ENEE 646: Digital Computer Design
Program Assignment #3 Report

Yang Xie 113085185

# 2.1 Working Set Characteriztion

Cache configuration:

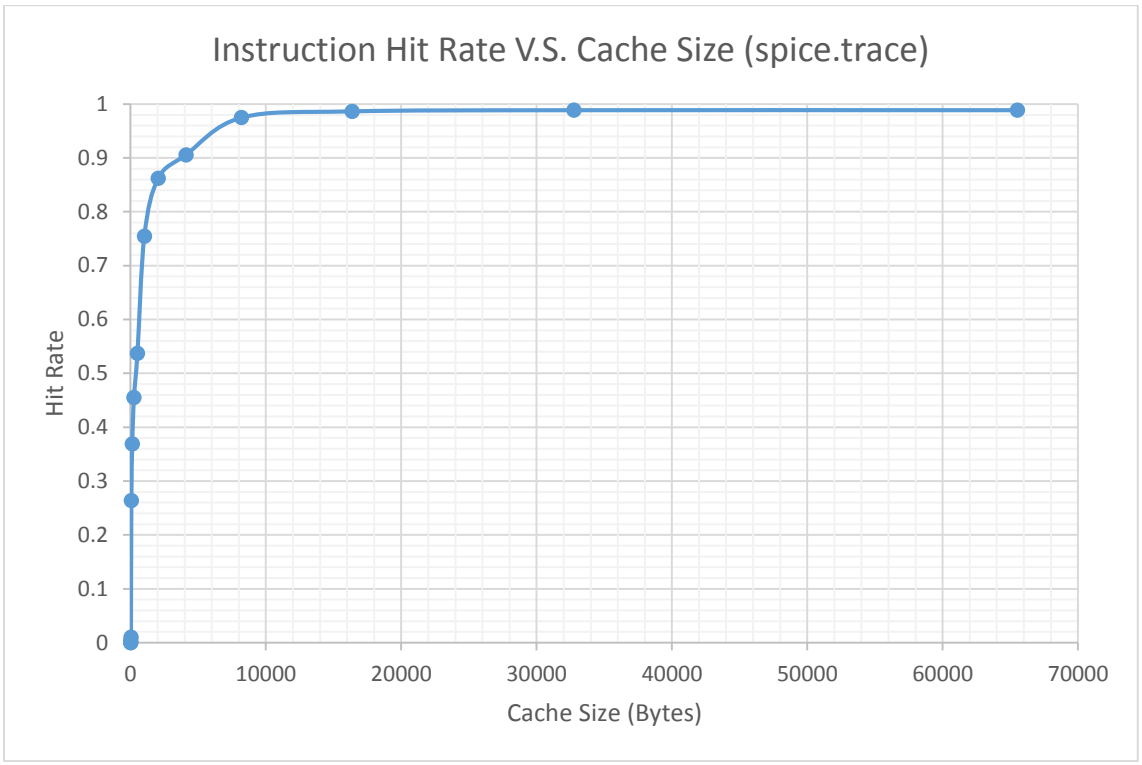
- Split, fully associative, block size 4 bytes, write-back, write-allocate
- Cache size increased from 4 bytes until the hit rate remains insensitive to cache size

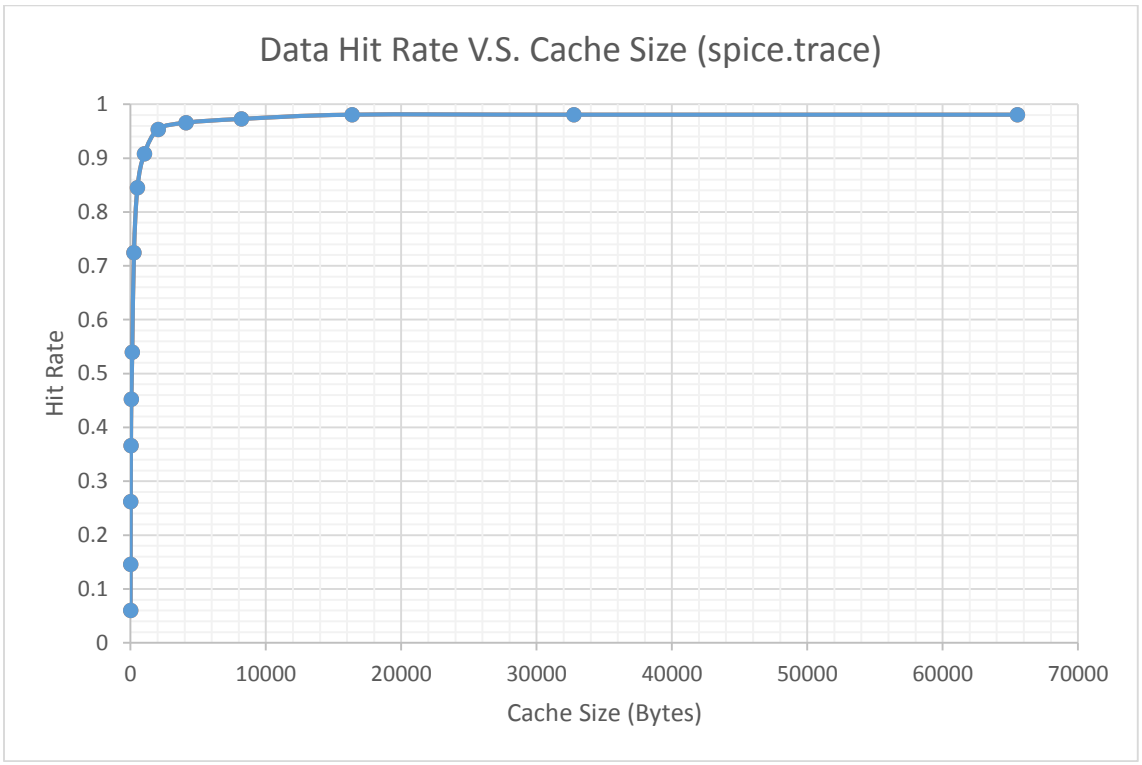
2.1.1 Data and plot for spice.trace
Data:

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 4 | 4 | 1 | 782764 | 782764 | 0 | 217237 | 204118 | 0.060390265 |
| 8 | 4 | 2 | 782764 | 782764 | 0 | 217237 | 185640 | 0.14544944 |
| 16 | 4 | 4 | 782764 | 780135 | 0.003358611 | 217237 | 160277 | 0.262202111 |
| 32 | 4 | 8 | 782764 | 774879 | 0.010073279 | 217237 | 137747 | 0.365913726 |
| 64 | 4 | 16 | 782764 | 576445 | 0.263577528 | 217237 | 119032 | 0.452063875 |
| 128 | 4 | 32 | 782764 | 494190 | 0.368660286 | 217237 | 100030 | 0.539535162 |
| 256 | 4 | 64 | 782764 | 426746 | 0.454821632 | 217237 | 59898 | 0.724273489 |
| 512 | 4 | 128 | 782764 | 362575 | 0.536801641 | 217237 | 33711 | 0.844819253 |
| 1024 | 4 | 256 | 782764 | 191922 | 0.754814989 | 217237 | 19993 | 0.907966875 |
| 2048 | 4 | 512 | 782764 | 107987 | 0.862043988 | 217237 | 10170 | 0.953184771 |
| 4096 | 4 | 1024 | 782764 | 73811 | 0.905704657 | 217237 | 7475 | 0.965590576 |
| 8192 | 4 | 2048 | 782764 | 19782 | 0.974728015 | 217237 | 5950 | 0.972610559 |
| 16384 | 4 | 4096 | 782764 | 10620 | 0.986432692 | 217237 | 4225 | 0.980551195 |
| 32768 | 4 | 8192 | 782764 | 8964 | 0.988548273 | 217237 | 4225 | 0.980551195 |
| 65536 | 4 | 16384 | 782764 | 8964 | 0.988548273 | 217237 | 4225 | 0.980551195 |

The hit rate for instruction remains insensitive when cache size reaches beyond 32KB, and the hit rate for data remains insensitive when cache size reaches beyond 16KB.

