Assignment 3 SimpleScalar Report

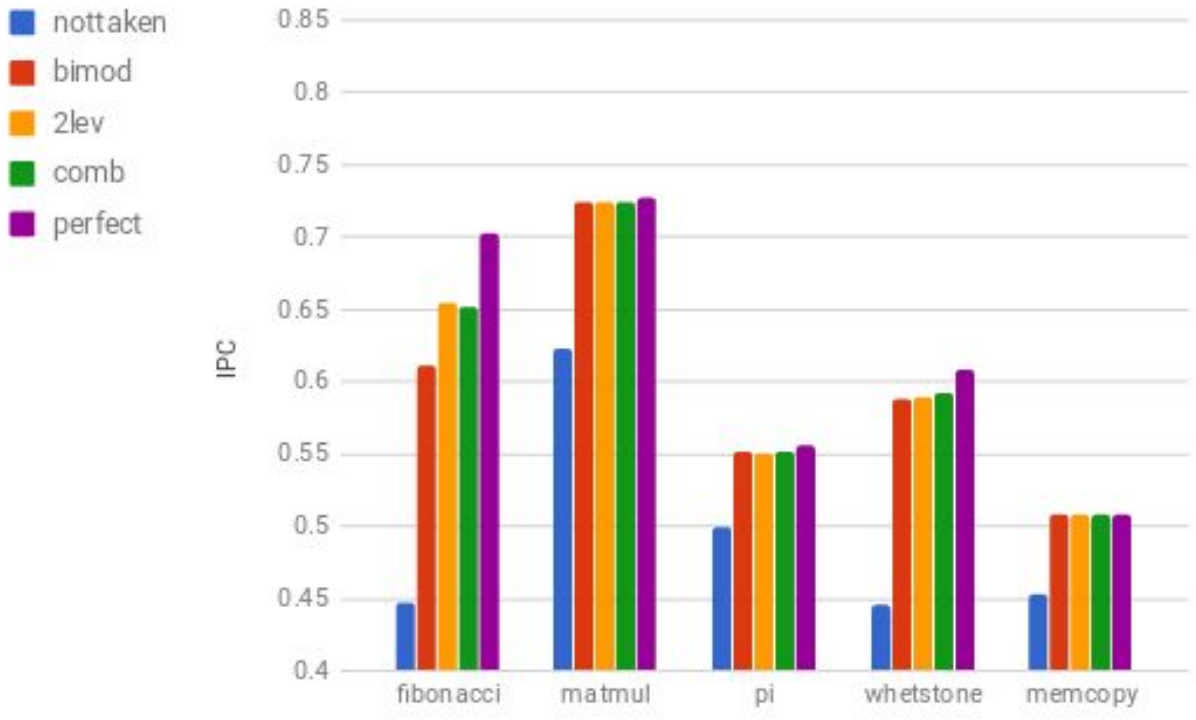
Shan Kuan, Yuting Fu

Exercise 1

IPC Results:

	nottaken	bimod	2lev	comb	perfect
fibonacci	0.4472	0.6107	0.6547	0.6523	0.7025
matmul	0.6226	0.7243	0.7243	0.7243	0.7267
pi	0.4989	0.551	0.5509	0.5511	0.5554
whetstone	0.4457	0.5876	0.5895	0.593	0.6077
memcopy	0.453	0.5086	0.5086	0.5086	0.5087

Bar Graphic of the above IPC Results :



