e-05	0.852157	0.147843	1.00591
128	0.99957	0.00043	0.999987	1.3e-05	0.854075	0.145925	1.00315





Explanation

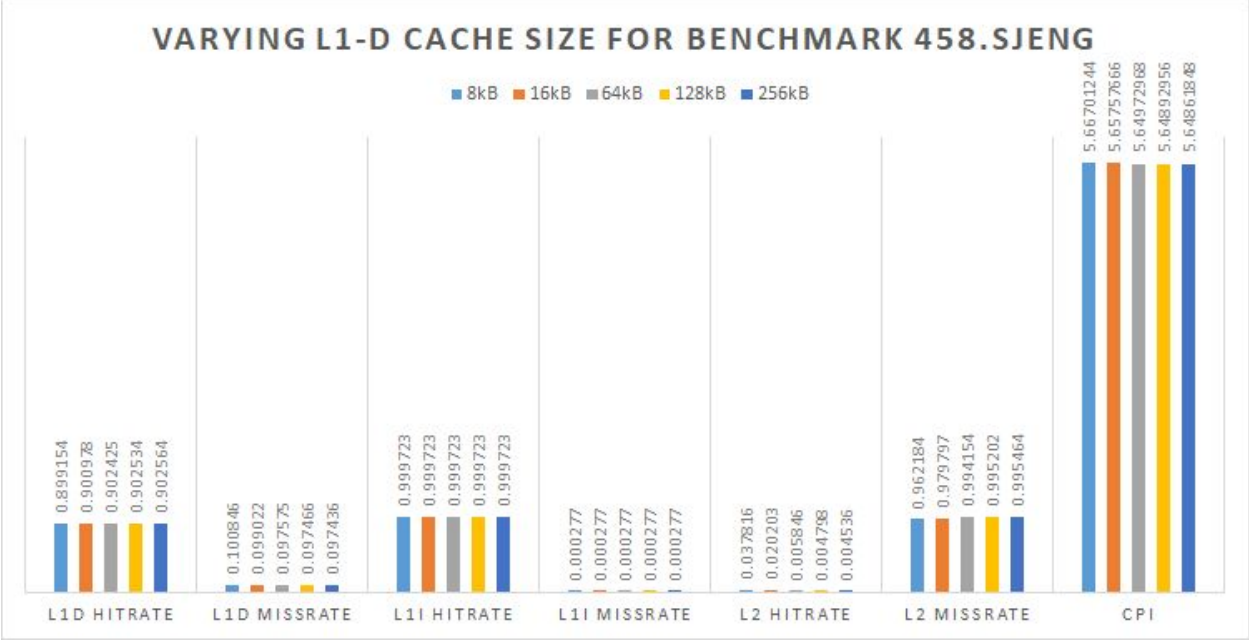
- Increase in block size increase the L2 miss-rate due to improvement in L1 hir-rate which in turn reduces the CPI.

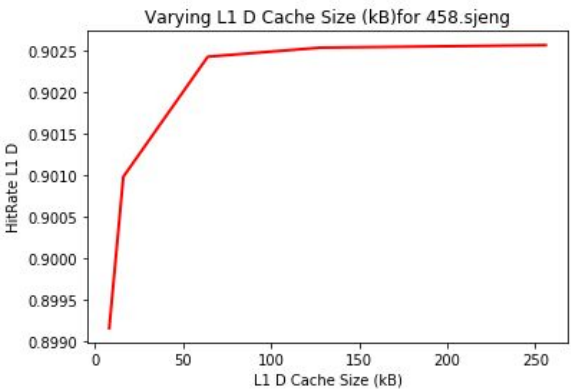
Benchmark : 458.sjeng

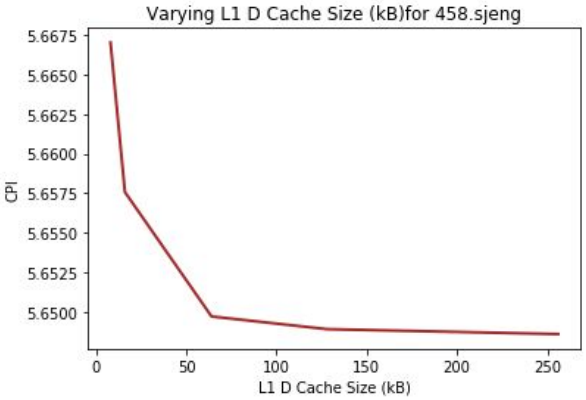
1. Varying L1-D cache size

Varying L1-D Cache Size for 458.sjeng

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8kB	0.899154	0.100846	0.999723	0.000277	0.037816	0.962184	5.66701
16kB	0.900978	0.099022	0.999723	0.000277	0.020203	0.979797	5.65758
64kB	0.902425	0.097575	0.999723	0.000277	0.005846	0.994154	5.64973
128kB	0.902534	0.097466	0.999723	0.000277	0.004798	0.995202	5.64893
256kB	0.902564	0.097436	0.999723	0.000277	0.004536	0.995464	5.64862







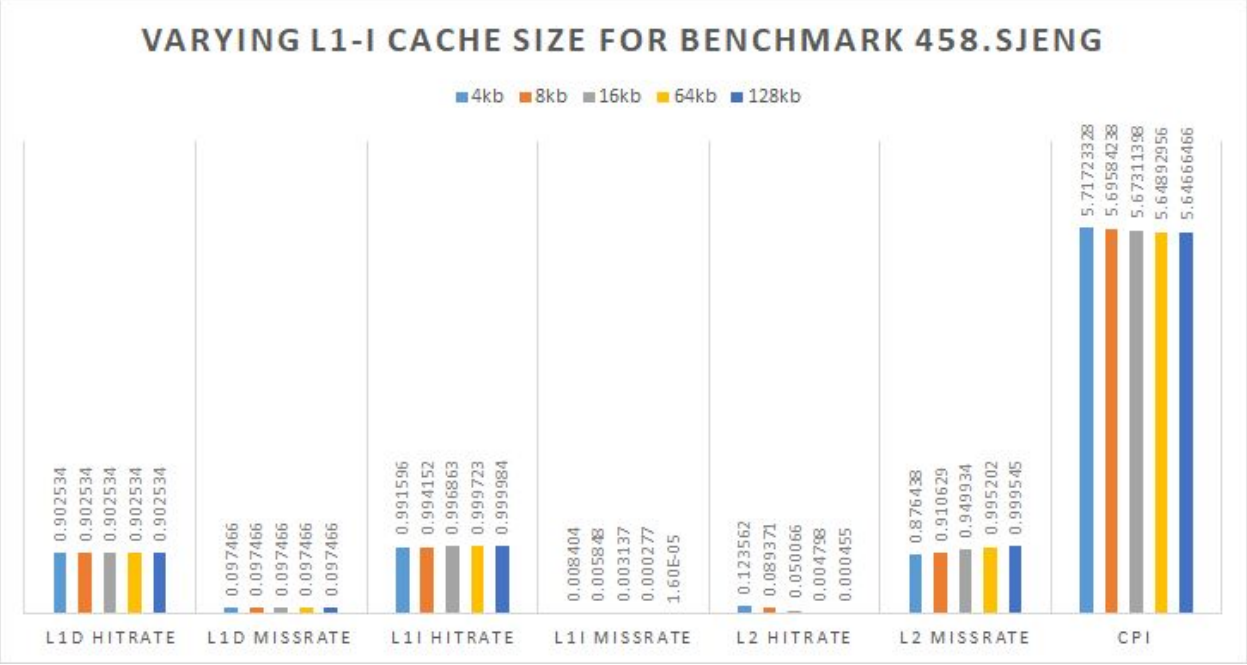
Explanation

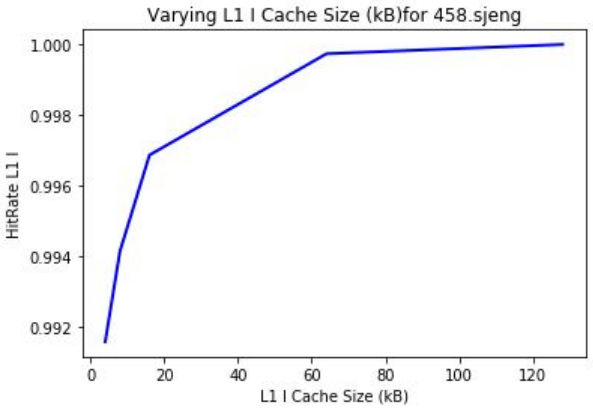
- This benchmark follows the usual trend for hit-rate increase but negligibly.
- The values are mostly constant with a very high CPI, i.e our instructions perform poorly on this benchmark.