Plot:


Instruction Hit Rate V.S. Cache Size (spice.trace)
Hit Rate
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10000
20000
30000
40000
50000
60000
70000
Cache Size (Bytes)


Data Hit Rate V.S. Cache Size (spice.trace)
Hit Rate
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10000
20000
30000
40000
50000
60000
70000
Cache Size (Bytes)

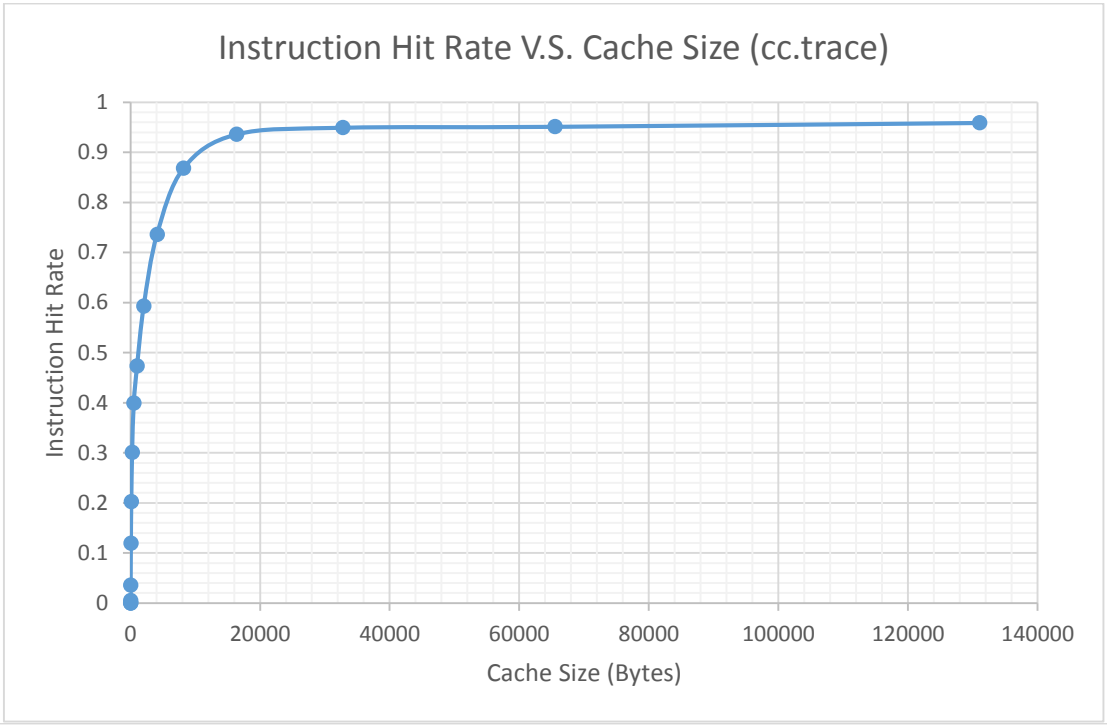
2.1.2 Data and plot for cc.trace

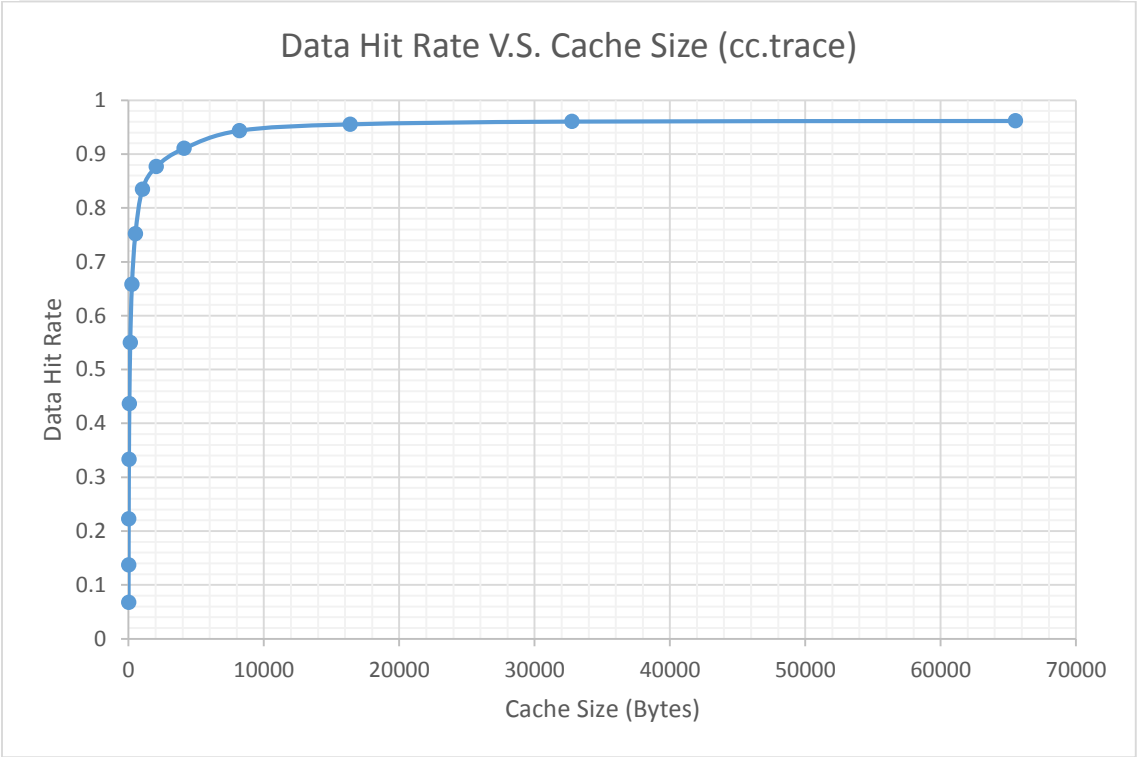
Data:

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 4 | 4 | 1 | 757341 | 757341 | 0 | 242661 | 226269 | 0.067551 |
| 8 | 4 | 2 | 757341 | 757341 | 0 | 242661 | 209336 | 0.137332 |
| 16 | 4 | 4 | 757341 | 753804 | 0.00467 | 242661 | 188581 | 0.222862 |
| 32 | 4 | 8 | 757341 | 730580 | 0.035335 | 242661 | 161866 | 0.332954 |
| 64 | 4 | 16 | 757341 | 666884 | 0.11944 | 242661 | 136720 | 0.43658 |
| 128 | 4 | 32 | 757341 | 604143 | 0.202284 | 242661 | 109141 | 0.550233 |
| 256 | 4 | 64 | 757341 | 529362 | 0.301026 | 242661 | 82925 | 0.658268 |
| 512 | 4 | 128 | 757341 | 454953 | 0.399276 | 242661 | 60111 | 0.752284 |
| 1024 | 4 | 256 | 757341 | 398640 | 0.473632 | 242661 | 40189 | 0.834382 |
| 2048 | 4 | 512 | 757341 | 308232 | 0.593008 | 242661 | 29941 | 0.876614 |
| 4096 | 4 | 1024 | 757341 | 199920 | 0.736024 | 242661 | 21715 | 0.910513 |
| 8192 | 4 | 2048 | 757341 | 99691 | 0.868367 | 242661 | 13670 | 0.943666 |
| 16384 | 4 | 4096 | 757341 | 48500 | 0.93596 | 242661 | 10797 | 0.955506 |
| 32768 | 4 | 8192 | 757341 | 38475 | 0.949197 | 242661 | 9554 | 0.960628 |
| 65536 | 4 | 16384 | 757341 | 37005 | 0.951138 | 242661 | 9273 | 0.961786 |
| 131072 | 4 | 32768 | 757341 | 31195 | 0.95881 | 242661 | 9273 | 0.961786 |

The hit rate for instruction remains insensitive when cache size reaches beyond 128KB, and the hit rate for data remains insensitive when cache size reaches beyond 64KB.

Plot:


Instruction Hit Rate V.S. Cache Size (cc.trace)
Instruction Hit Rate
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
20000
40000
60000
80000
100000
120000
140000
Cache Size (Bytes)


Data Hit Rate V.S. Cache Size (cc.trace)
Data Hit Rate
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10000
20000
30000
40000
50000
60000
70000
Cache Size (Bytes)

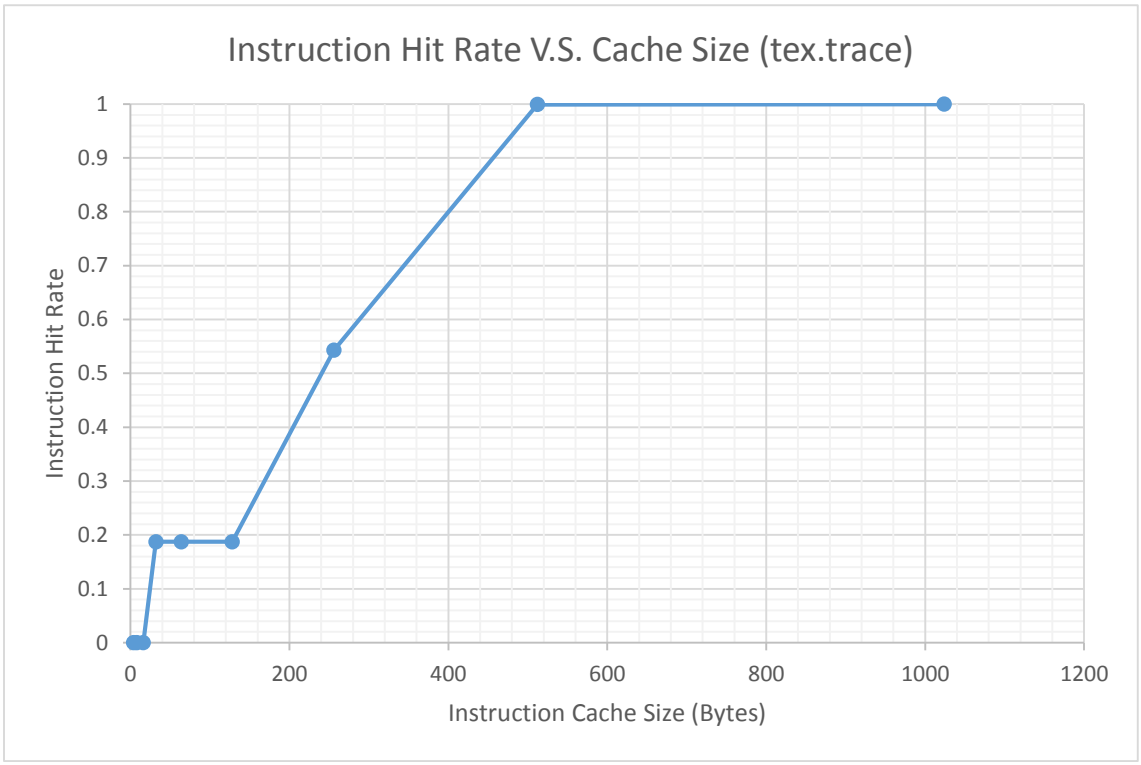
2.1.3 Data and plot for tex.trace

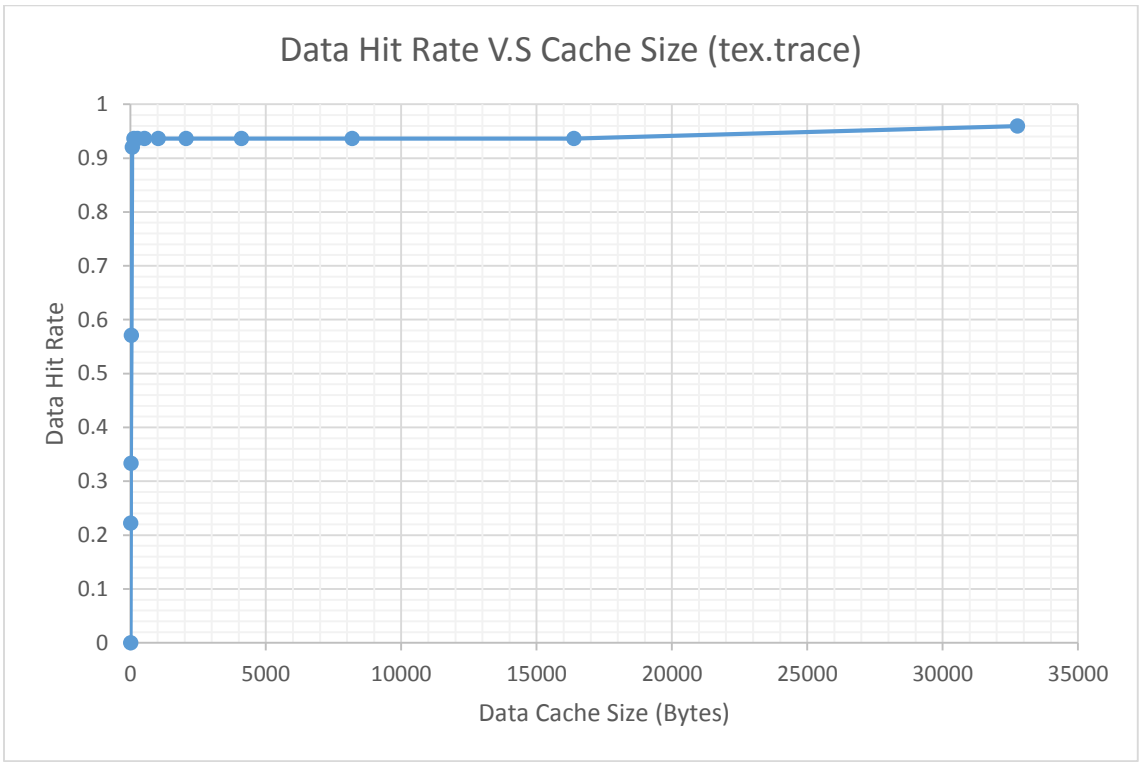
Data

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 4 | 4 | 1 | 597309 | 597309 | 0 | 235168 | 235166 | 8.5E-06 |
| 8 | 4 | 2 | 597309 | 597309 | 0 | 235168 | 182914 | 0.222199 |
| 16 | 4 | 4 | 597309 | 597309 | 0 | 235168 | 156786 | 0.333302 |
| 32 | 4 | 8 | 597309 | 485394 | 0.187365 | 235168 | 100815 | 0.571306 |
| 64 | 4 | 16 | 597309 | 485394 | 0.187365 | 235168 | 18714 | 0.920423 |
| 128 | 4 | 32 | 597309 | 485394 | 0.187365 | 235168 | 14944 | 0.936454 |
| 256 | 4 | 64 | 597309 | 272807 | 0.543273 | 235168 | 14944 | 0.936454 |
| 512 | 4 | 128 | 597309 | 530 | 0.999113 | 235168 | 14944 | 0.936454 |
| 1024 | 4 | 256 | 597309 | 160 | 0.999732 | 235168 | 14944 | 0.936454 |
| 2048 | 4 | 512 | 597309 | 160 | 0.999732 | 235168 | 14944 | 0.936454 |
| 4096 | 4 | 1024 | 597309 | 160 | 0.999732 | 235168 | 14944 | 0.936454 |
| 8192 | 4 | 2048 | 597309 | 160 | 0.999732 | 235168 | 14944 | 0.936454 |
| 16384 | 4 | 4096 | 597309 | 160 | 0.999732 | 235168 | 14944 | 0.936454 |
| 32768 | 4 | 8192 | 597309 | 160 | 0.999732 | 235168 | 9522 | 0.95951 |
| 65536 | 4 | 16384 | 597309 | 160 | 0.999732 | 235168 | 9522 | 0.95951 |

The hit rate for instruction remains insensitive when cache size reaches beyond 1KB, and the hit rate for data remains insensitive when cache size reaches beyond 32KB.

Plot:


Instruction Hit Rate V.S. Cache Size (tex.trace)
Instruction Hit Rate
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
200
400
600
800
1000
1200
Instruction Cache Size (Bytes)


Data Hit Rate V.S Cache Size (tex.trace)
Data Hit Rate
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5000
10000
15000
20000
25000
30000
35000
Data Cache Size (Bytes)

2.1.4 Analysis

a. Explain what this experiment is doing, and how it works. Also, explain the significance of the features in the hit-rate vs. cache size plots.

This experiment is analyzing the relationship between cache size and hit rate. It fixes all other configurations and only change the cache size so that we can find out the relationship between hit rate and cache size.

Comparing three data sets and their plots, we can conclude that:

- When the cache size increases from 4 bytes to several kilobytes, the capacity miss will be decreased and the cache hit rate increases dramatically until certain point.
- After that, the cache hit rate will increase slowly until it reaches an upper bound, which means that increasing cache size will no longer increase the cache hit rate. This happens because increasing cache size can't avoid the compulsory misses.

b. What is the total instruction working set size and data working set size for each of the three sample traces?

spice.trace: Instruction working set size: 8964; Data working set size: 4225
cc.trace: Instruction working set size: 31195; Data working set size: 9273
tex.trace: Instruction working set size: 160; Data working set size: 9522

## 2.2 Impact of Block Size

Cache configuration:

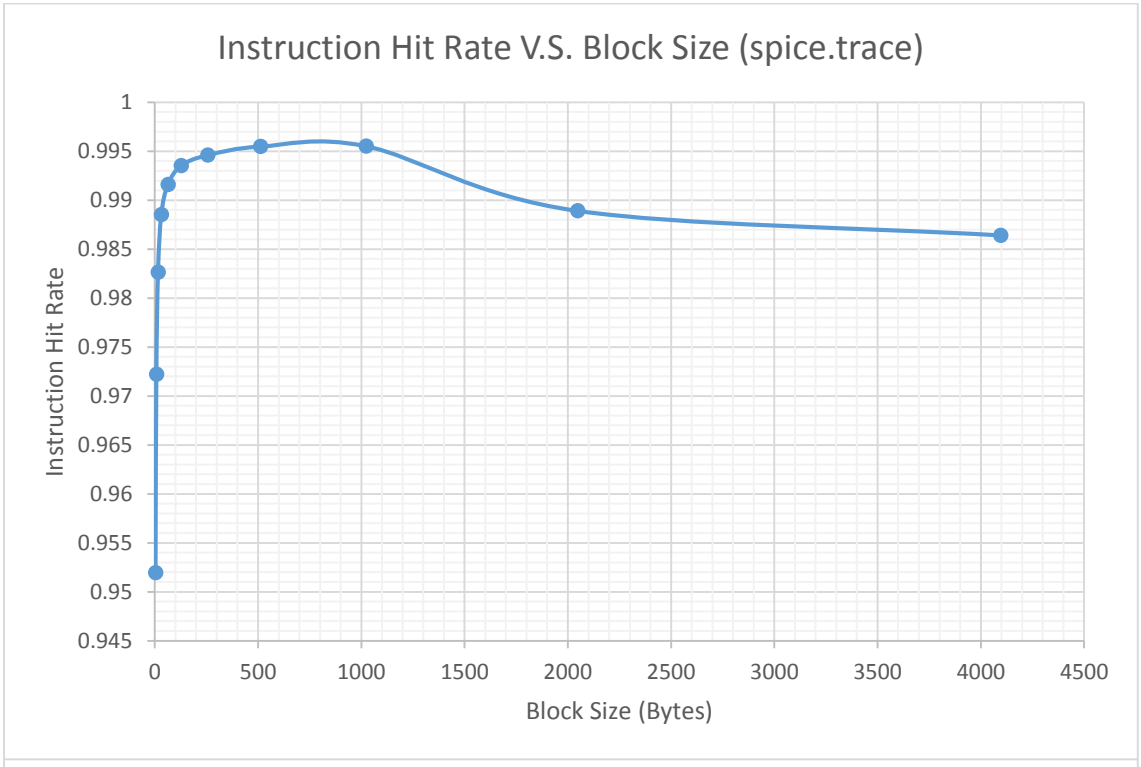
- Split, 2 way associative, cache size 8K bytes, write-back, write-allocate
- Block size from 4 bytes to 4KB

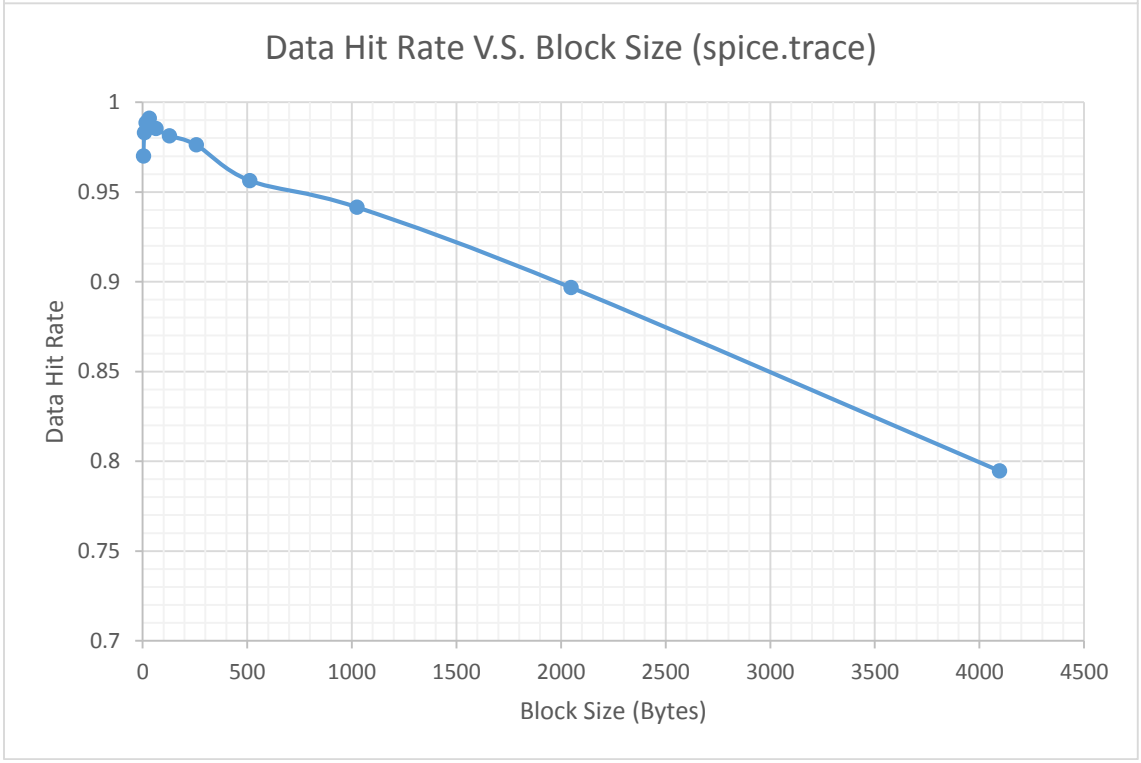
2.2.1 Data and Plot for spice.trace

Data:

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 8192 | 4 | 2 | 782764 | 37618 | 0.951942 | 217237 | 6491 | 0.97012 |
| 8192 | 8 | 2 | 782764 | 21743 | 0.972223 | 217237 | 3683 | 0.983046 |
| 8192 | 16 | 2 | 782764 | 13585 | 0.982645 | 217237 | 2467 | 0.988644 |
| 8192 | 32 | 2 | 782764 | 8976 | 0.988533 | 217237 | 1913 | 0.991194 |
| 8192 | 64 | 2 | 782764 | 6590 | 0.991581 | 217237 | 3160 | 0.985454 |
| 8192 | 128 | 2 | 782764 | 5073 | 0.993519 | 217237 | 4039 | 0.981407 |
| 8192 | 256 | 2 | 782764 | 4230 | 0.994596 | 217237 | 5157 | 0.976261 |
| 8192 | 512 | 2 | 782764 | 3543 | 0.995474 | 217237 | 9479 | 0.956366 |
| 8192 | 1024 | 2 | 782764 | 3516 | 0.995508 | 217237 | 12701 | 0.941534 |
| 8192 | 2048 | 2 | 782764 | 8681 | 0.98891 | 217237 | 22415 | 0.896818 |
| 8192 | 4096 | 2 | 782764 | 10638 | 0.98641 | 217237 | 44607 | 0.794662 |

Plot:


Instruction Hit Rate V.S. Block Size (spice.trace)
Instruction Hit Rate
1
0.995
0.99
0.985
0.98
0.975
0.97
0.965
0.96
0.955
0.95
0.945
0
500
1000
1500
2000
2500
3000
3500
4000
4500
Block Size (Bytes)

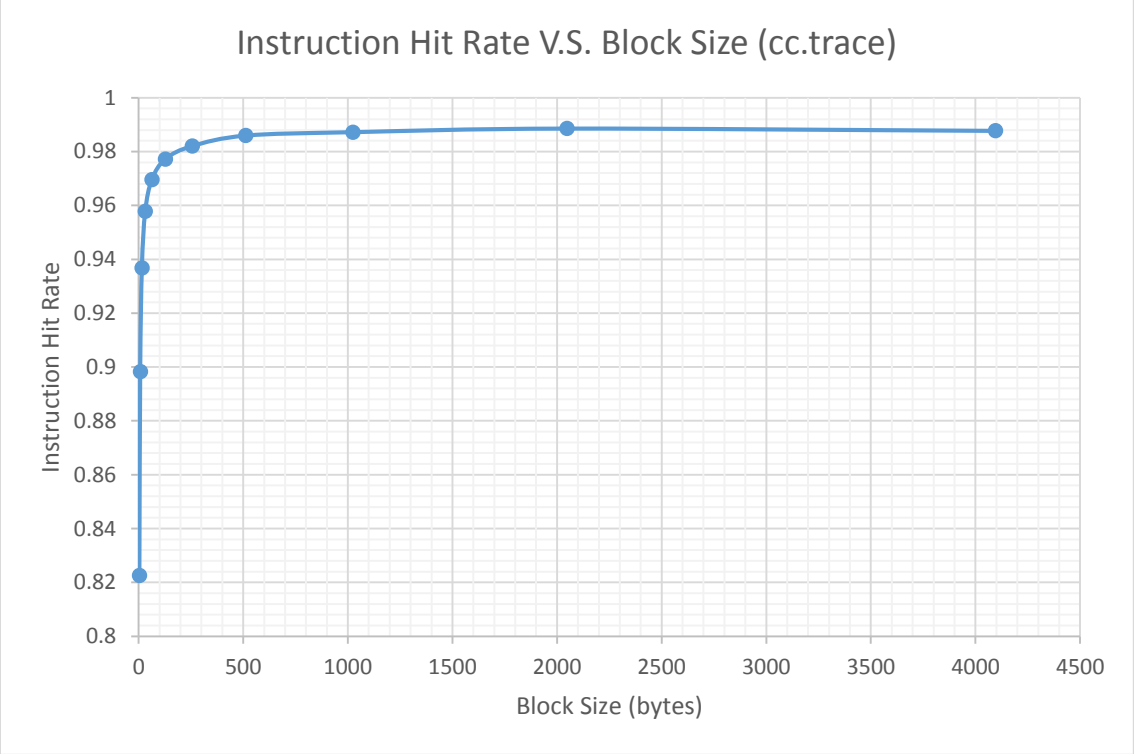

Data Hit Rate V.S. Block Size (spice.trace)
Data Hit Rate
1
0.95
0.9
0.85
0.8
0.75
0.7
0
500
1000
1500
2000
2500
3000
3500
4000
4500
Block Size (Bytes)

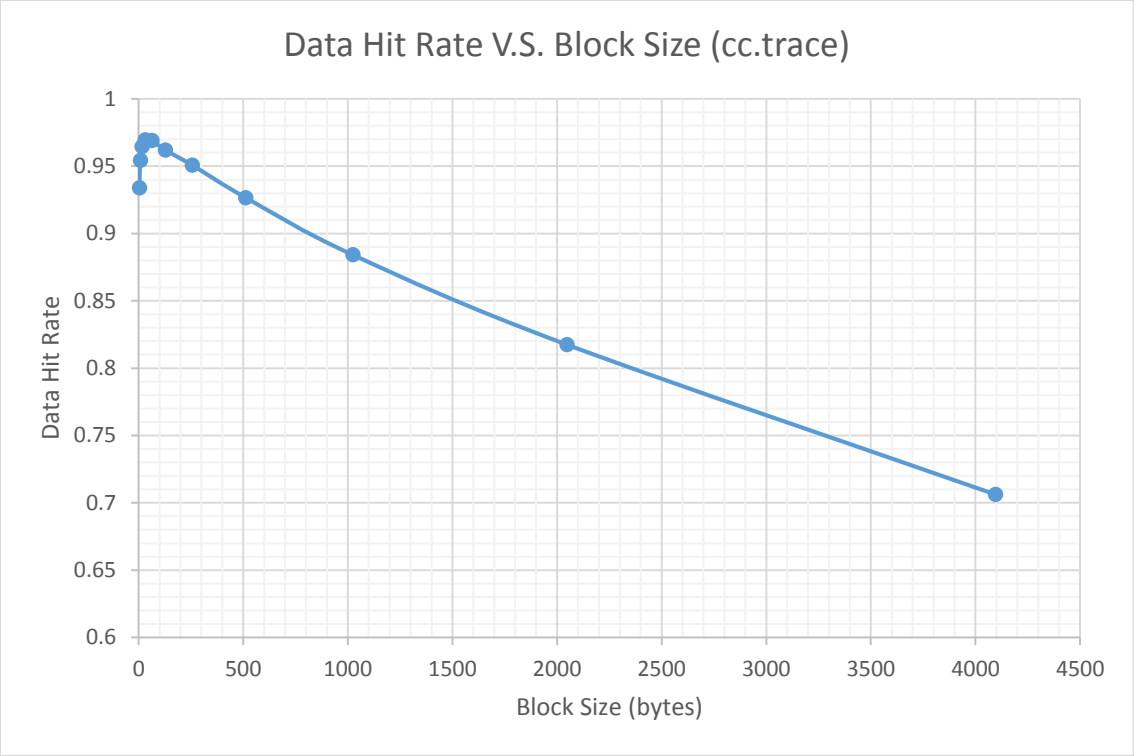
2.2.2 Data and Plot for cc.trace

Data:

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 8192 | 4 | 2 | 757341 | 134420 | 0.822511 | 242661 | 16045 | 0.933879 |
| 8192 | 8 | 2 | 757341 | 77069 | 0.898237 | 242661 | 11059 | 0.954426 |
| 8192 | 16 | 2 | 757341 | 47894 | 0.93676 | 242661 | 8582 | 0.964634 |
| 8192 | 32 | 2 | 757341 | 31983 | 0.957769 | 242661 | 7397 | 0.969517 |
| 8192 | 64 | 2 | 757341 | 23040 | 0.969578 | 242661 | 7521 | 0.969006 |
| 8192 | 128 | 2 | 757341 | 17301 | 0.977156 | 242661 | 9225 | 0.961984 |
| 8192 | 256 | 2 | 757341 | 13624 | 0.982011 | 242661 | 11937 | 0.950808 |
| 8192 | 512 | 2 | 757341 | 10679 | 0.985899 | 242661 | 17793 | 0.926675 |
| 8192 | 1024 | 2 | 757341 | 9713 | 0.987175 | 242661 | 28109 | 0.884164 |
| 8192 | 2048 | 2 | 757341 | 8709 | 0.988501 | 242661 | 44295 | 0.817461 |
| 8192 | 4096 | 2 | 757341 | 9347 | 0.987658 | 242661 | 71292 | 0.706207 |

Plot:



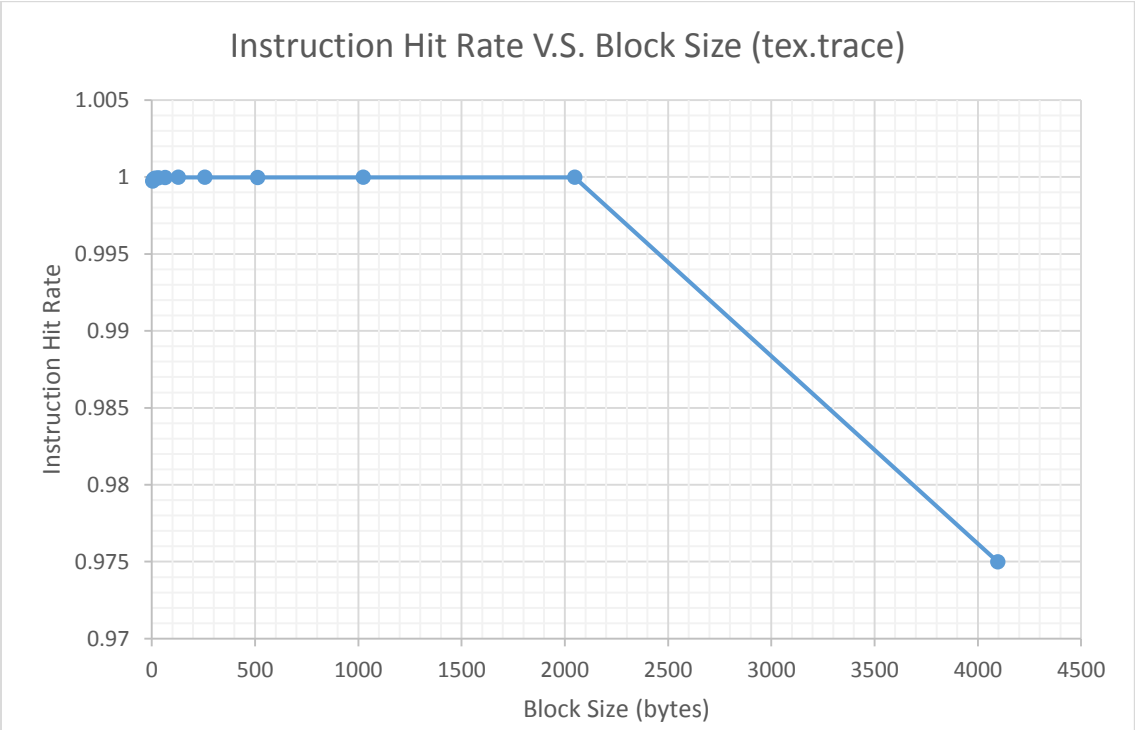


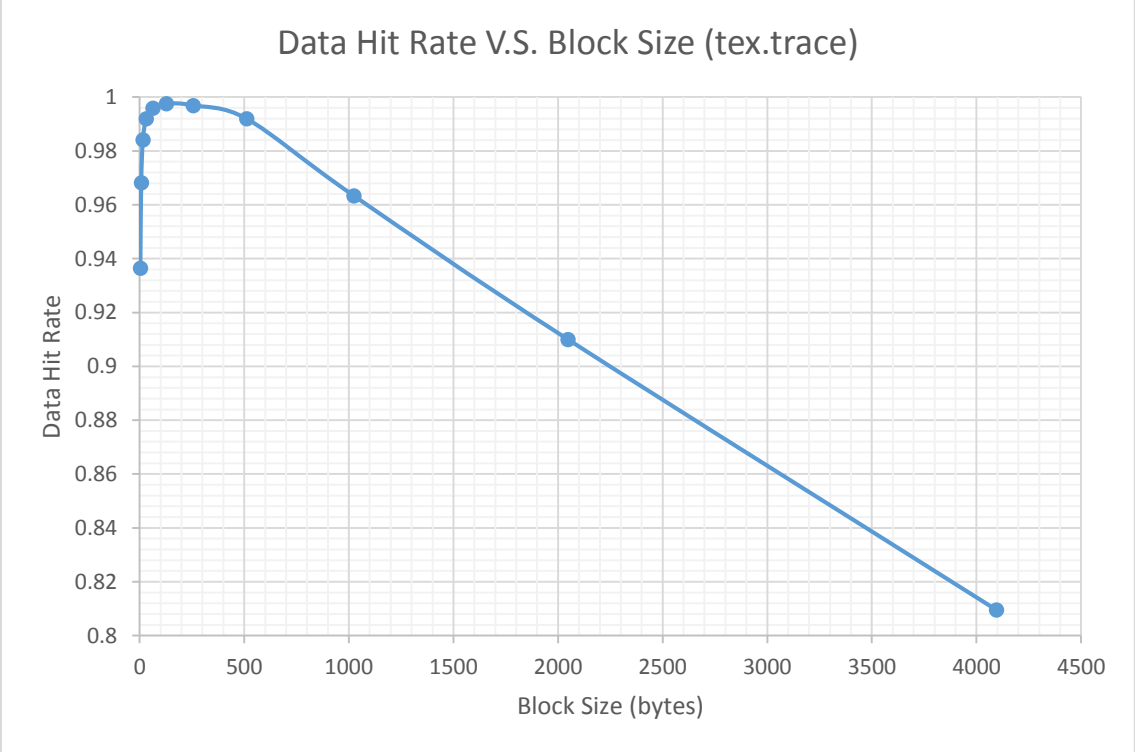

2.2.3 Data and Plot for cc.trace

Data:

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 8192 | 4 | 2 | 597309 | 160 | 0.999732 | 235168 | 14944 | 0.936454 |
| 8192 | 8 | 2 | 597309 | 87 | 0.999854 | 235168 | 7478 | 0.968201 |
| 8192 | 16 | 2 | 597309 | 51 | 0.999915 | 235168 | 3745 | 0.984075 |
| 8192 | 32 | 2 | 597309 | 29 | 0.999951 | 235168 | 1882 | 0.991997 |
| 8192 | 64 | 2 | 597309 | 20 | 0.999967 | 235168 | 961 | 0.995914 |
| 8192 | 128 | 2 | 597309 | 13 | 0.999978 | 235168 | 566 | 0.997593 |
| 8192 | 256 | 2 | 597309 | 10 | 0.999983 | 235168 | 734 | 0.996879 |
| 8192 | 512 | 2 | 597309 | 15 | 0.999975 | 235168 | 1900 | 0.991921 |
| 8192 | 1024 | 2 | 597309 | 12 | 0.99998 | 235168 | 8620 | 0.963345 |
| 8192 | 2048 | 2 | 597309 | 11 | 0.999982 | 235168 | 21175 | 0.909958 |
| 8192 | 4096 | 2 | 597309 | 14936 | 0.974995 | 235168 | 44798 | 0.809506 |

Plot:




2.2.4 Analysis

a. Explain why the hit rate vs. block size plot has the shape that it does. In particular, explain the relevance of spatial locality to the shape of this curve.

The spatial locality concludes that the likelihood of referencing a resource is higher if a resource

near it was just referenced. At the beginning, when the block size starts to increase, the hit rate also increase because of the spatial locality. However, due to the limitation of constant cache size, when we increase the block size, the number of sets will be decreased and thus the resources are more likely to have conflict miss. Also, the spatial locality has a workable scope. When the block size is larger than that scope, some cached blocks will be redundant. Therefore, when we further increase the block size, the cache hit rate will reach a peak and then decrease. There exists an optimal block size.

When the block size is smaller than the spatial locality scope, the curve will be ascending; when the block size is larger than the spatial locality scope, the curve will be descending.

b. What is the optimal block size (consider instruction and data references separately) for each trace?

spice.trace: instruction: 1KB; data: 32 bytes
cc. trace: instruction: 2KB; data: 32 bytes
tex.trace: instruction: 2KB; data: 128 bytes

c. Is the optimal block size for instruction and data references different? What does this tell you about the nature of instruction references versus data references?

Yes. The spatial locality is more applicable to instruction references than data references. Thus the block size for instruction cache should be larger.

## 2.3 Impact of Associativity

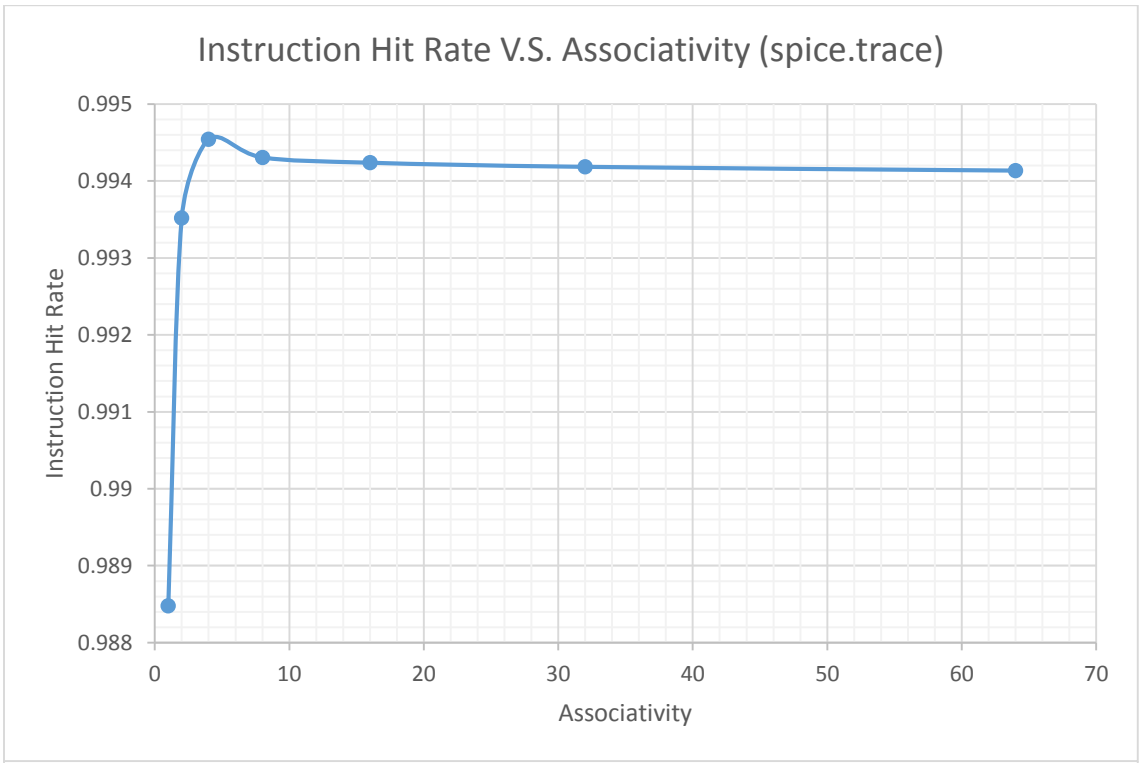
Cache configuration:

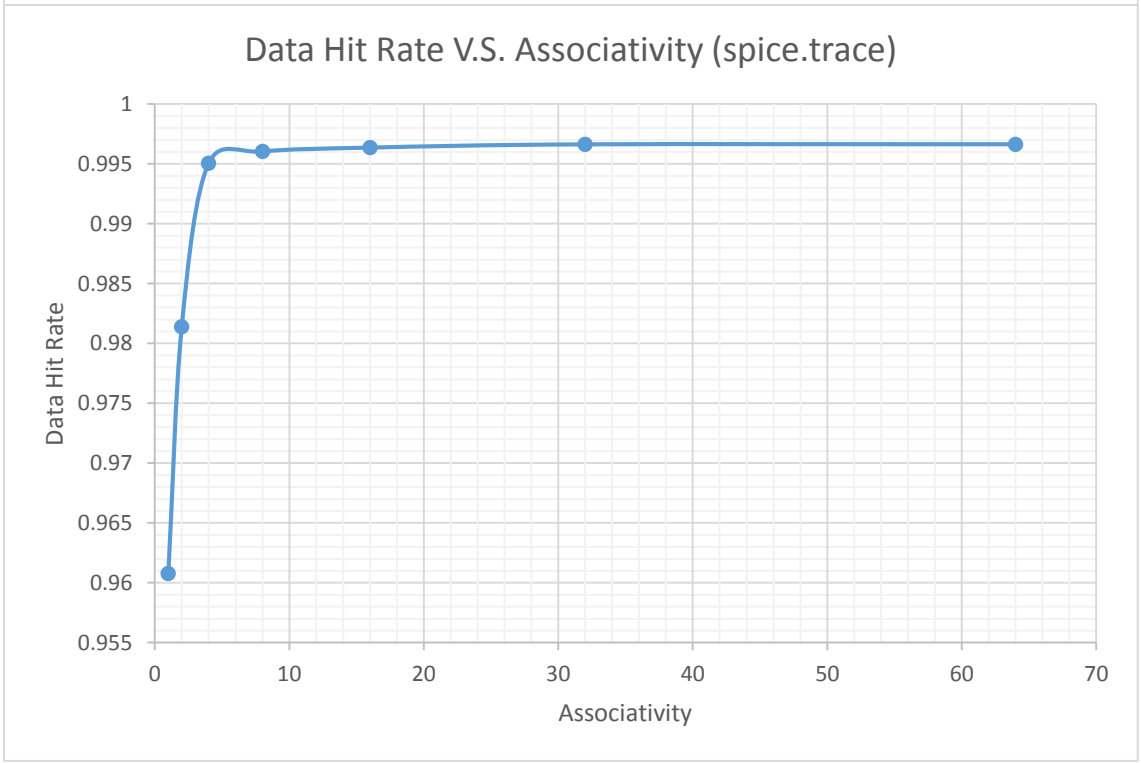
- Split, block size 128 bytes, cache size 8K bytes, write-back, write-allocate
- Associativity from 1 to 64

2.3.1 Data and plot for spice.trace
Data:

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 8192 | 128 | 1 | 782764 | 9018 | 0.988479 | 217237 | 8519 | 0.960785 |
| 8192 | 128 | 2 | 782764 | 5073 | 0.993519 | 217237 | 4039 | 0.981407 |
| 8192 | 128 | 4 | 782764 | 4273 | 0.994541 | 217237 | 1075 | 0.995051 |
| 8192 | 128 | 8 | 782764 | 4457 | 0.994306 | 217237 | 860 | 0.996041 |
| 8192 | 128 | 16 | 782764 | 4511 | 0.994237 | 217237 | 789 | 0.996368 |
| 8192 | 128 | 32 | 782764 | 4553 | 0.994183 | 217237 | 730 | 0.99664 |
| 8192 | 128 | 64 | 782764 | 4591 | 0.994135 | 217237 | 728 | 0.996649 |

Plot:


Instruction Hit Rate V.S. Associativity (spice.trace)
Instruction Hit Rate
0.995
0.994
0.993
0.992
0.991
0.99
0.989
0.988
0
10
20
30
40
50
60
70
Associativity

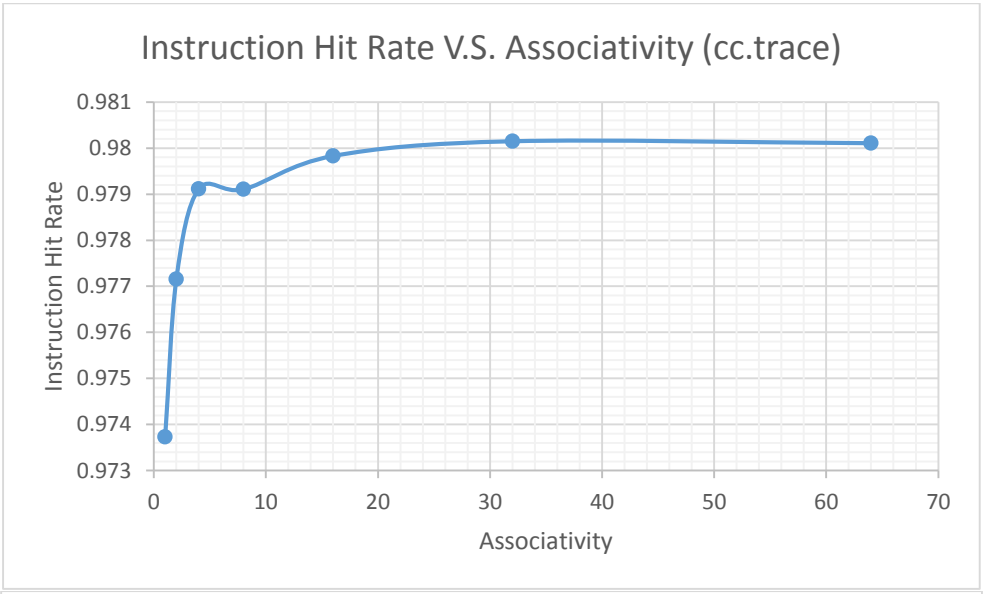

Data Hit Rate V.S. Associativity (spice.trace)
Data Hit Rate
1
0.995
0.99
0.985
0.98
0.975
0.97
0.965
0.96
0.955
0
10
20
30
40
50
60
70
Associativity

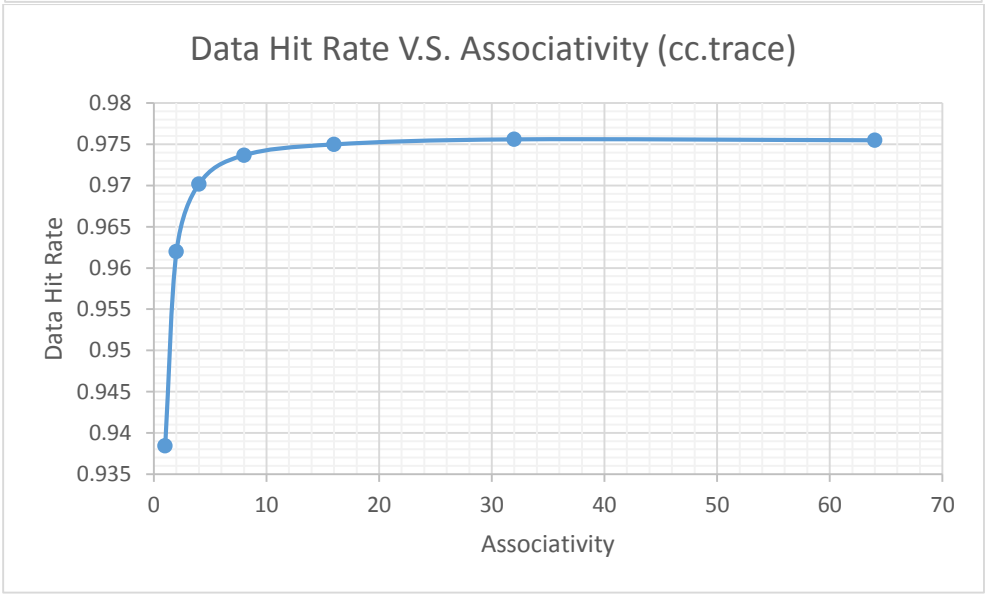
2.3.2 Data and plot for cc.trace

Data:

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 8192 | 128 | 1 | 757341 | 19894 | 0.973732 | 242661 | 14934 | 0.938457 |
| 8192 | 128 | 2 | 757341 | 17301 | 0.977156 | 242661 | 9225 | 0.961984 |
| 8192 | 128 | 4 | 757341 | 15819 | 0.979112 | 242661 | 7232 | 0.970197 |
| 8192 | 128 | 8 | 757341 | 15823 | 0.979107 | 242661 | 6386 | 0.973683 |
| 8192 | 128 | 16 | 757341 | 15275 | 0.979831 | 242661 | 6067 | 0.974998 |
| 8192 | 128 | 32 | 757341 | 15033 | 0.98015 | 242661 | 5919 | 0.975608 |
| 8192 | 128 | 64 | 757341 | 15065 | 0.980108 | 242661 | 5946 | 0.975497 |

Plot:



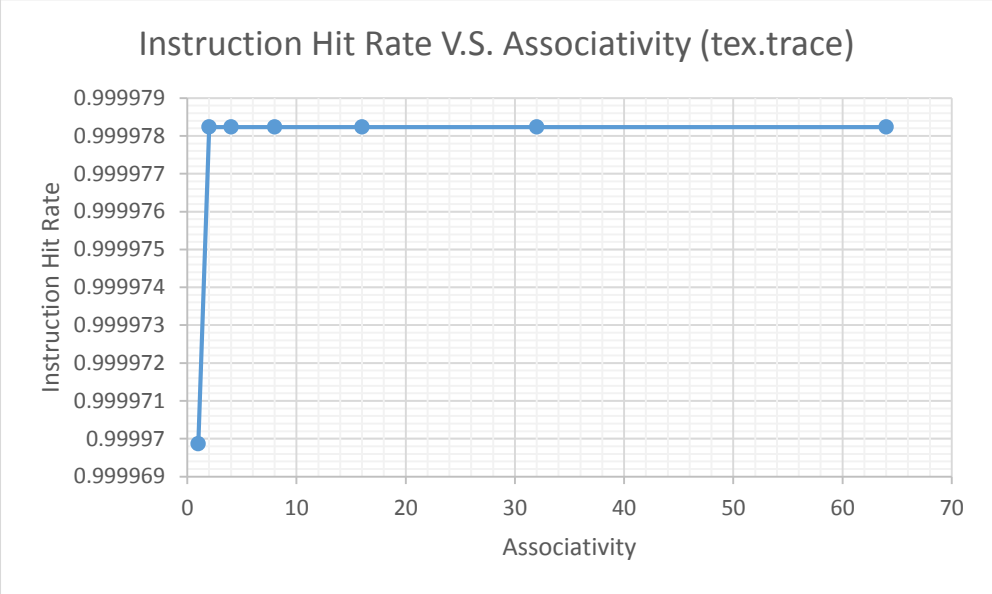



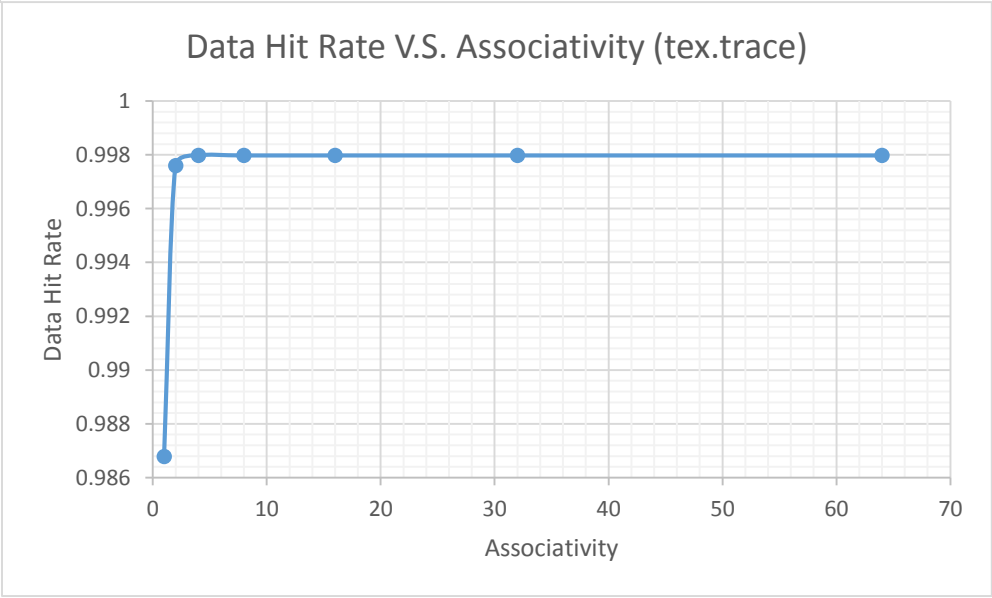
2.3.3 Data and plot for tex.trace

Data

| Cache Size (byte) | Block Size (byte) | Assoc. | Instruction Accesses | Instruction Misses | Instruction Hit Rate | Data Accesses | Data Misses | Data Hit Rate |
|---|---|---|---|---|---|---|---|---|
| 8192 | 128 | 1 | 597309 | 18 | 0.99997 | 235168 | 3107 | 0.986788 |
| 8192 | 128 | 2 | 597309 | 13 | 0.999978 | 235168 | 566 | 0.997593 |
| 8192 | 128 | 4 | 597309 | 13 | 0.999978 | 235168 | 476 | 0.997976 |
| 8192 | 128 | 8 | 597309 | 13 | 0.999978 | 235168 | 476 | 0.997976 |
| 8192 | 128 | 16 | 597309 | 13 | 0.999978 | 235168 | 476 | 0.997976 |
| 8192 | 128 | 32 | 597309 | 13 | 0.999978 | 235168 | 476 | 0.997976 |
| 8192 | 128 | 64 | 597309 | 13 | 0.999978 | 235168 | 476 | 0.997976 |

Plot:



2.3.4 Analysis

a. Explain why the hit rate vs. associativity plot has the shape that it does.

Increasing the associativity will decrease the conflict misses. Thus, at the beginning of the plot, the hit rate increases with the associativity. However, after certain point, the increase of associativity can no longer decrease the conflict misses and result in a diminishing return.

b. Is there a difference between the plots for instruction and data references? What does this tell you about the difference in impact of associativity on instruction versus data references?

From the plot, we can observe that the impact of associativity on instruction and data references are almost the same. The associativity might have more impact on data cache because when we increase the associativity, the improved data hit rate percentage:

(improved data hit rate – original data hit rage)/original data hit rate

is larger than the improved instruction hit rate percentage.

# 2.4 Memory Bandwidth

2.4.1 Fix write-no-allocate policy, compare write-back and write-through policy

| Trace | WB/WT | WA/NW | Cache Size (byte) | Block Size (byte) | Assoc. | DF | CB | Total |
|---|---|---|---|---|---|---|---|---|
| spice | WB | NW | 8K | 64 | 2 | 151104 | 13624 | 164728 |
| spice | WT | NW | 8K | 64 | 2 | 151104 | 66538 | 217642 |
| spice | WB | NW | 8K | 128 | 2 | 280768 | 32287 | 313055 |
| spice | WT | NW | 8K | 128 | 2 | 280768 | 66538 | 347306 |
| spice | WB | NW | 8K | 64 | 4 | 107296 | 9219 | 116515 |
| spice | WT | NW | 8K | 64 | 4 | 107296 | 66538 | 173834 |
| spice | WB | NW | 16K | 64 | 2 | 57008 | 8252 | 65260 |
| spice | WT | NW | 16K | 64 | 2 | 57008 | 66538 | 123546 |
| cc | WB | NW | 8K | 64 | 2 | 474256 | 39426 | 513682 |
| cc | WT | NW | 8K | 64 | 2 | 474256 | 83030 | 557286 |
| cc | WB | NW | 8K | 128 | 2 | 824192 | 78872 | 903064 |
| cc | WT | NW | 8K | 128 | 2 | 824192 | 83030 | 907222 |
| cc | WB | NW | 8K | 64 | 4 | 423104 | 32102 | 455206 |
| cc | WT | NW | 8K | 64 | 4 | 423104 | 83030 | 506134 |
| cc | WB | NW | 16K | 64 | 2 | 252096 | 24048 | 276144 |
| cc | WT | NW | 16K | 64 | 2 | 252096 | 83030 | 335126 |
| tex | WB | NW | 8K | 64 | 2 | 2880 | 29903 | 32783 |
| tex | WT | NW | 8K | 64 | 2 | 2880 | 104513 | 107393 |
| tex | WB | NW | 8K | 128 | 2 | 3488 | 29935 | 33423 |
| tex | WT | NW | 8K | 128 | 2 | 3488 | 104513 | 108001 |
| tex | WB | NW | 8K | 64 | 4 | 3312 | 29903 | 33215 |
| tex | WT | NW | 8K | 64 | 4 | 3312 | 104513 | 107825 |
| tex | WB | NW | 16K | 64 | 2 | 2464 | 29909 | 32373 |
| tex | WT | NW | 16K | 64 | 2 | 2464 | 104513 | 106977 |

2.4.2 Fix write-back policy and compare write-allocate and write-no-allocate policy

| Trace | WB/WT | WA/NW | Cache Size (byte) | Block Size (byte) | Assoc. | DF | CB | Total |
|---|---|---|---|---|---|---|---|---|
| spice | WB | WA | 8K | 64 | 2 | 156000 | 18880 | 174880 |
| spice | WB | NW | 8K | 64 | 2 | 151104 | 13624 | 164728 |
| spice | WB | WA | 8K | 128 | 2 | 291584 | 36256 | 327840 |
| spice | WB | NW | 8K | 128 | 2 | 280768 | 32287 | 313055 |
| spice | WB | WA | 8K | 64 | 4 | 110400 | 7296 | 117696 |
| spice | WB | NW | 8K | 64 | 4 | 107296 | 9219 | 116515 |
| spice | WB | WA | 16K | 64 | 2 | 59856 | 6208 | 66064 |
| spice | WB | NW | 16K | 64 | 2 | 57008 | 8252 | 65260 |
| cc | WB | WA | 8K | 64 | 2 | 488976 | 41840 | 530816 |
| cc | WB | NW | 8K | 64 | 2 | 474256 | 39426 | 513682 |
| cc | WB | WA | 8K | 128 | 2 | 848832 | 91744 | 940576 |
| cc | WB | NW | 8K | 128 | 2 | 824192 | 78872 | 903064 |
| cc | WB | WA | 8K | 64 | 4 | 435408 | 33216 | 468624 |
| cc | WB | NW | 8K | 64 | 4 | 423104 | 32102 | 455206 |
| cc | WB | WA | 16K | 64 | 2 | 261248 | 23584 | 284832 |
| cc | WB | NW | 16K | 64 | 2 | 252096 | 24048 | 276144 |
| tex | WB | WA | 8K | 64 | 2 | 15696 | 7664 | 23360 |
| tex | WB | NW | 8K | 64 | 2 | 2880 | 29903 | 32783 |
| tex | WB | WA | 8K | 128 | 2 | 18528 | 8608 | 27136 |
| tex | WB | NW | 8K | 128 | 2 | 3488 | 29935 | 33423 |
| tex | WB | WA | 8K | 64 | 4 | 15408 | 7568 | 22976 |
| tex | WB | NW | 8K | 64 | 4 | 3312 | 29903 | 33215 |
| tex | WB | WA | 16K | 64 | 2 | 11696 | 7616 | 19312 |
| tex | WB | NW | 16K | 64 | 2 | 2464 | 29909 | 32373 |

2.4.3 Analysis

a. Which cache has the smaller memory traffic, the write-through cache or the write-back cache? Why?

Write-back cache

b. Is there any scenario under which your answer to the question above would flip? Explain.

Yes. When block size is large, for write-back policy, copies_back will cost a lot. For example, I test the three files with the following configurations:

| Trace | WB/WT | WA/NW | Cache Size (byte) | Block Size (byte) | Assoc. | DF | CB | Total |
|---|---|---|---|---|---|---|---|---|
| spice | WB | NW | 8K | 1K | 2 | 3669248 | 1199855 | **4869103** |
| spice | WT | NW | 8K | 1K | 2 | 3669248 | 66538 | **3735786** |
| cc | WB | NW | 8K | 1K | 2 | 8280576 | 1291279 | **9571855** |
| cc | WT | NW | 8K | 1K | 2 | 8280576 | 83030 | **8363606** |
| tex | WB | NW | 8K | 1K | 2 | 506368 | 30368 | **536736** |
| tex | WT | NW | 8K | 1K | 2 | 506368 | 104513 | **610881** |

We can observed that for the first two test file, the total memory traffic for write-through policy is smaller than the that for the write-back policy.

c. Which cache has the smaller memory traffic, the write-allocate or the write-no-allocate cache? Why?

Write-no-allocate

d. Is there any scenario under which your answer to the question above would flip? Explain

tex.trace gives us a good example. With write allocate and write-back policy, the program might have more write hits and thus the data needed to be written back is decreased. So even though the write-no-allocate policy might have less data fetches, it can have more data needed to be written back, and thus results in a larger total memory traffic.