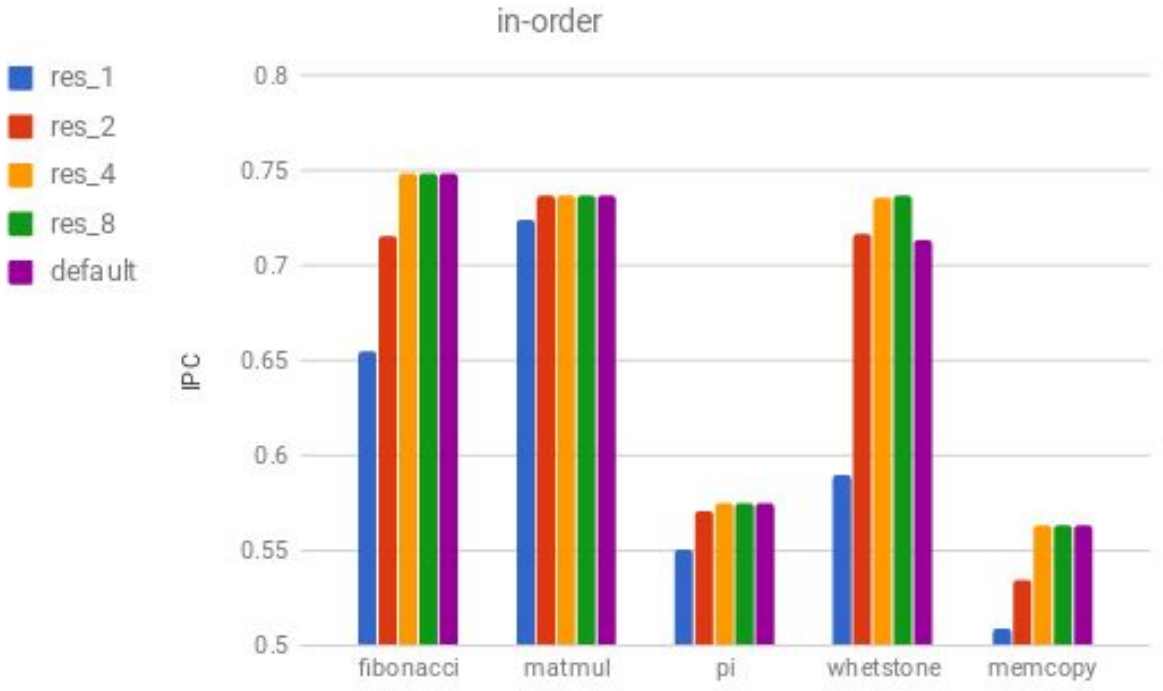
Exercise 2

- in-order

IPC Results:

in-order	res_1	res_2	res_4	res_8	default
fibonacci	0.6547	0.716	0.7489	0.749	0.749
matmul	0.7243	0.7369	0.7371	0.7372	0.7372
pi	0.5509	0.5703	0.5748	0.5754	0.5754
whetstone	0.5895	0.7169	0.7359	0.7373	0.7132
memcopy	0.5086	0.5348	0.5638	0.5638	0.5638

IPC Results in bar graphic:



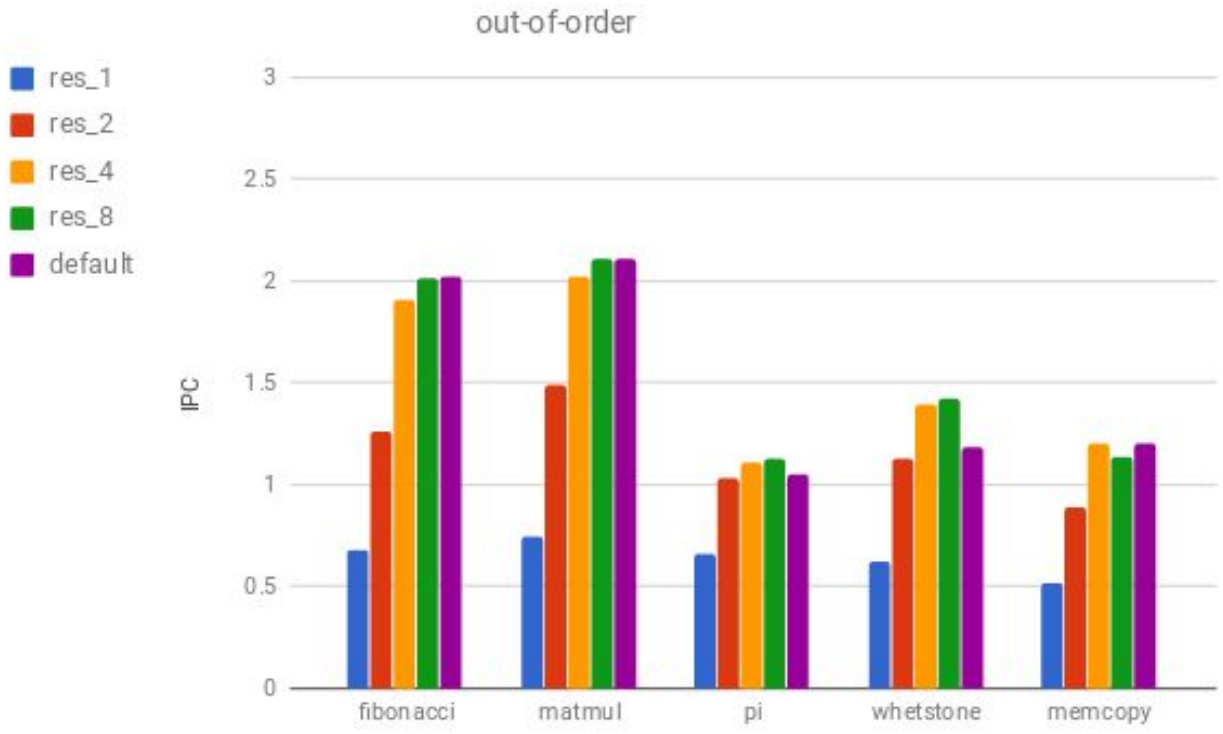
- Out of order

IPC Results:

out-of-order	res_1	res_2	res_4	res_8	default
fibonacci	0.6793	1.2647	1.9052	2.0114	2.0201
matmul	0.7478	1.4864	2.0207	2.1119	2.1119
pi	0.659	1.0277	1.1076	1.1244	1.0492

whetstone	0.626	1.1296	1.3895	1.4232	1.1887
memcopy	0.5217	0.8906	1.2039	1.1388	1.2044

