COL729: Lab 0 Benchmarking

Intel compiler, when incorporated with high optimization(O3), performs better than the other two compilers(with same optimization level, that is O3) in general. In most of the benchmarks, Intel compiler and GCC are observed to perform nearly equally on average. To be more specific, the performance of GCC and Intel compiler is not very different. But, in some benchmarks(eg. mcf_r, deepsjeng_r, leela_r), the superiority of Intel compiler over other two is significantly observed. Intel compiler is found to be very efficient at vectorization of loops. There are benchmarks where clang is found to outperform GCC with some specific optimization level(eg. x264_r). There are also some cases where clang exhibits significantly poorer performance compared to GCC(eg. xalancbmk_r,leela_r). A problem regarding the LLVM compiler is that the memory model for LLVM's intermediate representation (IR) is informal and the semantics of corner cases are not always clear to all compiler developers. For that reason, it might mis-compile certain C/C++ programs. GCC is a stable compiler which is said to have lesser bugs compared to LLVM.

In general, the 64-bit compilers seem to be faster than 32-bit compilers in the case of GCC. The scores obtained by GCC 64-bit compiler is better than or nearly comparable to GCC 32-bit compiler across almost all the benchmarks.

In case of clang compiler with optimization levels O1 and O2, the scores obtained by 64-bit version is greater than those obtained by the 32-bit version in almost all benchmarks. When the optimization level is O3, a similar trend is observed except an outlier(eg. omnetpp_r).

In case of Intel compiler, 64-bit seems to obtain higher scores than 32-bit compiler across all the benchmarks. The trend is similar to that of the GCC.

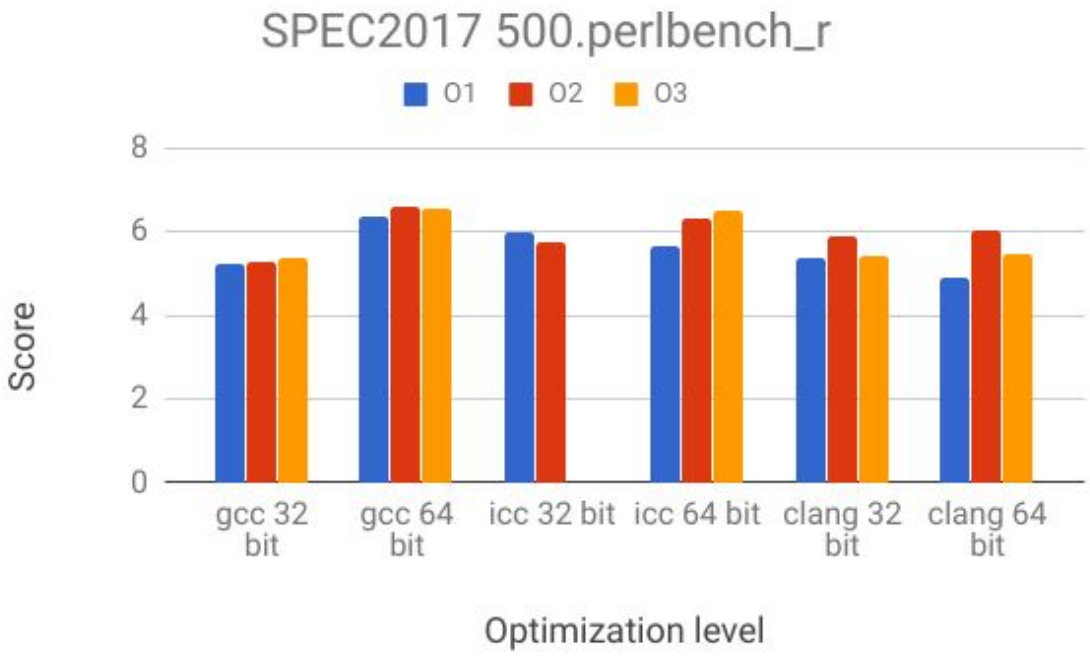
In GCC compilers(both 32-bit and 64-bit versions), the scores obtained by almost all the benchmarks monotonically increases as the optimization levels become higher.