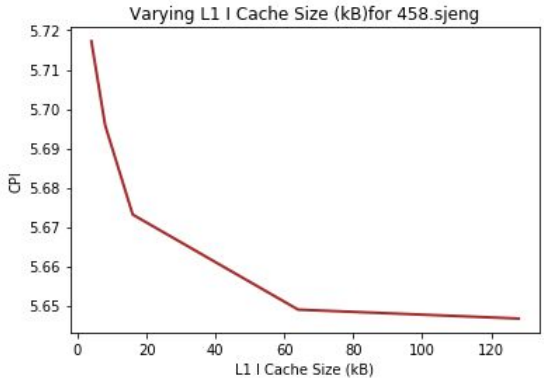
2. Varying L1-I cache size

Varying L1-I Cache Size for 458.sjeng

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
4kb	0.902534	0.097466	0.991596	0.008404	0.123562	0.876438	5.71723
8kb	0.902534	0.097466	0.994152	0.005848	0.089371	0.910629	5.69584
16kb	0.902534	0.097466	0.996863	0.003137	0.050066	0.949934	5.67311
64kb	0.902534	0.097466	0.999723	0.000277	0.004798	0.995202	5.64893
128kb	0.902534	0.097466	0.999984	1.6e-05	0.000455	0.999545	5.64666







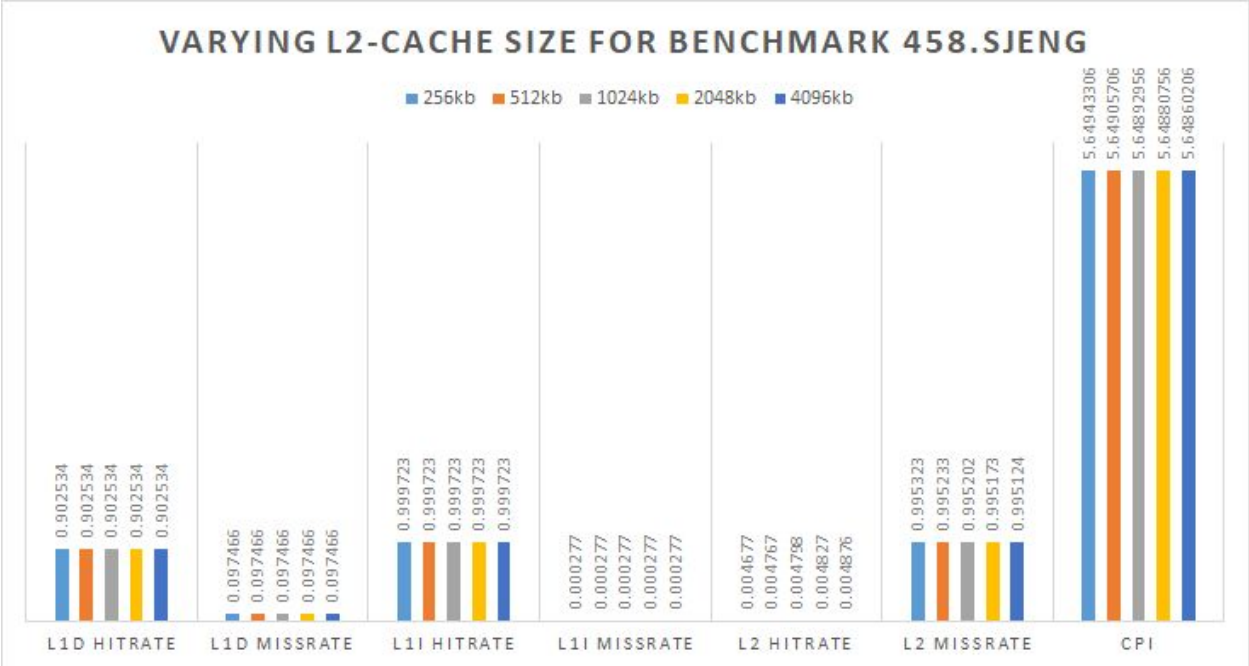
Explanation

- Performance on this benchmark is very poor with a very high CPI and very low L1 hit and miss-rates.

3. Varying L2 cache size

Varying L2 Cache Size for 458.sjeng

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
256kb	0.902534	0.097466	0.999723	0.000277	0.004677	0.995323	5.64943
512kb	0.902534	0.097466	0.999723	0.000277	0.004767	0.995233	5.64906
1024kb	0.902534	0.097466	0.999723	0.000277	0.004798	0.995202	5.64893
2048kb	0.902534	0.097466	0.999723	0.000277	0.004827	0.995173	5.64881
4096kb	0.902534	0.097466	0.999723	0.000277	0.004876	0.995124	5.6486



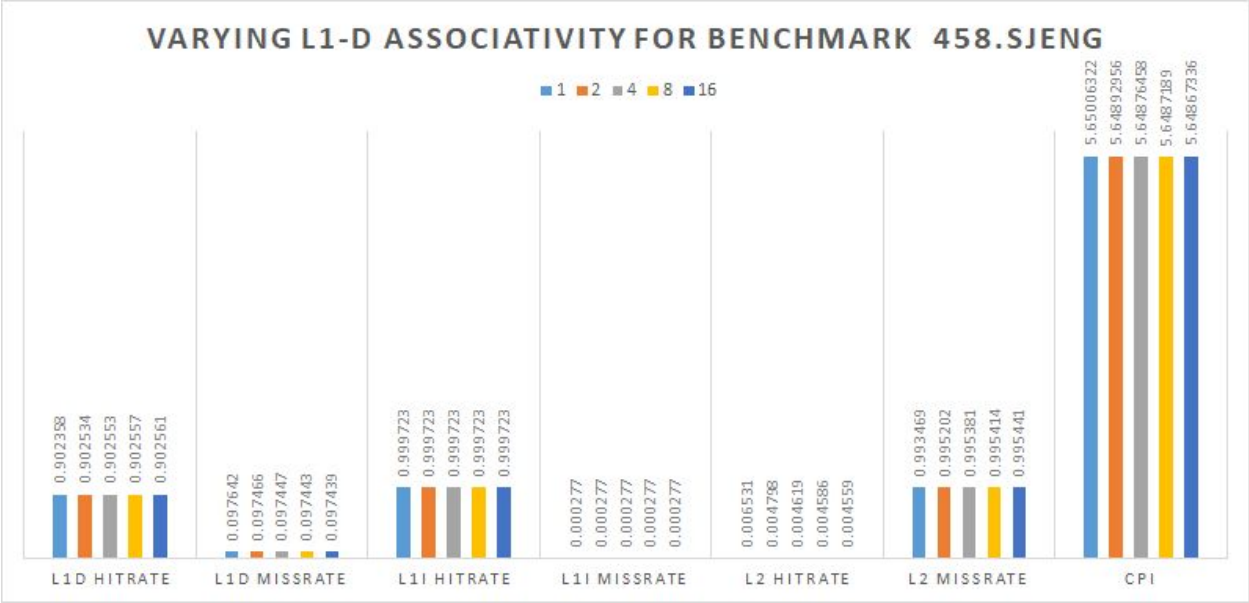
Explanation

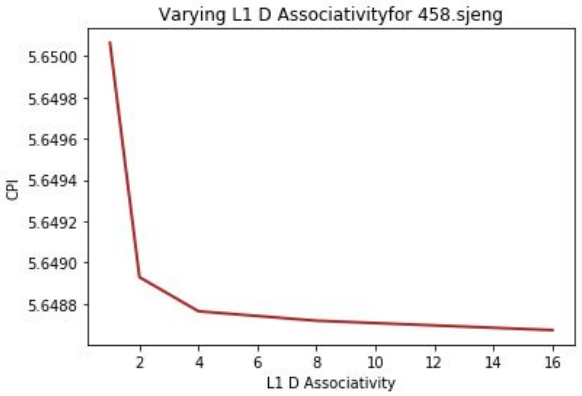
- Performance on this benchmark is quite back and there is barely any change except for a slight decrease in the L2 miss-rate.

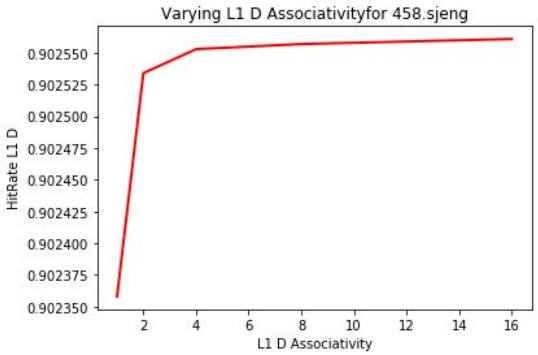
4. Varying L1-D Associativity

Varying L1 D Associative Cache Size for 458.sjeng

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.902358	0.097642	0.999723	0.000277	0.006531	0.993469	5.65006
2	0.902534	0.097466	0.999723	0.000277	0.004798	0.995202	5.64893
4	0.902553	0.097447	0.999723	0.000277	0.004619	0.995381	5.64876
8	0.902557	0.097443	0.999723	0.000277	0.004586	0.995414	5.64872
16	0.902561	0.097439	0.999723	0.000277	0.004559	0.995441	5.64867







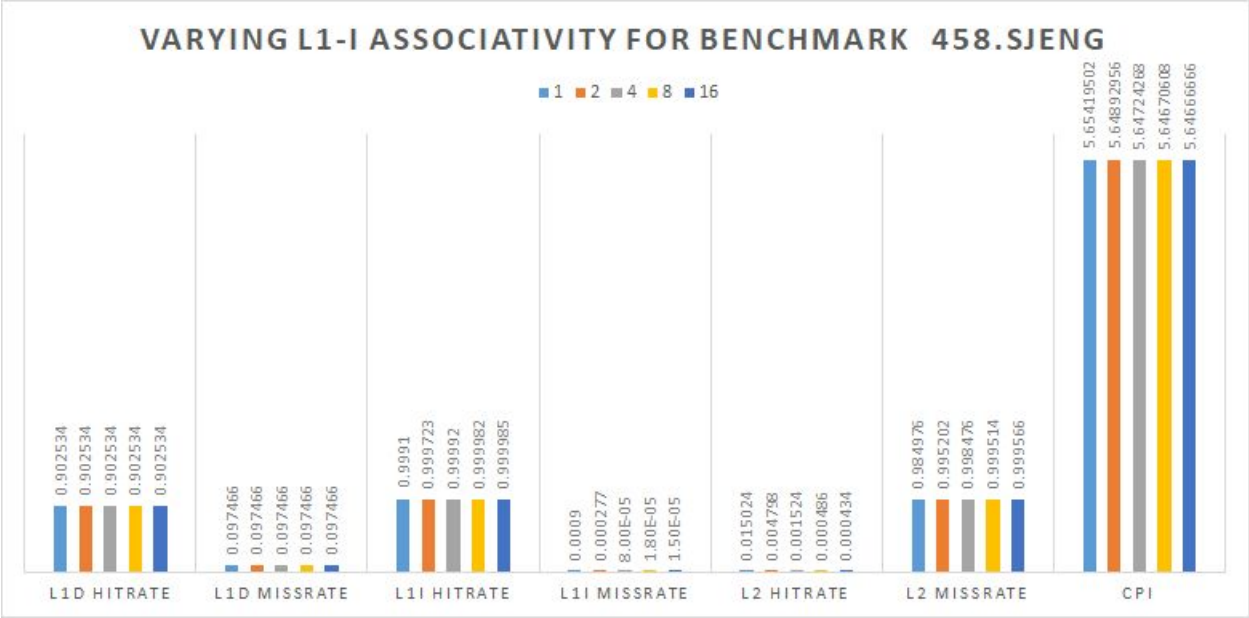
Explanation

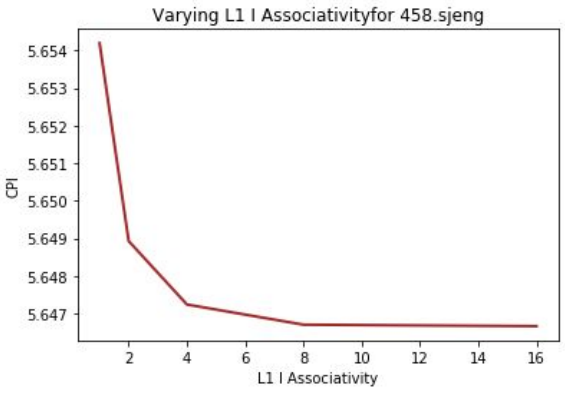
- Performance on this benchmark is quite sub par.

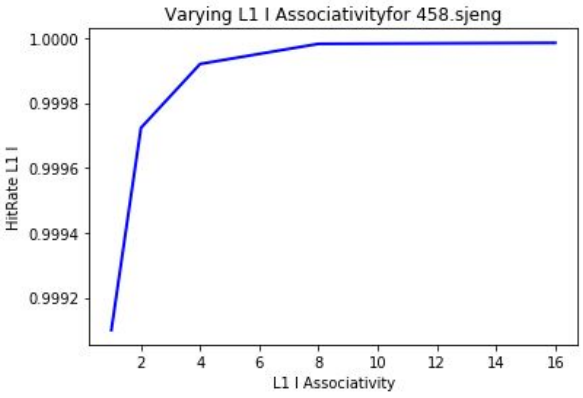
5. Varying L1-I Associativity

Varying L1 I Associative Cache Size for 458.sjeng

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.902534	0.097466	0.9991	0.0009	0.015024	0.984976	5.6542
2	0.902534	0.097466	0.999723	0.000277	0.004798	0.995202	5.64893
4	0.902534	0.097466	0.99992	8e-05	0.001524	0.998476	5.64724
8	0.902534	0.097466	0.999982	1.8e-05	0.000486	0.999514	5.64671
16	0.902534	0.097466	0.999985	1.5e-05	0.000434	0.999566	5.64667







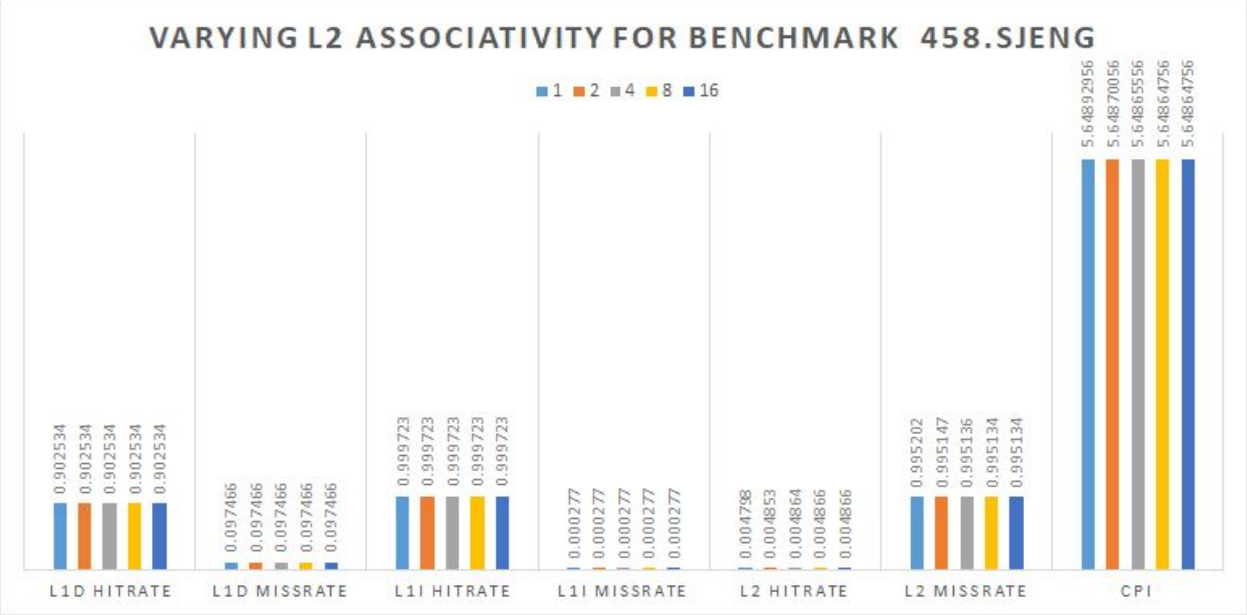
Explanation

- CPI is intermediate and there the effect of change in associativity is not too high.

6. Varying L2 Associativity

Varying L2 Associative Cache Size for 458.sjeng

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.902534	0.097466	0.999723	0.000277	0.004798	0.995202	5.64893
2	0.902534	0.097466	0.999723	0.000277	0.004853	0.995147	5.6487
4	0.902534	0.097466	0.999723	0.000277	0.004864	0.995136	5.64866
8	0.902534	0.097466	0.999723	0.000277	0.004866	0.995134	5.64865
16	0.902534	0.097466	0.999723	0.000277	0.004866	0.995134	5.64865



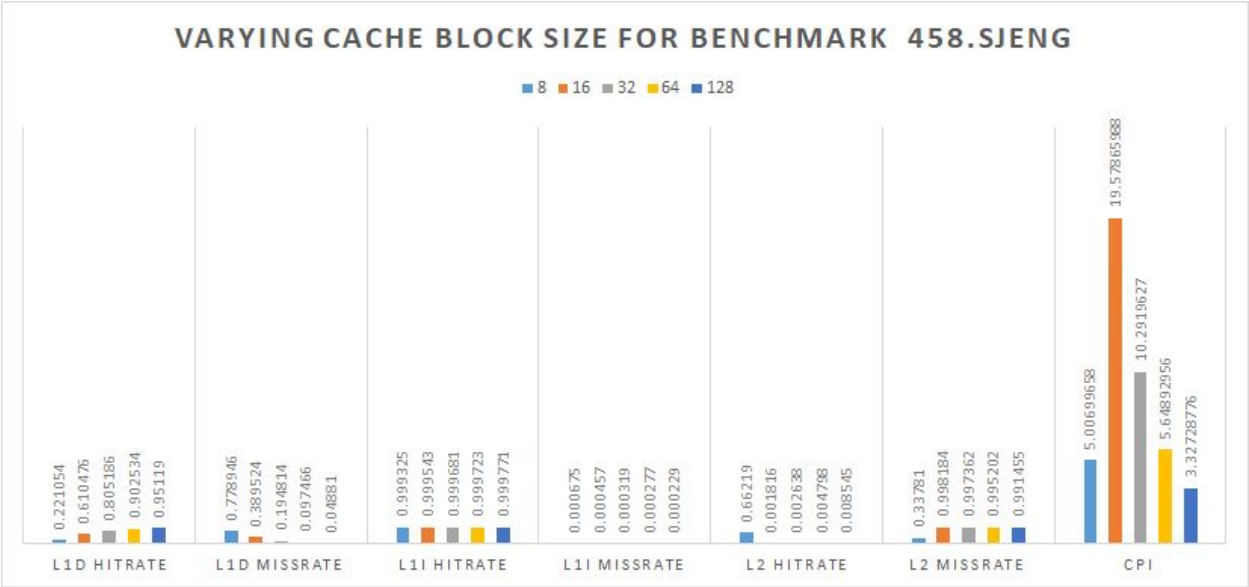
Explanation

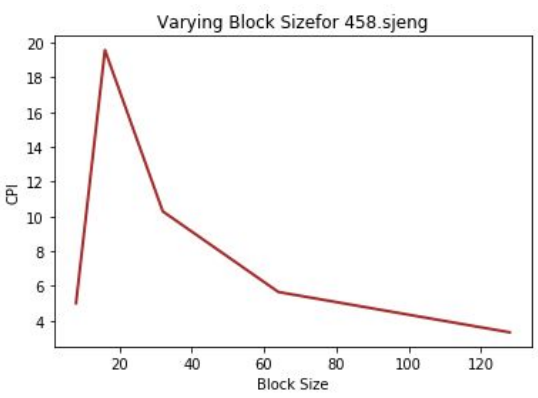
- Poor performance and negligible change due to change in associativity.

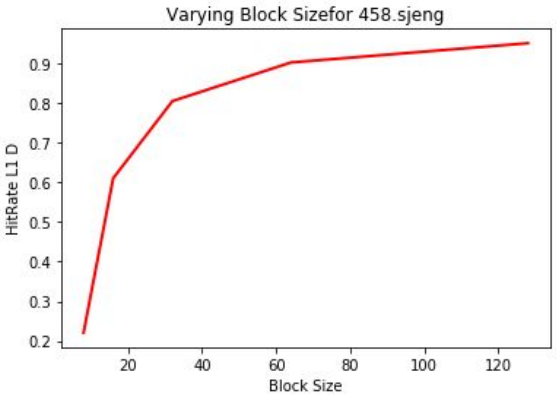
7. Varying Cache Block size

Varying Cache Block Size for 458.sjeng

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8	0.221054	0.778946	0.999325	0.000675	0.66219	0.33781	5.007
16	0.610476	0.389524	0.999543	0.000457	0.001816	0.998184	19.5787
32	0.805186	0.194814	0.999681	0.000319	0.002638	0.997362	10.292
64	0.902534	0.097466	0.999723	0.000277	0.004798	0.995202	5.64893
128	0.95119	0.04881	0.999771	0.000229	0.008545	0.991455	3.32729







Explanation

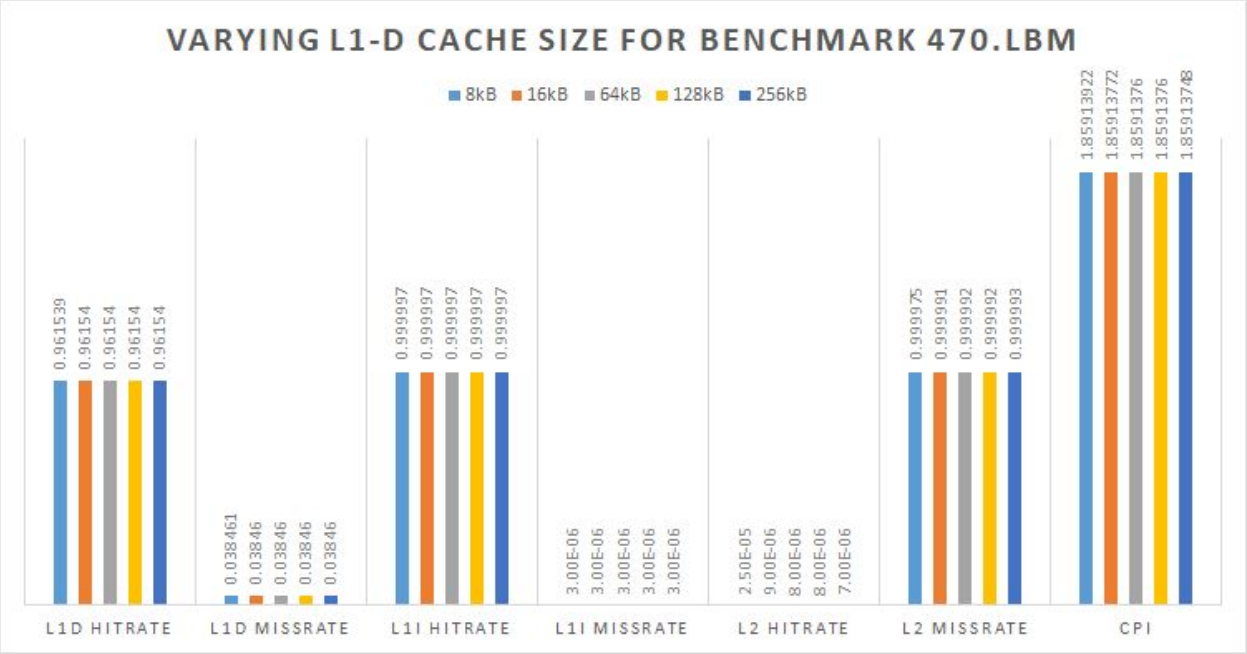
- The performance on this benchmark is mediocre, with widely varying CPI and low hit-rates.
- This may be due to the arbitrariness of the instructions on the benchmark.

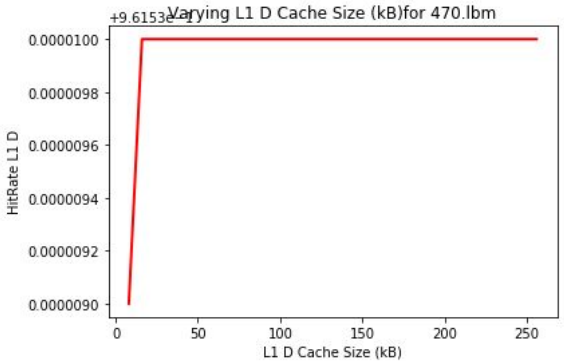
Benchmark : 470.lbm

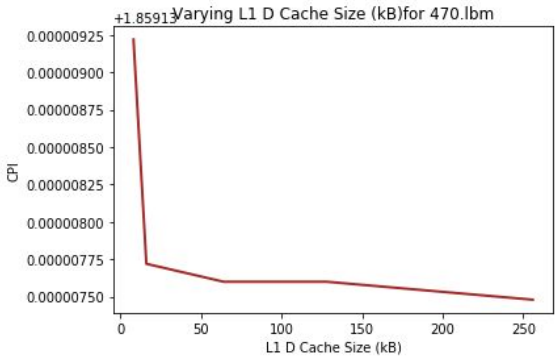
1. Varying L1-D cache size

Varying L1-D Cache Size for 470.lbm

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8kB	0.961539	0.038461	0.999997	3e-06	2.5e-05	0.999975	1.85914
16kB	0.96154	0.03846	0.999997	3e-06	9e-06	0.999991	1.85914
64kB	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
128kB	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
256kB	0.96154	0.03846	0.999997	3e-06	7e-06	0.999993	1.85914







Explanation

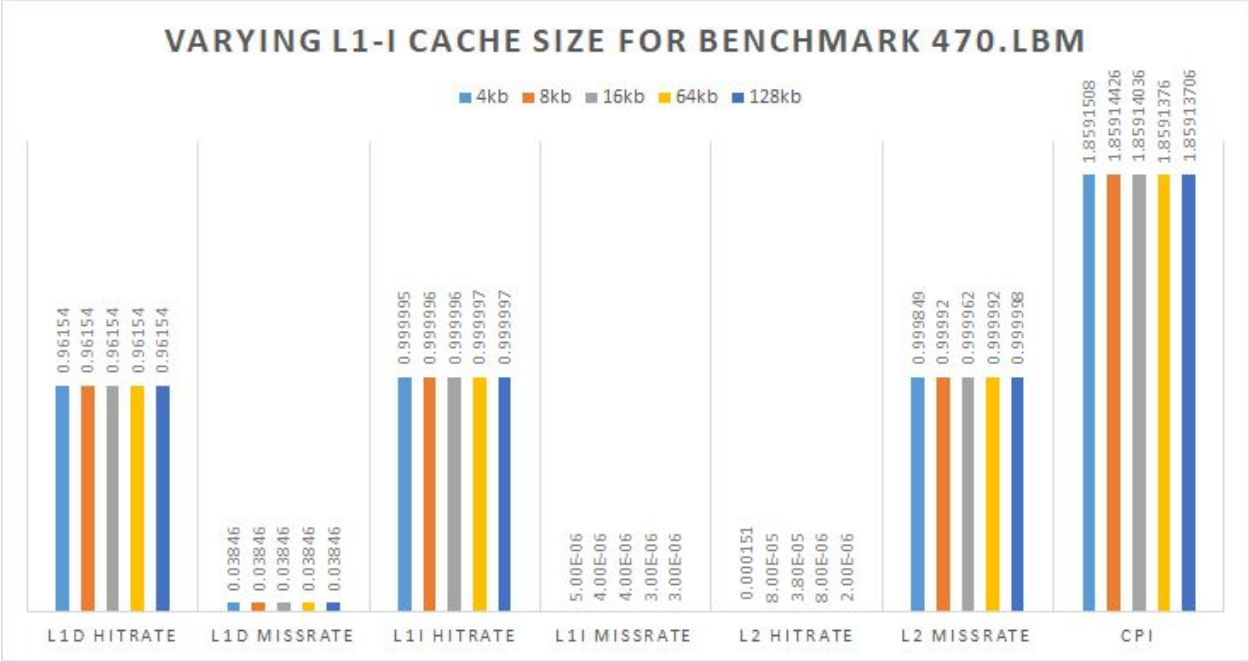
- The L2 hit-rate is very low due to high L1 hit-rates.

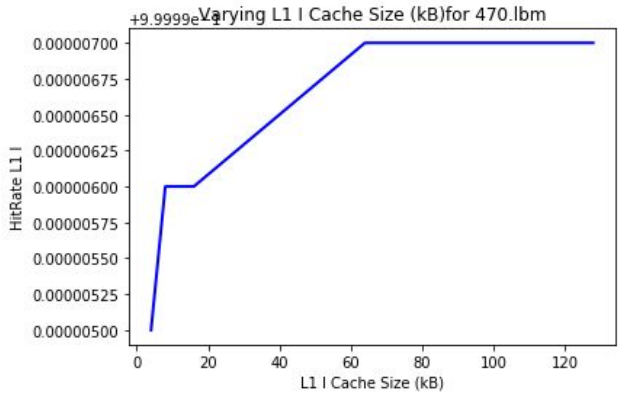
- The CPI sharply reduces as the L1-D cache is increased.

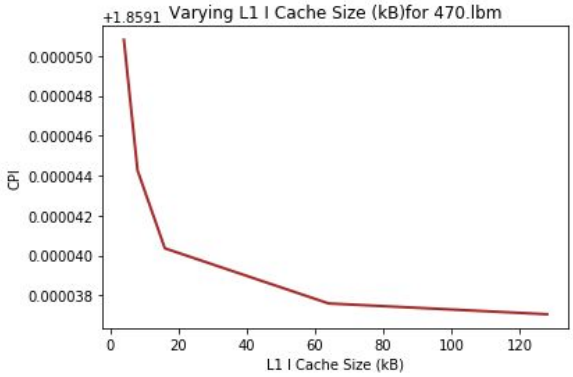
2. Varying L1-I cache size

Varying L1-I Cache Size for 470.lbm

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
4kb	0.96154	0.03846	0.999995	5e-06	0.000151	0.999849	1.85915
8kb	0.96154	0.03846	0.999996	4e-06	8e-05	0.99992	1.85914
16kb	0.96154	0.03846	0.999996	4e-06	3.8e-05	0.999962	1.85914
64kb	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
128kb	0.96154	0.03846	0.999997	3e-06	2e-06	0.999998	1.85914







Explanation

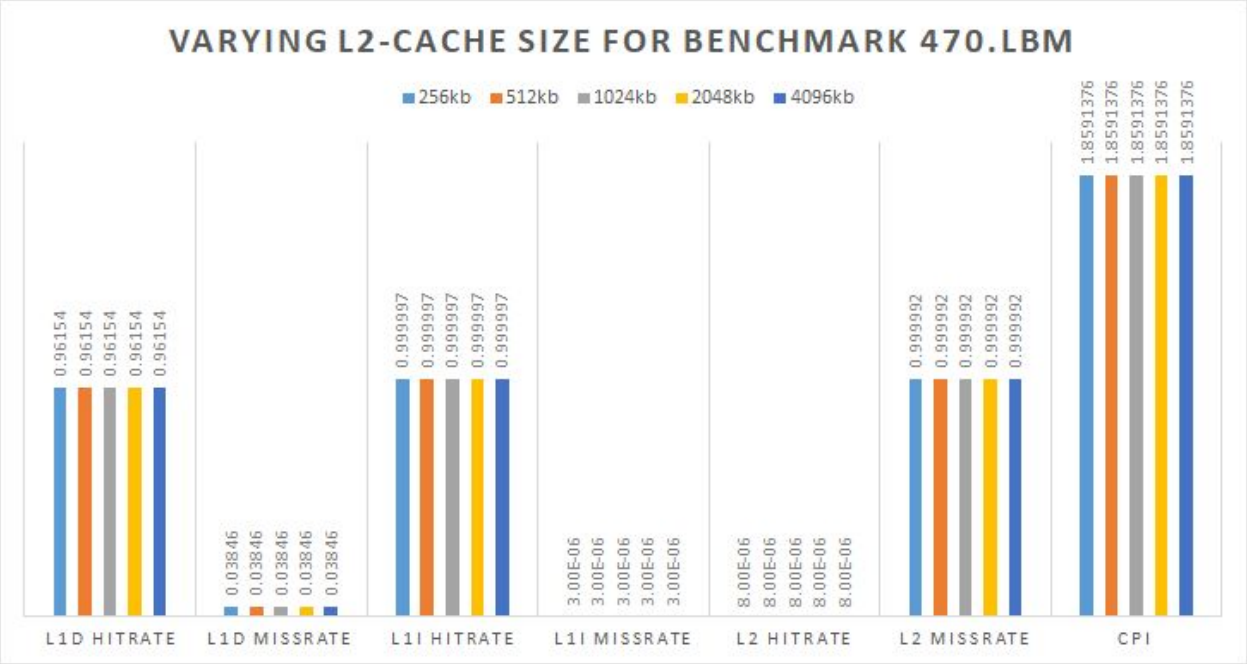
- Inferences are similar to L1-D cache size.

- There is a arbitrary variation in the L1-I hit-rate possibly due to race conditions.

3. Varying L2 cache size

Varying L2 Cache Size for 470.lbm

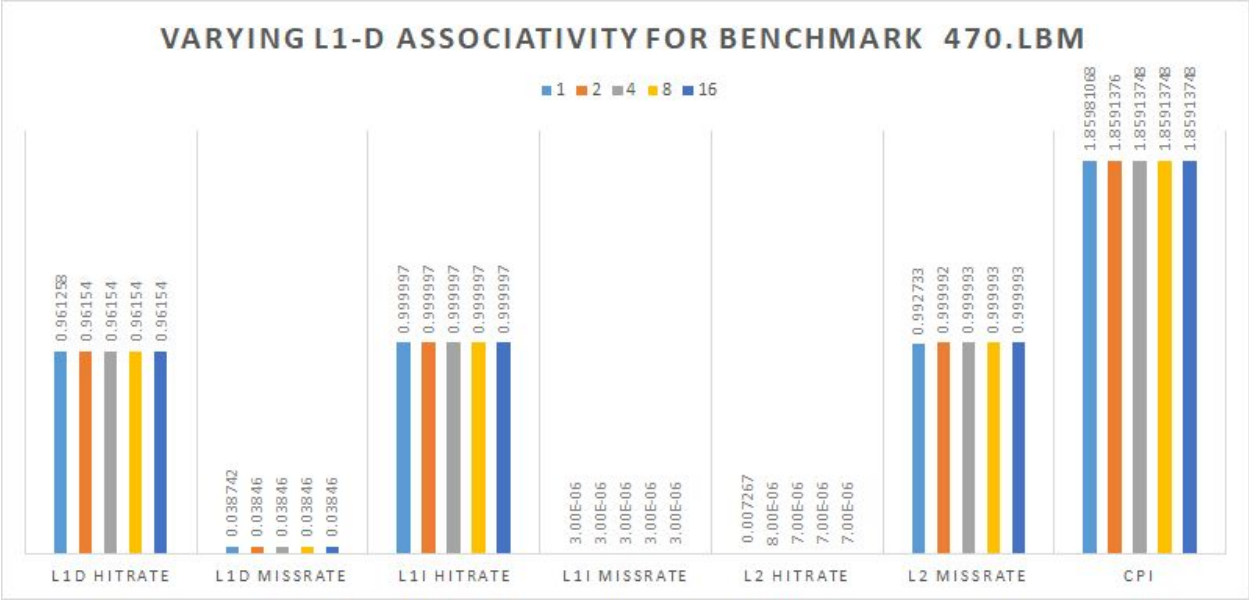
Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
256kb	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
512kb	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
1024kb	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
2048kb	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
4096kb	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914

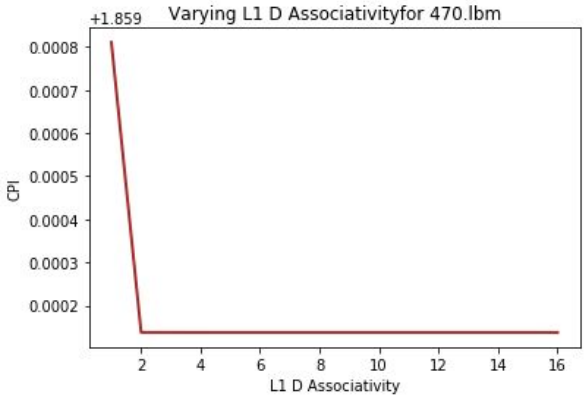


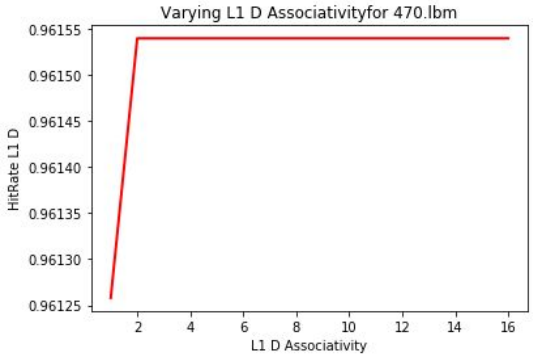
4. Varying L1-D Associativity

Varying L1 D Associative Cache Size for 470.lbm

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.961258	0.038742	0.999997	3e-06	0.007267	0.992733	1.85981
2	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
4	0.96154	0.03846	0.999997	3e-06	7e-06	0.999993	1.85914
8	0.96154	0.03846	0.999997	3e-06	7e-06	0.999993	1.85914
16	0.96154	0.03846	0.999997	3e-06	7e-06	0.999993	1.85914



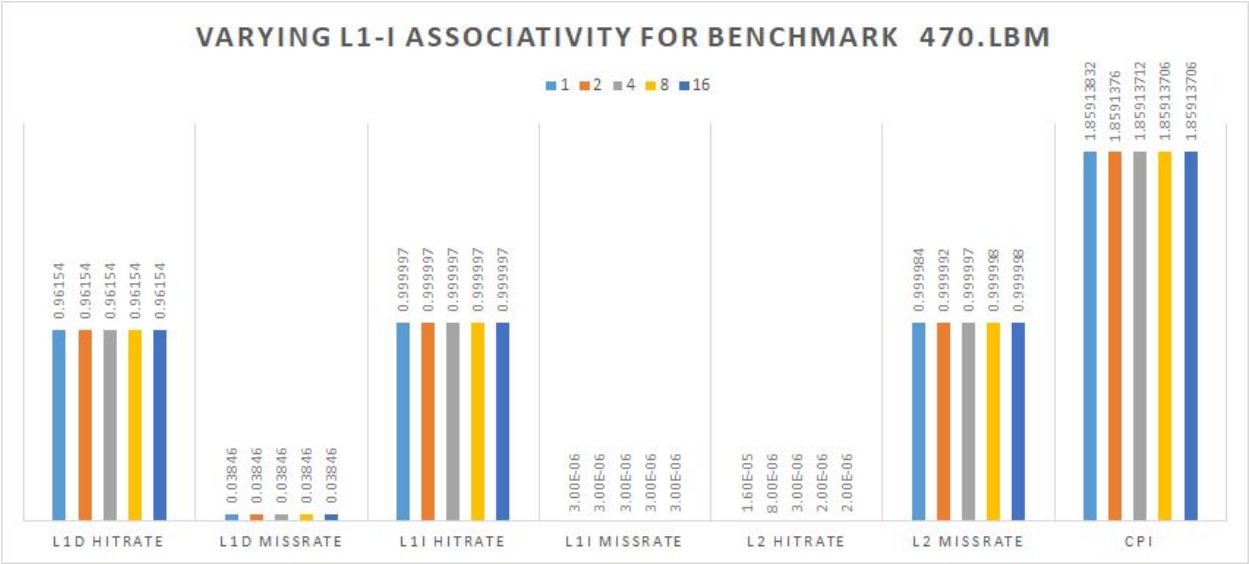




5. Varying L1-I Associativity

Varying L1 I Associative Cache Size for 470.lbm

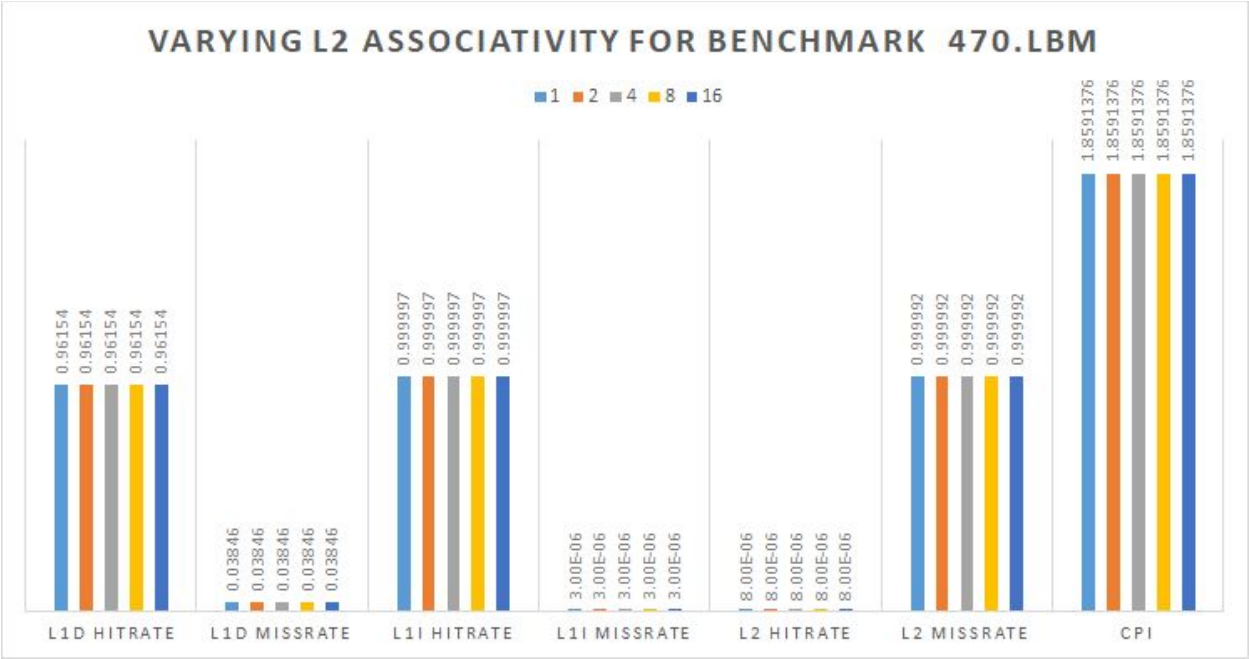
Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.96154	0.03846	0.999997	3e-06	1.6e-05	0.999984	1.85914
2	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
4	0.96154	0.03846	0.999997	3e-06	3e-06	0.999997	1.85914
8	0.96154	0.03846	0.999997	3e-06	2e-06	0.999998	1.85914
16	0.96154	0.03846	0.999997	3e-06	2e-06	0.999998	1.85914



6. Varying L2 Associativity

Varying L2 Associative Cache Size for 470.lbm

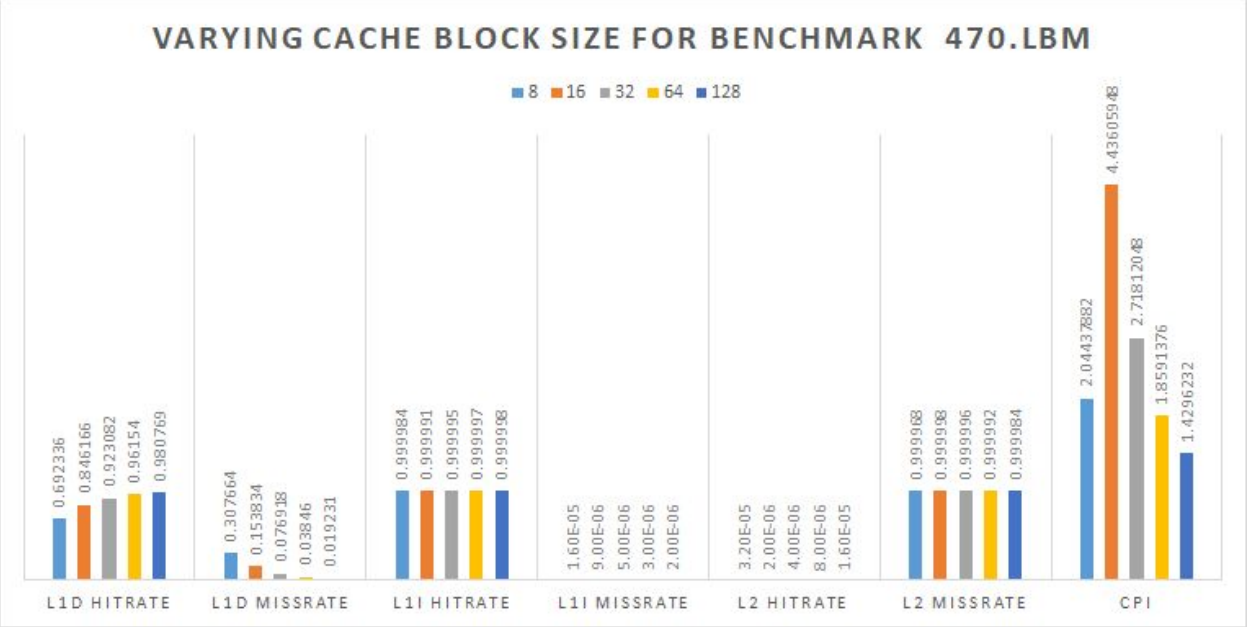
Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
1	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
2	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
4	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
8	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
16	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914

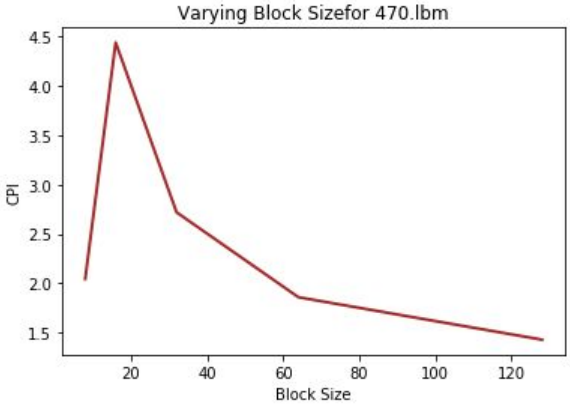


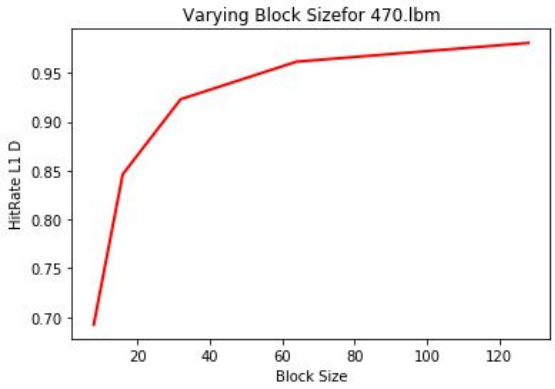
7. Varying Cache Block size

Varying Cache Block Size for 470.lbm

Size	L1D HitRate	L1D MissRate	L1I HitRate	L1I MissRate	L2 HitRate	L2 MissRate	CPI
8	0.692336	0.307664	0.999984	1.6e-05	3.2e-05	0.999968	2.04438
16	0.846166	0.153834	0.999991	9e-06	2e-06	0.999998	4.43606
32	0.923082	0.076918	0.999995	5e-06	4e-06	0.999996	2.71812
64	0.96154	0.03846	0.999997	3e-06	8e-06	0.999992	1.85914
128	0.980769	0.019231	0.999998	2e-06	1.6e-05	0.999984	1.42962







Explanation

- There is a lot of variation in the CPI here with change in block-size.
- This may be due to variation in the spatial locality of the data being loaded and the increase in block size.

- The CPI increases and then reduces due to the locality of data and in turn increases the L1 -D hit-rate.

CONCLUSION

- When the size of cache is increased, the hit-rate associated with it also increases.
- The Cycles Per Instruction(CPI) tend to decrease with increased cache sizes due to more instructions/data being loaded in a clock cycle.
- Block size affects CPI due to spatial locality of data and an increased block size increases the L1-D cache hit-rate. Too much increase in block-size however is not good and may result in an increased miss-rate.
- Increase in associativity improves the hit-rates.
- In terms of benchmarks:
    - **429.mcf** and **456.hmmer** have a low CPI of around 1 and show good consistent performance.
    - **470.lbm** has a CPI of about 1.8 which makes it intermediate in performance.
    - **401.bzip2** has a CPI of about 2.
    - **458.sjeng** has a CPI of around 5 with inconsistent performance.

INSTRUCTIONS RAN

Code for running all 175 instructions : runCALab2.sh

```
import subprocess
import os

stat_files = ["stats_l1_d_cache_", "stats_l1_i_cache_", "stats_l2_cache_", "stats_l1_d_assoc_",
"stats_l1_i_assoc_", "stats_l2_assoc_", "stats_block_size_"]
l1d_sizes = ["8kB", "16kB" ,"64kB", "128kB", "256kB"]
l1i_sizes = ["4kB", "8kB", "16kB", "64kB", "128kB"]
l2_sizes = ["256kB", "512kB", "1MB", "2MB", "4MB"]
l1d_assoc = ["1","2","4","8","16"]
l1i_assoc = ["1","2","4","8","16"]
l2_assoc = ["1","2","4","8","16"]
block_size = ["8","16","32","64","128"]
benchmarks = ["401.bzip2", "429.mcf", "456.hmmer", "458.sjeng", "470.lbm"]
final_out_dir = "/home/csgrad/hgarg/ca_lab2_out/final/"
gem5_script_file = "/util/gem5/configs/example/se.py"
out_dir = "/home/csgrad/hgarg/ca_lab2_out/instance/"
ex_com = "/util/gem5/build/X86/gem5.opt"
gem5benchdir = "/util/gem5/benchmark/"
max_instr = "100000000"

dict = {
  "401.bzip2" : ["input.program"],
```

```
  "429.mcf" : ["inp.in", "mcf.out"],
  "456.hmmer" : ["bombesin.hmm", "bombesin.hmm.new"],
  "458.sjeng" : ["test.txt"],
  "470.lbm" : ["100_100_130_cf_a.of","lbm.in"]
}

base_l1d = "128kB"
base_l1i = "64kB"
base_l2 = "1MB"
base_l1dassoc = "2"
base_l1iassoc = "2"
base_l2assoc = "1"
base_block = "64"

'''******************** RUNNING FOR L1-D CACHE FOR ALL BENCHMARKS **********************'''

#creating stats files for benchmark 401.bzip2 for L1-D cache
arg_file = gem5benchdir+benchmarks[0]+"/data/"+dict[benchmarks[0]][0]
src_file = gem5benchdir+benchmarks[0]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[0]+l1d_sizes[i]+"_"+benchmarks[0]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+l1d_sizes[i], "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 429.mcf for L1-D cache
arg_file = gem5benchdir+benchmarks[1]+"/data/"+dict[benchmarks[1]][0]
src_file = gem5benchdir+benchmarks[1]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[0]+l1d_sizes[i]+"_"+benchmarks[1]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+l1d_sizes[i], "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 456.hmmer for L1-D cache
arg_file = gem5benchdir+benchmarks[2]+"/data/"+dict[benchmarks[2]][1]
src_file = gem5benchdir+benchmarks[2]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[0]+l1d_sizes[i]+"_"+benchmarks[2]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+l1d_sizes[i], "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 458.sjeng for L1-D cache
arg_file = gem5benchdir+benchmarks[3]+"/data/"+dict[benchmarks[3]][0]
src_file = gem5benchdir+benchmarks[3]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[0]+l1d_sizes[i]+"_"+benchmarks[3]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+l1d_sizes[i], "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 470.lbm for L1-D cache
arg_file = " 20 reference.dat 0 1 "+gem5benchdir+benchmarks[4]+"/data/"+dict[benchmarks[4]][0]
src_file = gem5benchdir+benchmarks[4]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[0]+l1d_sizes[i]+"_"+benchmarks[4]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+l1d_sizes[i], "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
```

```
'''******************** RUNNING FOR L1-I CACHE FOR EACH BENCHMARK ************************'''

#creating stats files for benchmark 401.bzip2 for L1-I cache
arg_file = gem5benchdir+benchmarks[0]+"/data/"+dict[benchmarks[0]][0]
src_file = gem5benchdir+benchmarks[0]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[1]+l1i_sizes[i]+"_"+benchmarks[0]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+l1i_sizes[i], "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 429.mcf for L1-I cache
arg_file = gem5benchdir+benchmarks[1]+"/data/"+dict[benchmarks[1]][0]
src_file = gem5benchdir+benchmarks[1]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[1]+l1i_sizes[i]+"_"+benchmarks[1]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+l1i_sizes[i], "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 456.hmmer for L1-I cache
arg_file = gem5benchdir+benchmarks[2]+"/data/"+dict[benchmarks[2]][1]
src_file = gem5benchdir+benchmarks[2]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[1]+l1i_sizes[i]+"_"+benchmarks[2]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+l1i_sizes[i], "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 458.sjeng for L1-I cache
arg_file = gem5benchdir+benchmarks[3]+"/data/"+dict[benchmarks[3]][0]
src_file = gem5benchdir+benchmarks[3]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[1]+l1i_sizes[i]+"_"+benchmarks[3]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+l1i_sizes[i], "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 470.lbm for L1-I cache
arg_file = " 20 reference.dat 0 1 "+gem5benchdir+benchmarks[4]+"/data/"+dict[benchmarks[4]][0]
src_file = gem5benchdir+benchmarks[4]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[1]+l1i_sizes[i]+"_"+benchmarks[4]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+l1i_sizes[i], "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])


'''******************** RUNNING FOR L2 CACHE FOR EACH BENCHMARK ************************'''

#creating stats files for benchmark 401.bzip2 for L1-I cache
arg_file = gem5benchdir+benchmarks[0]+"/data/"+dict[benchmarks[0]][0]
src_file = gem5benchdir+benchmarks[0]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[2]+l2_sizes[i]+"_"+benchmarks[0]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+l2_sizes[i], "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
```

```
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 429.mcf for L1-I cache
arg_file = gem5benchdir+benchmarks[1]+"/data/"+dict[benchmarks[1]][0]
src_file = gem5benchdir+benchmarks[1]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[2]+l2_sizes[i]+"_"+benchmarks[1]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+l2_sizes[i], "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 456.hmmer for L1-I cache
arg_file = gem5benchdir+benchmarks[2]+"/data/"+dict[benchmarks[2]][1]
src_file = gem5benchdir+benchmarks[2]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[2]+l2_sizes[i]+"_"+benchmarks[2]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+l2_sizes[i], "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 458.sjeng for L1-I cache
arg_file = gem5benchdir+benchmarks[3]+"/data/"+dict[benchmarks[3]][0]
src_file = gem5benchdir+benchmarks[3]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[2]+l2_sizes[i]+"_"+benchmarks[3]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+l2_sizes[i], "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 470.lbm for L1-I cache
arg_file = " 20 reference.dat 0 1 "+gem5benchdir+benchmarks[4]+"/data/"+dict[benchmarks[4]][0]
src_file = gem5benchdir+benchmarks[4]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[2]+l2_sizes[i]+"_"+benchmarks[4]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+l2_sizes[i], "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])

'''******************** RUNNING FOR L1-D ASSOCIATIVITY FOR EACH BENCHMARK **********************'''

#creating stats files for benchmark 401.bzip2 for L1-D ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[0]+"/data/"+dict[benchmarks[0]][0]
src_file = gem5benchdir+benchmarks[0]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[3]+l1d_assoc[i]+"_"+benchmarks[0]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+l1d_assoc[i], "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 429.mcf for L1-D ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[1]+"/data/"+dict[benchmarks[1]][0]
src_file = gem5benchdir+benchmarks[1]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[3]+l1d_assoc[i]+"_"+benchmarks[1]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+l1d_assoc[i], "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 456.hmmer L1-D ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[2]+"/data/"+dict[benchmarks[2]][1]
```

```
src_file = gem5benchdir+benchmarks[2]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[3]+l1d_assoc[i]+"_"+benchmarks[2]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+l1d_assoc[i], "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 458.sjeng L1-D ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[3]+"/data/"+dict[benchmarks[3]][0]
src_file = gem5benchdir+benchmarks[3]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[3]+l1d_assoc[i]+"_"+benchmarks[3]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+l1d_assoc[i], "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 470.lbm for L1-D ASSOCIATIVITY
arg_file = " 20 reference.dat 0 1 "+gem5benchdir+benchmarks[4]+"/data/"+dict[benchmarks[4]][0]
src_file = gem5benchdir+benchmarks[4]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[3]+l1d_assoc[i]+"_"+benchmarks[4]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+l1d_assoc[i], "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])


'''******************** RUNNING FOR L1-I ASSOCIATIVITY FOR EACH BENCHMARK **********************'''

#creating stats files for benchmark 401.bzip2 for L1-I ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[0]+"/data/"+dict[benchmarks[0]][0]
src_file = gem5benchdir+benchmarks[0]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[4]+l1i_assoc[i]+"_"+benchmarks[0]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+l1i_assoc[i], "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 429.mcf for L1-I ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[1]+"/data/"+dict[benchmarks[1]][0]
src_file = gem5benchdir+benchmarks[1]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[4]+l1i_assoc[i]+"_"+benchmarks[1]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+l1i_assoc[i], "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 456.hmmer L1-I ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[2]+"/data/"+dict[benchmarks[2]][1]
src_file = gem5benchdir+benchmarks[2]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[4]+l1i_assoc[i]+"_"+benchmarks[2]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+l1i_assoc[i], "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 458.sjeng L1-I ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[3]+"/data/"+dict[benchmarks[3]][0]
src_file = gem5benchdir+benchmarks[3]+"/src/benchmark"
for i in range(5):
```

```
  final_file = final_out_dir+stat_files[4]+l1i_assoc[i]+"_"+benchmarks[3]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+l1i_assoc[i], "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 470.lbm for L1-I ASSOCIATIVITY
arg_file = " 20 reference.dat 0 1 "+gem5benchdir+benchmarks[4]+"/data/"+dict[benchmarks[4]][0]
src_file = gem5benchdir+benchmarks[4]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[4]+l1i_assoc[i]+"_"+benchmarks[4]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+l1i_assoc[i], "--l2_assoc="+base_l2assoc,
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])


'''******************** RUNNING FOR L2 ASSOCIATIVITY FOR EACH BENCHMARK ************************'''

#creating stats files for benchmark 401.bzip2 for L2 ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[0]+"/data/"+dict[benchmarks[0]][0]
src_file = gem5benchdir+benchmarks[0]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[5]+l2_assoc[i]+"_"+benchmarks[0]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+l2_assoc[i],
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 429.mcf for L2 ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[1]+"/data/"+dict[benchmarks[1]][0]
src_file = gem5benchdir+benchmarks[1]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[5]+l2_assoc[i]+"_"+benchmarks[1]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+l2_assoc[i],
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 456.hmmer L2 ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[2]+"/data/"+dict[benchmarks[2]][1]
src_file = gem5benchdir+benchmarks[2]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[5]+l2_assoc[i]+"_"+benchmarks[2]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+l2_assoc[i],
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 458.sjeng L2 ASSOCIATIVITY
arg_file = gem5benchdir+benchmarks[3]+"/data/"+dict[benchmarks[3]][0]
src_file = gem5benchdir+benchmarks[3]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[5]+l2_assoc[i]+"_"+benchmarks[3]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+l2_assoc[i],
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 470.lbm for L2 ASSOCIATIVITY
arg_file = " 20 reference.dat 0 1 "+gem5benchdir+benchmarks[4]+"/data/"+dict[benchmarks[4]][0]
src_file = gem5benchdir+benchmarks[4]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[5]+l2_assoc[i]+"_"+benchmarks[4]+".txt"
```

```
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+l2_assoc[i],
"--cacheline_size="+base_block])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])

'''******************** RUNNING FOR BLOCK SIZE FOR EACH BENCHMARK ***********************'''
#creating stats files for benchmark 401.bzip2 for BLOCK SIZE
arg_file = gem5benchdir+benchmarks[0]+"/data/"+dict[benchmarks[0]][0]
src_file = gem5benchdir+benchmarks[0]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[6]+block_size[i]+"_"+benchmarks[0]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+block_size[i]])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 429.mcf for BLOCK SIZE
arg_file = gem5benchdir+benchmarks[1]+"/data/"+dict[benchmarks[1]][0]
src_file = gem5benchdir+benchmarks[1]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[6]+block_size[i]+"_"+benchmarks[1]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+block_size[i]])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 456.hmmer BLOCK SIZE
arg_file = gem5benchdir+benchmarks[2]+"/data/"+dict[benchmarks[2]][1]
src_file = gem5benchdir+benchmarks[2]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[6]+block_size[i]+"_"+benchmarks[2]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+block_size[i]])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 458.sjeng BLOCK SIZE
arg_file = gem5benchdir+benchmarks[3]+"/data/"+dict[benchmarks[3]][0]
src_file = gem5benchdir+benchmarks[3]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[6]+block_size[i]+"_"+benchmarks[3]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+block_size[i]])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
#creating stats files for benchmark 470.lbm for BLOCK SIZE
arg_file = " 20 reference.dat 0 1 "+gem5benchdir+benchmarks[4]+"/data/"+dict[benchmarks[4]][0]
src_file = gem5benchdir+benchmarks[4]+"/src/benchmark"
for i in range(5):
  final_file = final_out_dir+stat_files[6]+block_size[i]+"_"+benchmarks[4]+".txt"
  subprocess.call(["time", ex_com, "-d",out_dir,gem5_script_file,"-c",src_file,"-o",arg_file,"-I", max_instr,
"--l1d_size="+base_l1d, "--l1i_size="+base_l1i, "--l2_size="+base_l2, "--caches",
"--l2cache","--l1d_assoc="+base_l1dassoc, "--l1i_assoc="+base_l1iassoc, "--l2_assoc="+base_l2assoc,
"--cacheline_size="+block_size[i]])
  subprocess.call(["mv", out_dir+"stats.txt", final_file])
```

Code for creating Graphs and Tables

```
import re
import sys
```

```
import os
all_files = os.listdir("stats/")
import matplotlib.pyplot as plt
from tabulate import tabulate
'''
system.cpu.dcache.overall_miss_rate::total
system.cpu.icache.overall_miss_rate::total
system.l2.overall_miss_rate::total
system.cpu.dcache.overall_misses::total
system.cpu.icache.overall_misses::total
system.cpu.dcache.overall_hits::total
system.cpu.icache.overall_hits::total
system.l2.overall_hits::total
system.l2.overall_misses::total
'''
import csv
def tables(vary,a,b,c,d,e,f,g,title = '',postfix = '',bench = '',csvname = 'default'):
   print('\n')
   print('                                    Varying ' + str(title)+ ' for '+ str(bench))
   row_form = [[0 for _ in range(8)] for _ in range(6)]
   headers = ['','L1D HitRate','L1D MissRate','L1I HitRate','L1I MissRate','L2 HitRate','L2 MissRate','CPI']
   for i in range(len(headers)):
       row_form[0][i] = headers[i]
   for i in range(5):
       row_form[i+1][0] = str(vary[i]) + postfix
       row_form[i+1][1] = a[i]
       row_form[i+1][2] = b[i]
       row_form[i+1][3] = c[i]
       row_form[i+1][4] = d[i]
       row_form[i+1][5] = e[i]
       row_form[i+1][6] = f[i]
       row_form[i+1][7] = g[i]
   with open("charts/"+str(csvname)+".csv","w+",newline = '') as my_csv:
       csvWriter = csv.writer(my_csv,delimiter=',')
       csvWriter.writerows(row_form)
   print(tabulate(row_form[1:],headers = ['Size','L1D HitRate','L1D MissRate','L1I HitRate','L1I MissRate','L2
HitRate','L2 MissRate','CPI'], tablefmt='fancy_grid'))

ins_num = 100000000
l1d = [8,16,64,128,256]
l1i = [4, 8, 16, 64, 128]
l2 = [256, 512, 1, 2, 4]
l2_graph = [256,512,1024,2048,4096]
l1d_assoc = [1,2,4,8,16]
l1i_assoc = [1,2,4,8,16]
l2_assoc = [1,2,4,8,16]
block_size = [8,16,32,64,128]
benchmarks = ["401.bzip2", "429.mcf", "456.hmmer", "458.sjeng", "470.lbm"]
stat_files = ["stats_l1_d_cache_", "stats_l1_i_cache_", "stats_l2_cache_", "stats_l1_d_assoc_",
"stats_l1_i_assoc_", "stats_l2_assoc_", "stats_block_size_"]

hits_l1d = []
misses_l1d = []
hits_l1i = []
misses_l1i = []
hits_l2 = []
misses_l2 = []
cpi_a = []

#Change this
bench = benchmarks[4]
varier = stat_files[6]
iterator = block_size
iterator_graph = block_size
iterator_label = 'block'
chart_labelsss = 'Cache Block Size'
```

```
inter = "_"
vary_postfix = ''
for indexf,filename in enumerate(iterator):
   #inter = "kB_"
   #if indexf > 1:
   #    inter = "MB_"
   file_to_open = 'stats/'+varier+ str(filename) + inter + str(bench) +'.txt'
   print(file_to_open)
   #if file_to_open not in all_files:
   #file_to_open = 'stats/'+varier+ str(filename) + 'MB_' + str(bench) +'.txt'

   with open(file_to_open ,'r') as textfile:
       all_lines = textfile.readlines()
       for i in range(len(all_lines)):
           all_lines[i] = all_lines[i].split()
           if len(all_lines[i]):
               if all_lines[i][0] == "system.cpu.dcache.overall_miss_rate::total":
                   l1d_miss = float(all_lines[i][1])
                   l1d_hit = 1 - l1d_miss
                   #print('Miss Rate L1 D' +' : '+ str(l1d_miss))
                    #print('\nHit Rate L1 D' +' : '+ str(l1d_hit))
                    hits_l1d.append(l1d_hit)
                   misses_l1d.append(l1d_miss)
               elif all_lines[i][0].rstrip() == "system.cpu.icache.overall_miss_rate::total":
                   l1i_miss = float(all_lines[i][1])
                   l1i_hit = 1 - l1i_miss
                   #print('Miss Rate L1 I' +' : '+ str(l1i_miss))
                    #print('\nHit Rate L1 I' +' : '+ str(l1i_hit))
                    hits_l1i.append(l1i_hit)
                   misses_l1i.append(l1i_miss)
               elif all_lines[i][0] == "system.l2.overall_miss_rate::total":
                   l2_miss = float(all_lines[i][1])
                   l2_hit = 1 - l2_miss
                   #print('Miss Rate L2' +' : '+ str(l2_miss))
                    #print('\nHit Rate L2' +' : '+ str(l2_hit))
                    hits_l2.append(l2_hit)
                   misses_l2.append(l2_miss)
               elif all_lines[i][0] == "system.cpu.dcache.overall_misses::total":
                   l1d_m = float(all_lines[i][1])
               elif all_lines[i][0] == "system.cpu.icache.overall_misses::total":
                   l1i_m = float(all_lines[i][1])
               elif all_lines[i][0] == "system.l2.overall_misses::total":
                   l2_m = float(all_lines[i][1])


       cpi = 1 + ((((l1i_m + l1d_m)*6) + (l2_m*50))/ ins_num)
       #print('CPI : ' + str(cpi))
       cpi_a.append(cpi)
       #print("\n\n\n##################################################")

#print(l1d)
#print(hits_l1d)
#print(misses_l1d)
#print(hits_l1i)
#print(misses_l1i)
#print(hits_l2)
#print(misses_l2)
#print(cpi_a)



chrtdir = "chrtimages/"
#Change this
x_axis = iterator
```

```
x_labels = {"l1d":"L1 D Cache Size (kB)", 'l1i':'L1 I Cache Size (kB)', 'l2':'L2 Cache Size (kB)', 'l1d_a':'L1 D
Associativity', 'l1i_a':'L1 I Associativity','l2_a':'L2 Associativity','block':'Block Size'}

#Change this, possible values l1d, l1i,l2,l1d_a,l1i_a,l2_a,block
cur = iterator_label

#benchmarks = ["401.bzip2", "429.mcf", "456.hmmer", "458.sjeng", "470.lbm"]

#Change this
cur_bench = bench

#plt.subplot(2,2,1)
plt.plot(x_axis,hits_l1d,color = 'red', linewidth = 2.0)
plt.xlabel(x_labels[cur])
plt.ylabel('HitRate L1 D')
plt.title('Varying '+str(x_labels[cur])+'for '+str(cur_bench))
plt.savefig(chrtdir+"HitRate L1 D")
plt.show()


#plt.subplot(2,2,1)
plt.plot(x_axis,misses_l1d, color = 'green', linewidth = 2.0)
plt.xlabel(x_labels[cur])
plt.ylabel('MissRate L1 D')
plt.title('Varying '+str(x_labels[cur])+'for '+str(cur_bench))
plt.savefig(chrtdir+"HitRateL1D.png")
plt.show()

#plt.subplot(2,2,1)
plt.plot(x_axis,hits_l1i, color = 'blue', linewidth = 2.0)
plt.xlabel(x_labels[cur])
plt.ylabel('HitRate L1 I')
plt.title('Varying '+str(x_labels[cur])+'for '+str(cur_bench))
plt.savefig(chrtdir+"HitRateL1I.png")
plt.show()

#plt.subplot(2,2,1)
plt.plot(x_axis,misses_l1i, color = 'yellow', linewidth = 2.0)
plt.xlabel(x_labels[cur])
plt.ylabel('MissRate L1 I')
plt.title('Varying '+str(x_labels[cur])+'for '+str(cur_bench))
plt.savefig(chrtdir+"MissRateL1I.png")
plt.show()

plt.plot(x_axis,hits_l2, color = 'black', linewidth = 2.0)
plt.xlabel(x_labels[cur])
plt.ylabel('HitRate L2')
plt.title('Varying '+str(x_labels[cur])+'for '+str(cur_bench))
plt.savefig(chrtdir+"HitRateL2.png")
plt.show()

plt.plot(x_axis,misses_l2, color = 'orange', linewidth = 2.0)
plt.xlabel(x_labels[cur])
plt.ylabel('MissRate L2')
plt.title('Varying '+str(x_labels[cur])+'for '+str(cur_bench))
plt.savefig(chrtdir+"MissRateL2.png")
plt.show()

plt.plot(x_axis,cpi_a, color = 'brown', linewidth = 2.0)
plt.xlabel(x_labels[cur])
plt.ylabel('CPI')
plt.title('Varying '+str(x_labels[cur])+'for '+str(cur_bench))
plt.savefig(chrtdir+"CPI.png")
plt.show()
```

```
tables(iterator_graph, hits_l1d,
misses_l1d,hits_l1i,misses_l1i,hits_l2,misses_l2,cpi_a,chart_labelsss,vary_postfix,bench=bench,csvname =
iterator_label+bench)
```