IPC Results in bar graphic



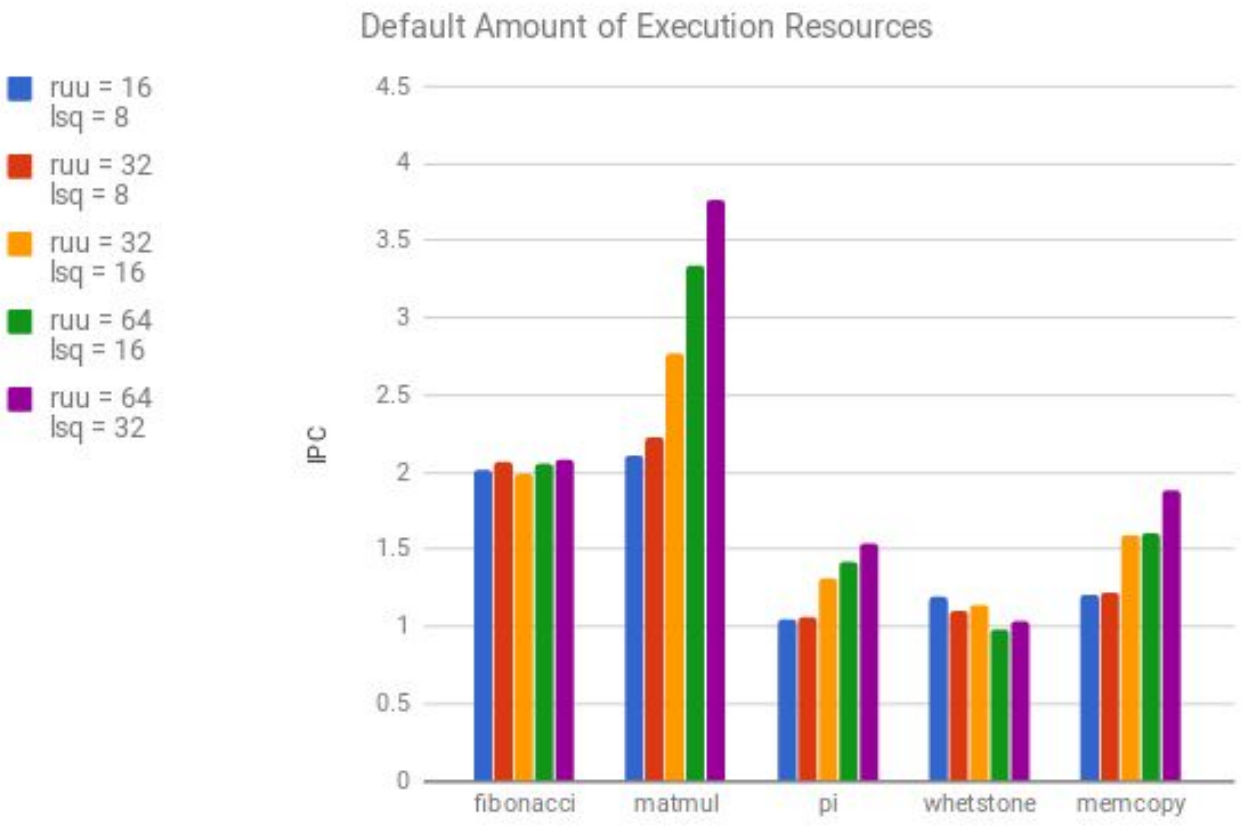
Exercise 3

- Default amount of execution resources

IPC Results:

Default Amount of Exe. Resources	ruu = 16 lsq = 8	ruu = 32 lsq = 8	ruu = 32 lsq = 16	ruu = 64 lsq = 16	ruu = 64 lsq = 32
fibonacci	2.0201	2.0722	1.9908	2.0591	2.0876
matmul	2.1119	2.2308	2.7657	3.3404	3.7595
pi	1.0492	1.0646	1.3072	1.4126	1.5446
whetstone	1.1887	1.1043	1.1466	0.9851	1.0316
memcopy	1.2044	1.2177	1.5909	1.6003	1.8868

Bar graphic:

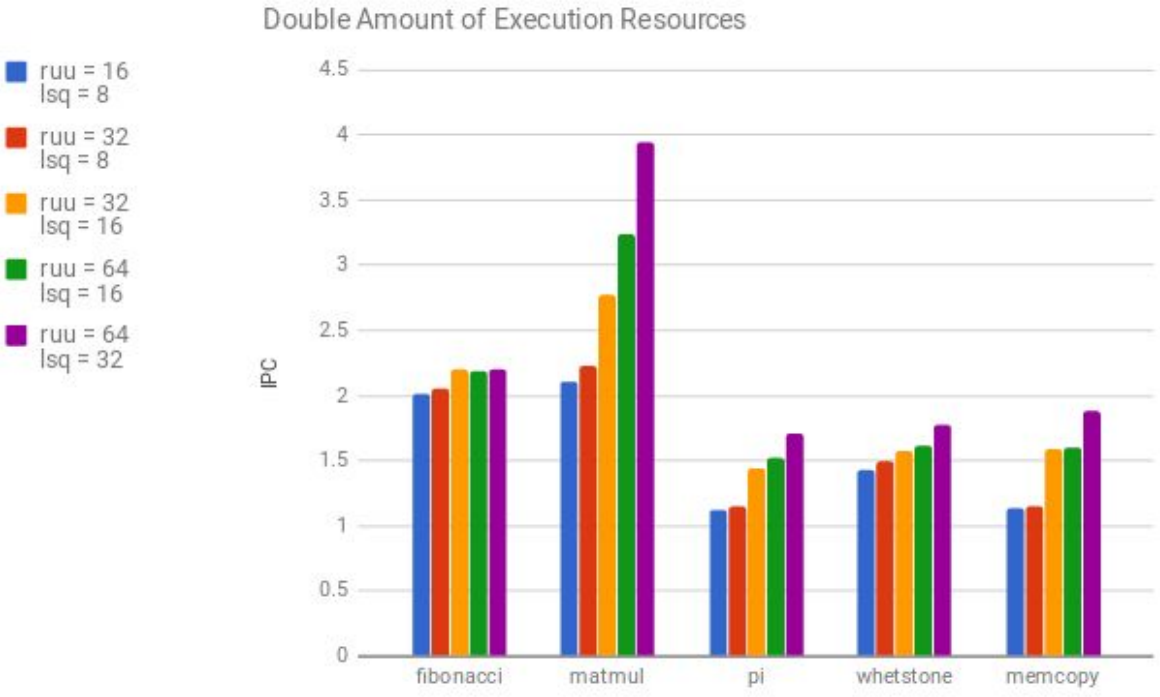


- Double amount of execution resources

IPC Results:

Double Amount of Exe. Resources	ruu = 16 lsq = 8	ruu = 32 lsq = 8	ruu = 32 lsq = 16	ruu = 64 lsq = 16	ruu = 64 lsq = 32
fibonacci	2.0114	2.0506	2.2055	2.1804	2.202
matmul	2.1119	2.2307	2.769	3.2409	3.9499
pi	1.1244	1.1417	1.4454	1.5217	1.7011
whetstone	1.4232	1.4988	1.5782	1.6193	1.773
memcopy	1.1388	1.1507	1.5909	1.6004	1.8869

Bar graphic:



Exercise 4

From Exercise 1 we can conclude that the ability of the predictor to accurately predict if the branch will be taken or not limits the IPC for the benchmarks. This is strongly supported by the fact that the nottaken predictor performs the worse on each and every benchmark, while bimod, 2lev and comb all work almost as well as the perfect predictor on all the benchmarks except the fibonacci.

In general, increasing the processor width and instruction window improves the IPC on each benchmark, as shown in Exercise 2, because it allows a bigger set of instructions to be examined in each cycle, which strengthens the parallel execution. And the results get even better when switched from in-order to out-of-order execution. Because out-of-order execution uses a reorder buffer to automatically rename registers and hold the results of pending instructions and thus sinificantly eliminate the stalls. In addition, the value of IPC declines when wetstone and pi take the default setting (4 ialu, 1 imult, 4 fpalu, 1 imult) . The reason is that the number of mul/div is 1 by default. However, wetstone and pi benchmarks have numerous mul/div operation in execution.

Increasing the load store queue also has a very positive impact on most of the benchmarks, judging from the results of Exercise 3. Store values are placed in the queue if the store is speculative. Loads are dispatched to the memory system when the addresses of all previous stores are known. Loads may be satisfied either by the memory system or by an earlier store value residing in the queue, if their addresses match. Increasing the queue obviously improves the accuracy of the predictor and enables the predictor to deal with more complex branches.

As to the benchmark that has the lowest IPC improvements, in Exercise 1, benchmark pi has the least improvement, because it only has one simple branch throughout

the program, prediction has very limited impact on the overall performance since it is just a simple program. In Exercise 2, the least improvement occurs in benchmark matmul in the in-order execution, because there are strong data dependencies among the matmul instructions and any optimization has limited impact as long as it is still executed in the original order. In Exercise 3, the lowest improvement again happens to fibonacci benchmark, because it only deals with a small amount of variables, which explains why doubling amount of execution resources or increasing the load store queue has little to do with the IPC outcome.

Exercise 5

AMAT = Hit Rate + Miss Rate * Miss Penalty

IPC and AMAT before the increased memory latency:

Default Memory latency <18 cycles>

IPC: 1.5901

AMAT: 6.297877

IPC and AMAT after the increased memory latency:

Increase Memory latency <200 cycles>

IPC: 0.5435

AMAT: 37.401677

- Proposed cache configuration:

In this section we choose to keep the cache L2 block size unchanged, and look into the impact of different number of sets, associativity and memory latency on IPC and AMAT as well, and try to find out the best configuration possible. We change the memory latency from default 6 to 7 and 8, and associativity from default 2 to 4, 8, 16, 32, and the number of sets from default 1024 to 2048, 4096 and 8192.

IPC and AMAT results of different configuration are shown in the figures below.