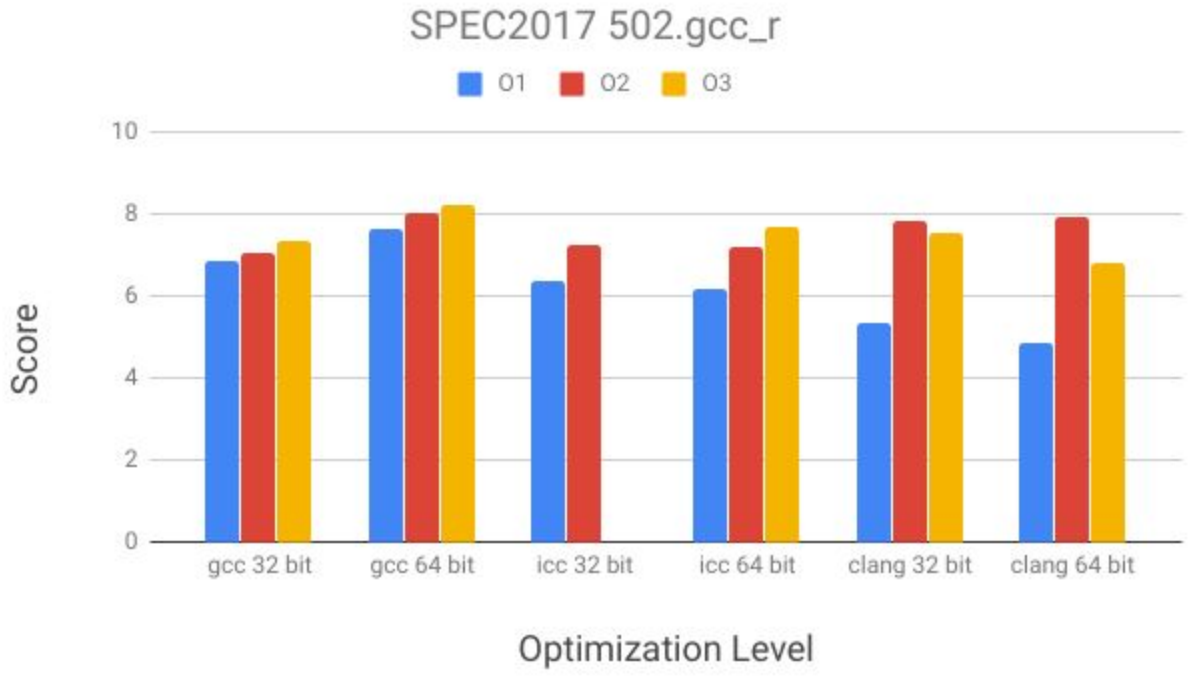
In the case of clang, O2 optimization level leads to higher score compared to other optimizations O1 and O3. Also in almost all benchmarks, the performance differences between O2 and O1 are much higher than the differences between O2 and O3.

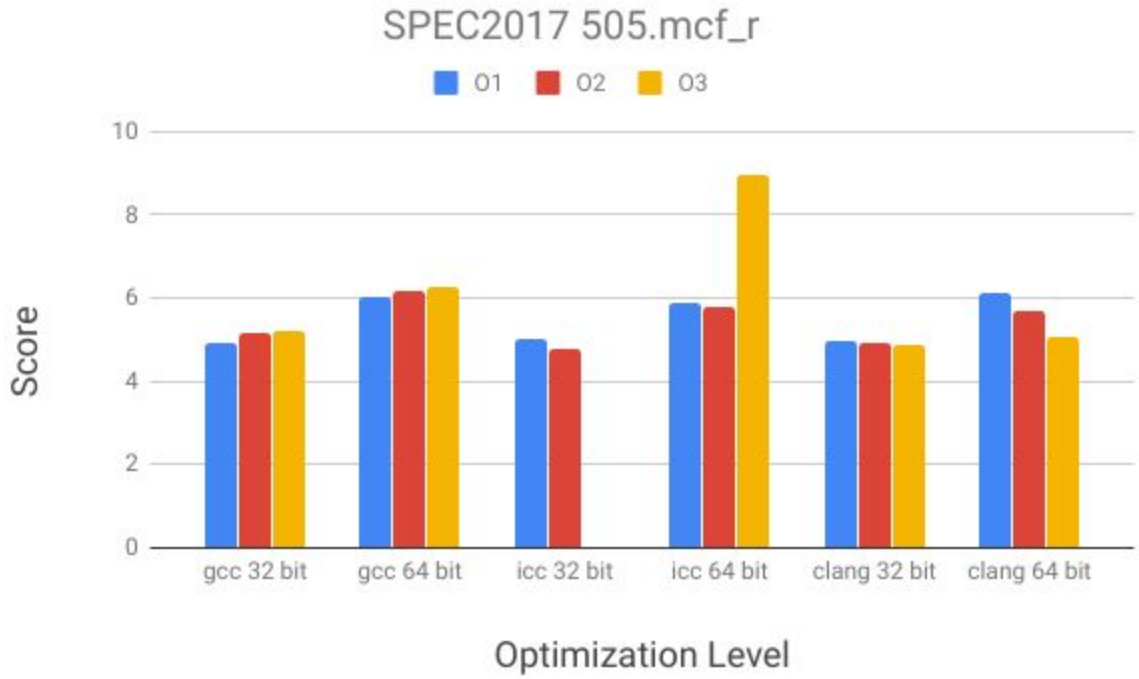
In the case of Intel compiler, performances get improved as optimization levels become higher. And this trend is stable in case of Intel compiler.

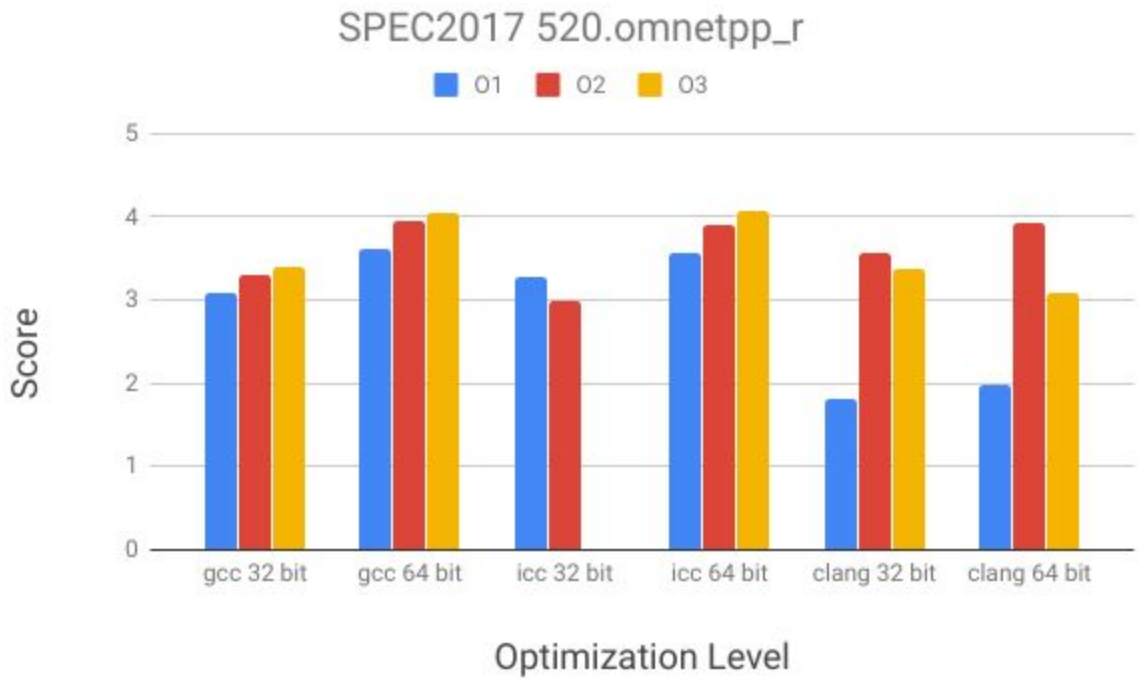
The relative performances of different benchmarks across several extents of optimization levels(O1, O2, O3) are not observed to be benchmark-independent. As for example, the GCC compiler seems to be better, in terms of the scores obtained due to benchmarking, compared to the other two in some benchmarks like gcc_r, omnetpp_r. Intel compiler with high optimization significantly outperforms the other two in case of x264_r benchmark. Some benchmark, like x264_r, shows a considerably large difference in performance between least and highest optimization level for Intel compiler. In some benchmarks like exchange2_r, omnetpp_r etc, clang, with the highest degree of optimization also, seems to perform poorly compared to the other two with their highest optimization levels. So, the performance scores are benchmark dependent. Across the three different compilers considered here, there is no general trend in the performance-scores which are followed in the benchmarking.

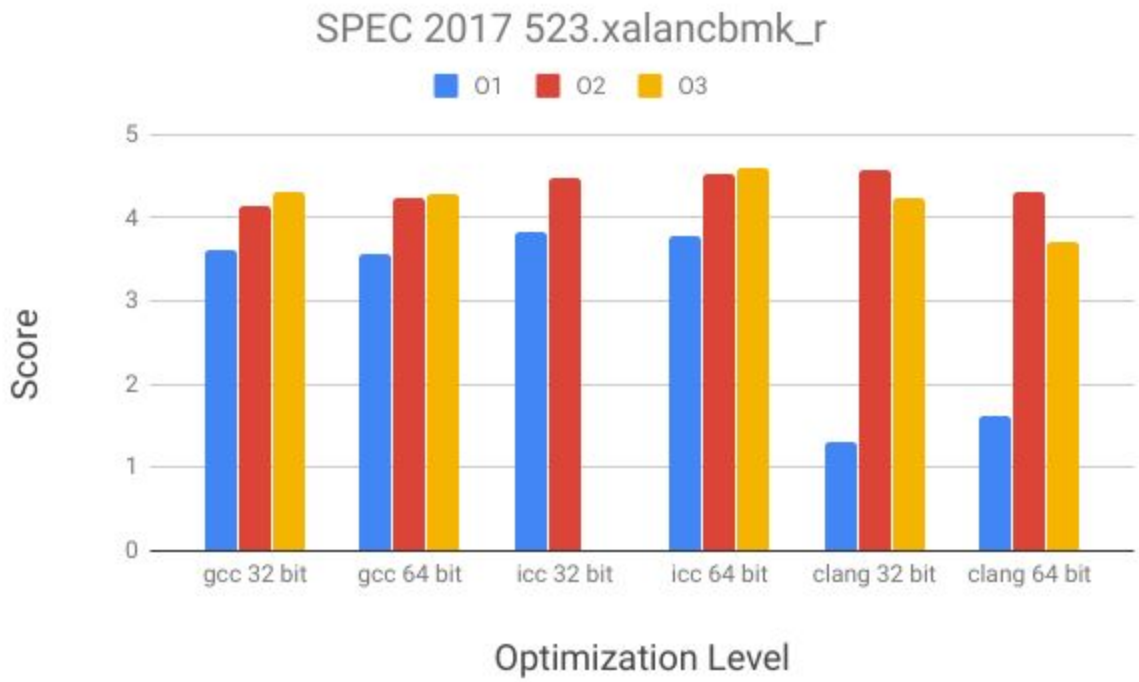
Graphical Plots: **Score** vs **Optimization Level** across all the benchmarks

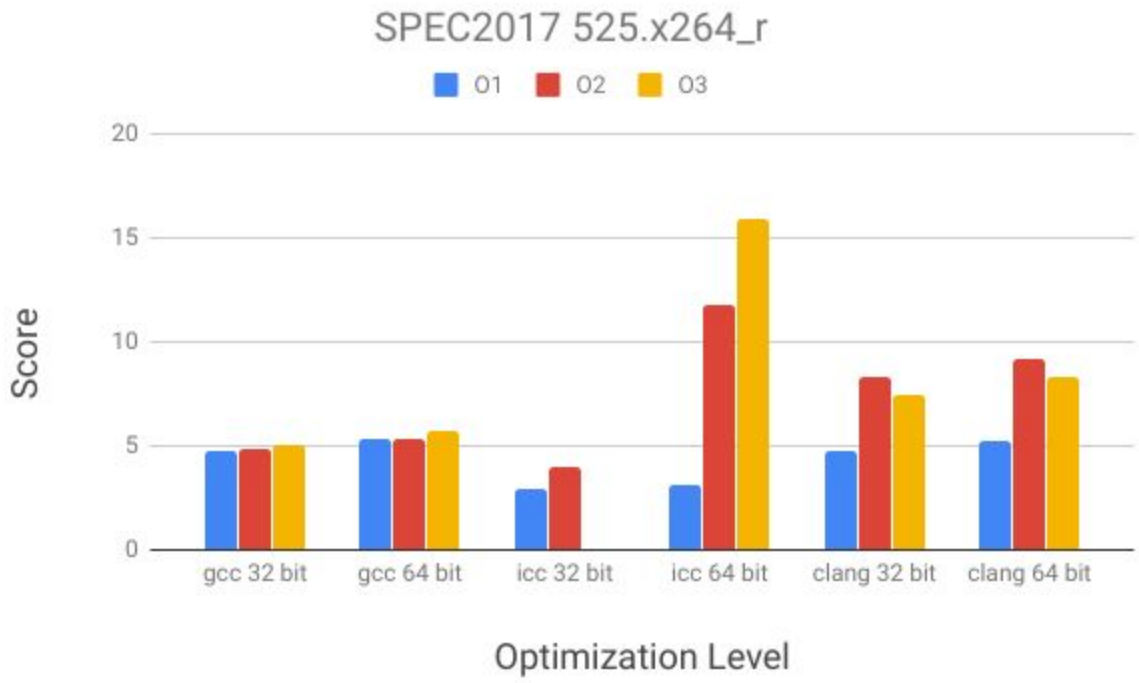


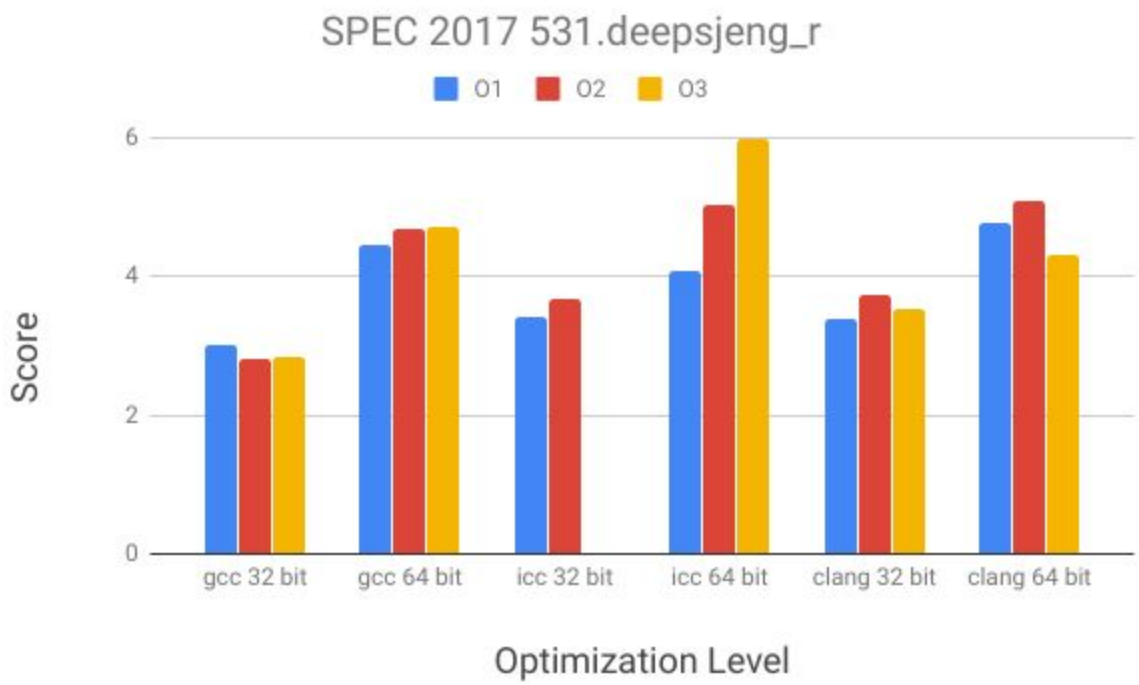

SPEC2017 502.gcc_r
O1
O2
O3
10
8
6
4
2
0
Score
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level

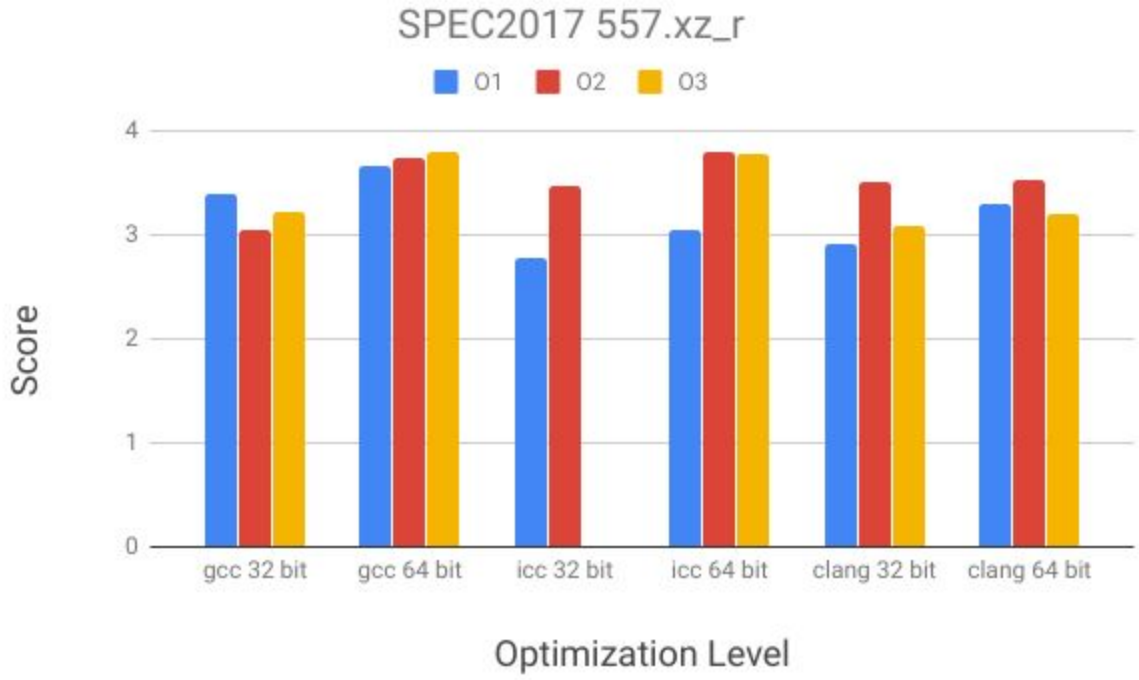

SPEC2017 505.mcf_r
O1
O2
O3
10
8
6
4
2
0
Score
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level

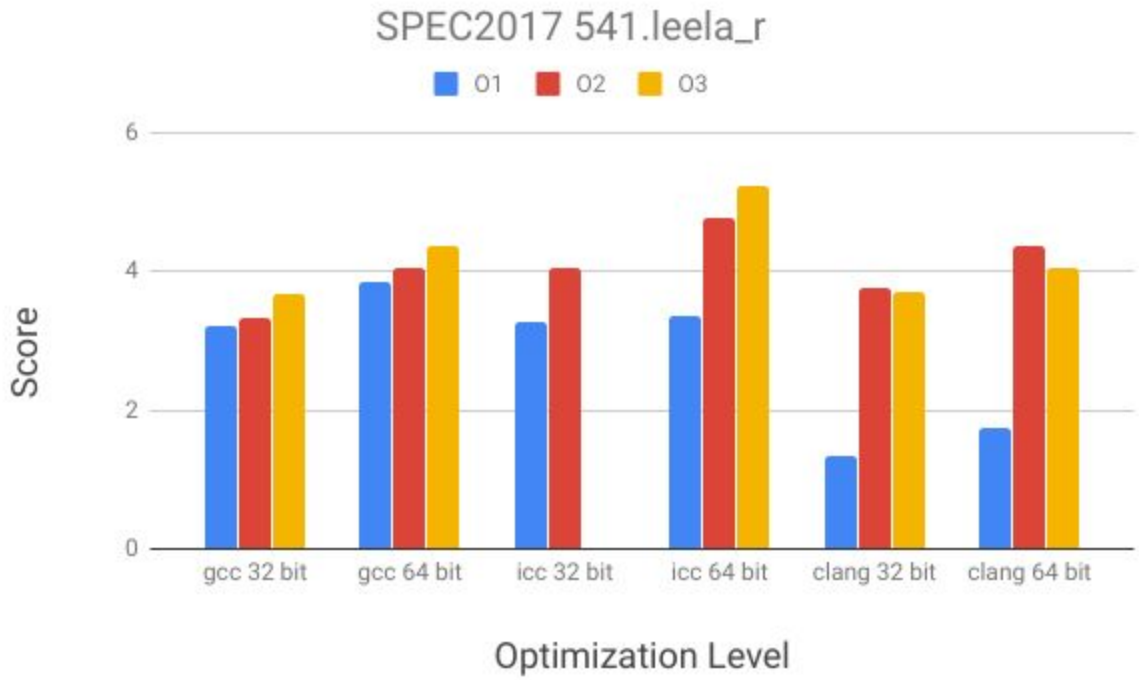

SPEC2017 520.omnetpp_r
O1
O2
O3
Score
5
4
3
2
1
0
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level

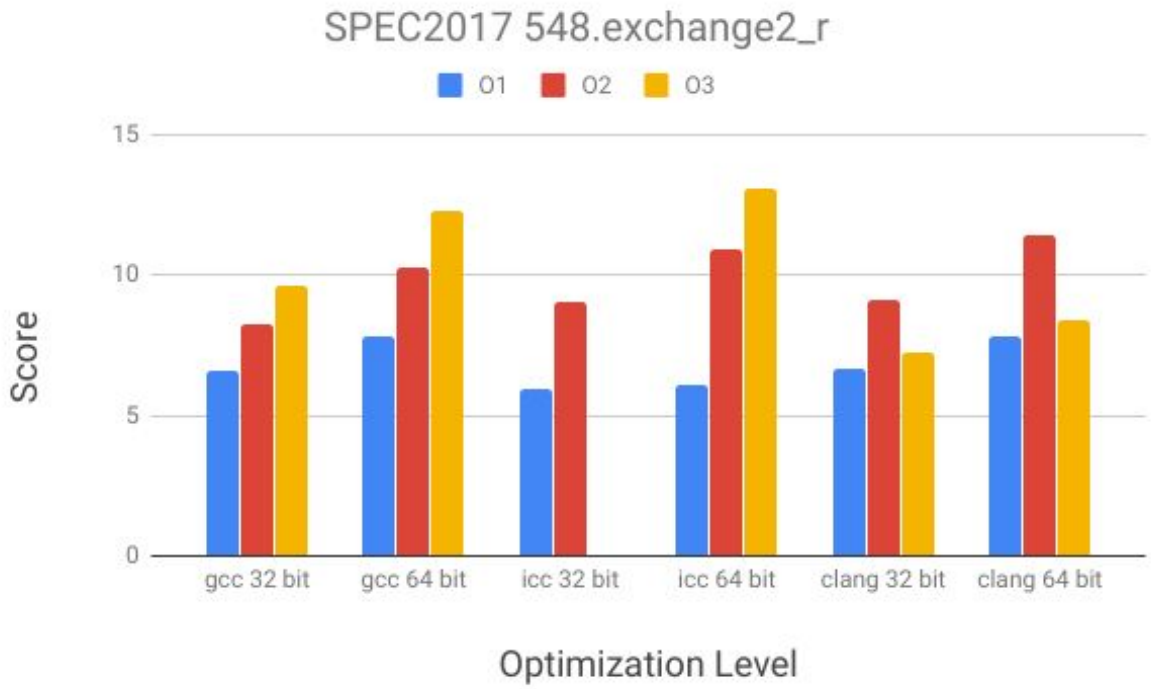

SPEC 2017 523.xalancbmk_r
O1
O2
O3
Score
5
4
3
2
1
0
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level


SPEC2017 525.x264_r
O1
O2
O3
20
15
10
5
0
Score
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level


SPEC 2017 531.deepsjeng_r
O1
O2
O3
6
4
2
0
Score
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level


SPEC2017 557.xz_r
O1
O2
O3
4
3
2
1
0
Score
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level


SPEC2017 541.leela_r
O1
O2
O3
6
4
2
0
Score
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level


SPEC2017 548.exchange2_r
O1
O2
O3
15
10
5
0
Score
gcc 32 bit
gcc 64 bit
icc 32 bit
icc 64 bit
clang 32 bit
clang 64 bit
Optimization Level