From the chart we can see that when the number of sets of L2 is 8192, cache hit latency is 6, and associativity is 32, the processor has the best performance, judging from the IPC.Obviously, under the same number of sets, same memory latency, the bigger the associativity, the smaller the AMAT is. Under the same condition in which IPC gets the best, the AMAT also gets the smallest value. Thus we conclude that this configuration is the best and we choose it as our cache configuration, which to be specific, the detailed configuration would be:

-cache:d12lat 6 #default

-cache L2: ul2:8192:64:32:l

-cache:il2lat 6 #default

-mem:lat 200 2

-mem:width 8 #default

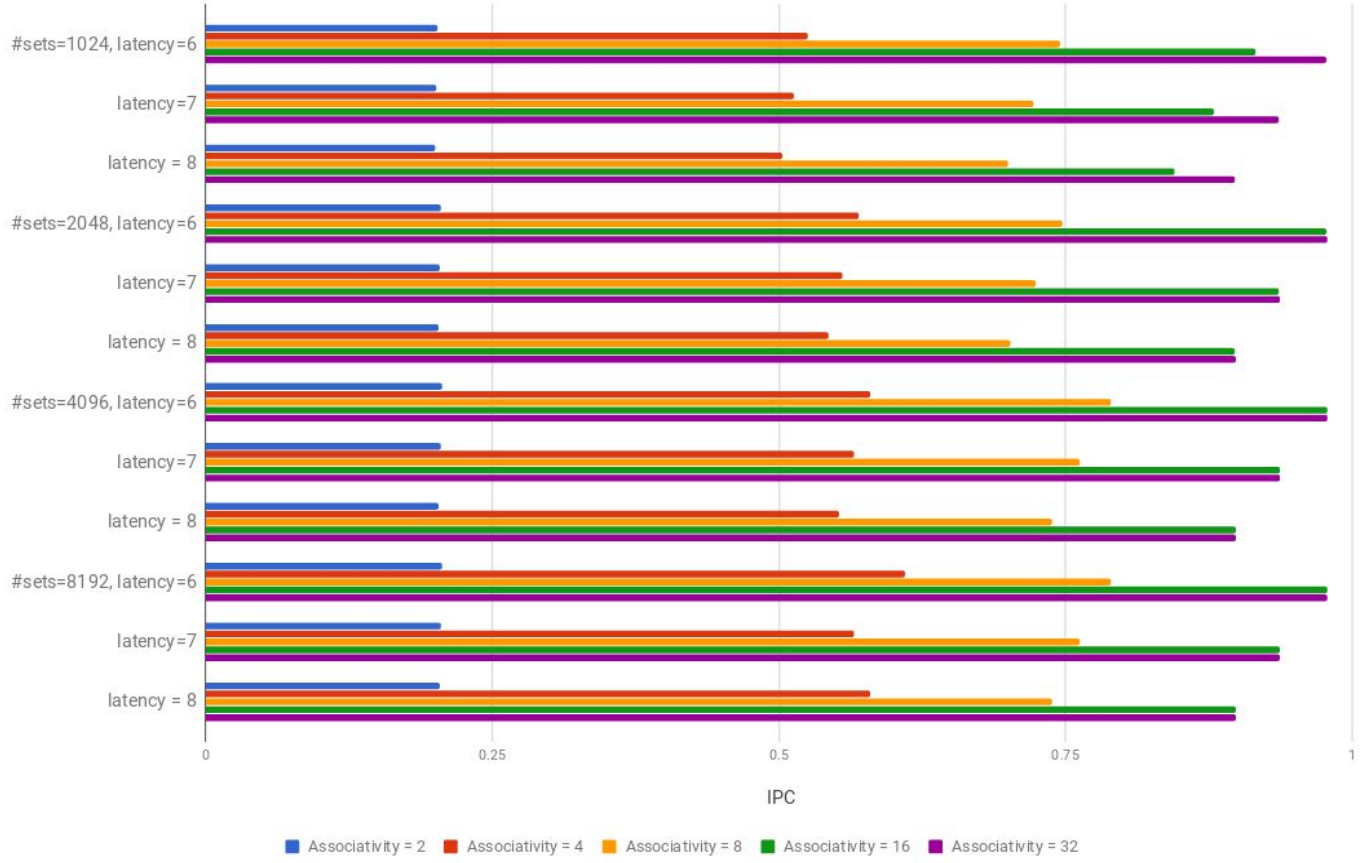
And the final IPC and AMAT of our cache would be:

IPC = 0.9784

AMAT = 10.798043

We can also conclude that the changing of number of sets, or the cache hit latency alone has very limited impact on the IPC or AMAT outcome. On the contrary, he associativity is the decisive factor on these two indicators.

IPC results:



AMAT results: