COL729 Lab0 Report

Manish Yadav
2015CS10460

1 32-bit vs 64-bit executables

It is observed that 32-bit executables are usually slower than the 64-bit ones. For a few benchmarks, the difference is large, but in most cases, 32-bit executables are only slightly slower than the 64-bit ones. It's also compiler dependent as can be seen in the graphs. For example, 525.x264_r takes longer time for 32-bit in clang but approximatly similar time in gcc whereas for 520.omnetpp_r, it's the opposite. The result also seem to depend on the optimization level. For example, using -O3 optimization in 523.xalancbmk_r, the 32-bit executables outperformed the 64-bit ones. The explanation for this is given in the optimization section and the graph can be seen in compiler comparison section.

The following graphs show runtime comparison for 32-bit and 64-bit using gcc and clang (unoptimized) -

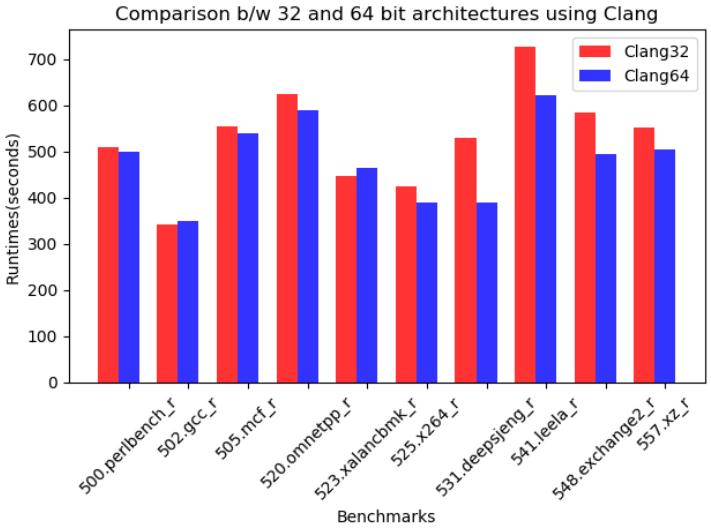

Figure 1: 32-bit vs 64-bit comparison in clang

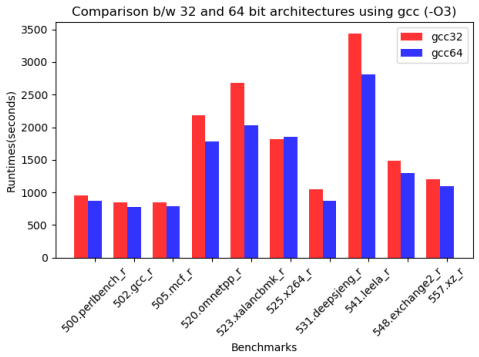

Figure 2: 32-bit vs 64-bit comparison in gcc

2 Compiler comparisons

It was found that icc is better than gcc and clang in most of the cases for both 32 and 64-bit executables. Clang performed only slightly better than gcc in most of the benchmarks except for 531.deepsjeng_r and 525.x264_r where clang outperformed even icc. The graphs are shown below (all use -O3) -

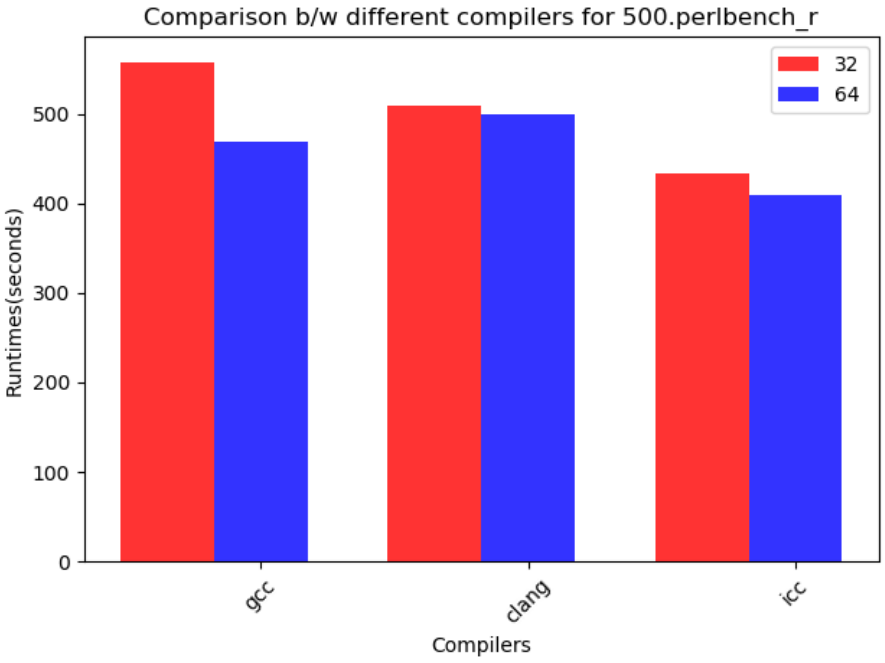

Figure 3

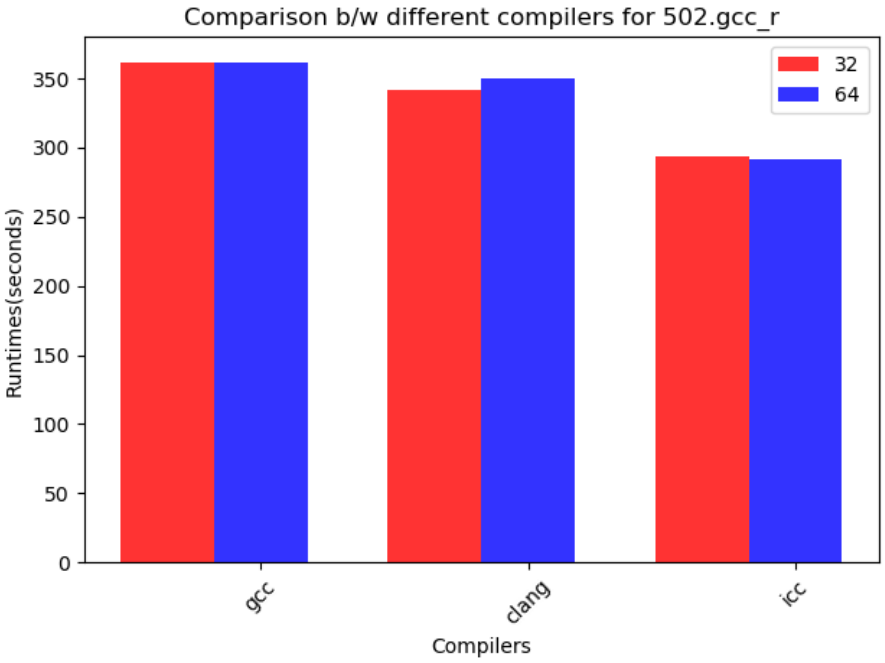

Figure 4

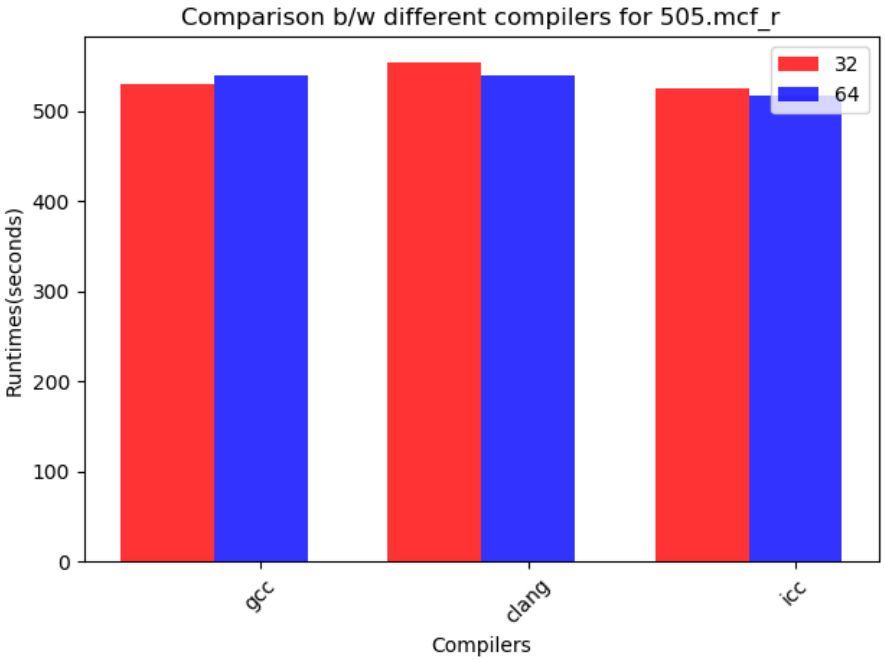

Figure 5

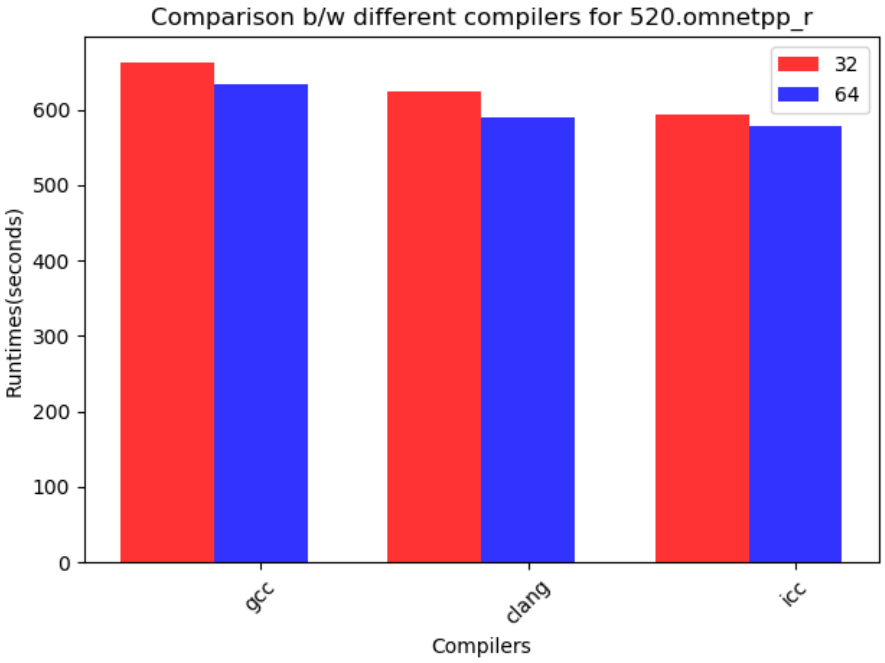

Figure 6

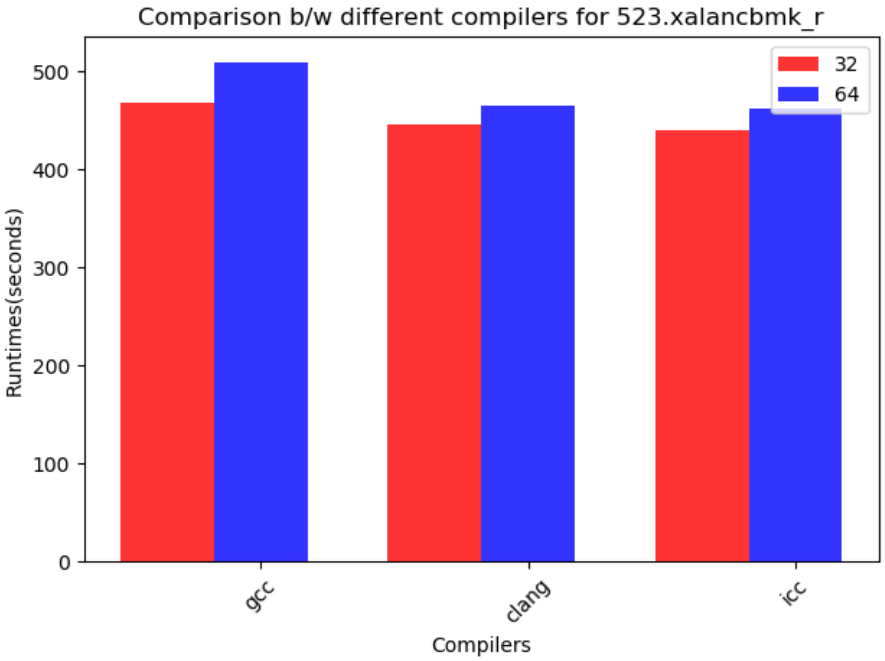

Figure 7

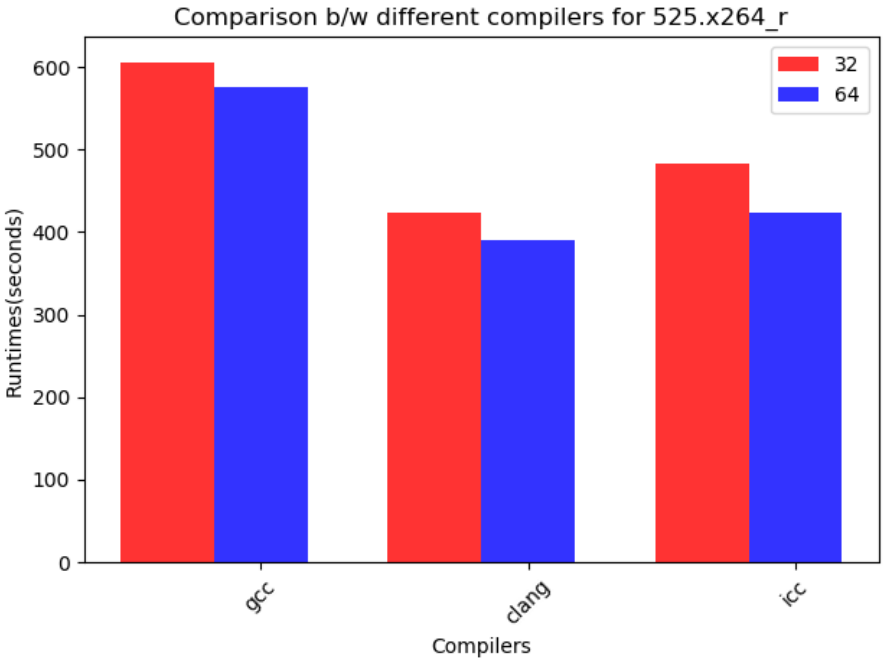

Figure 8

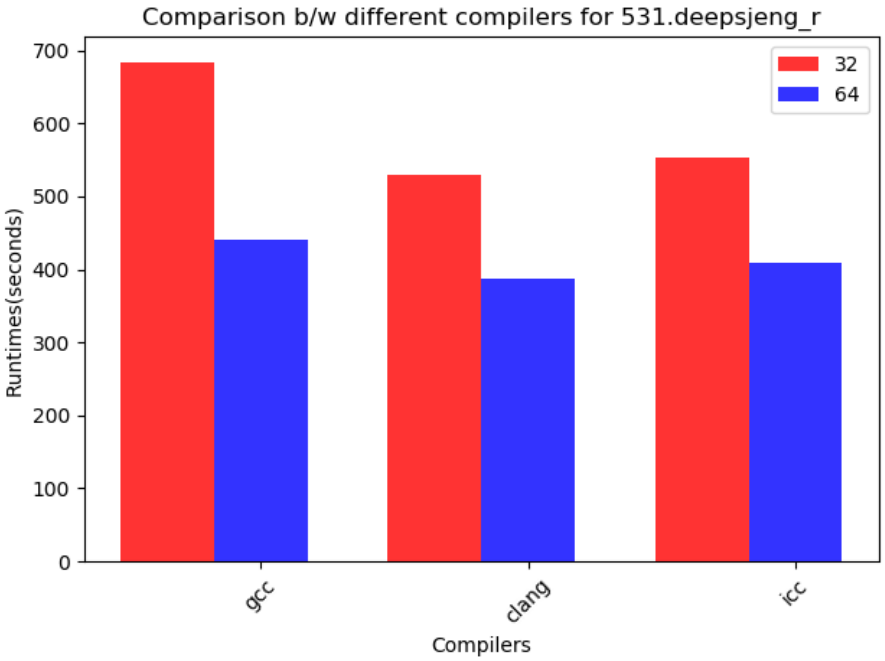

Figure 9

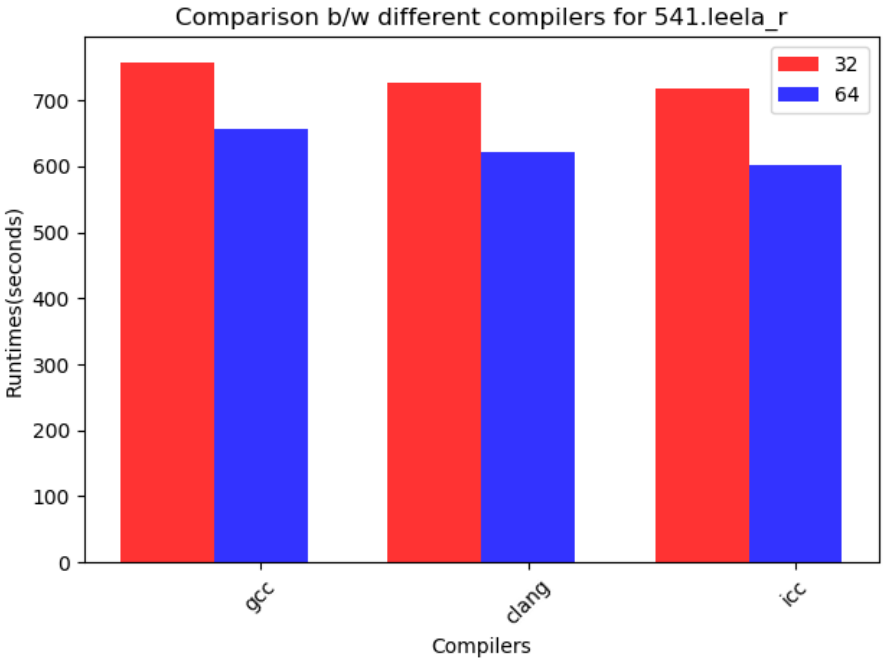

Figure 10

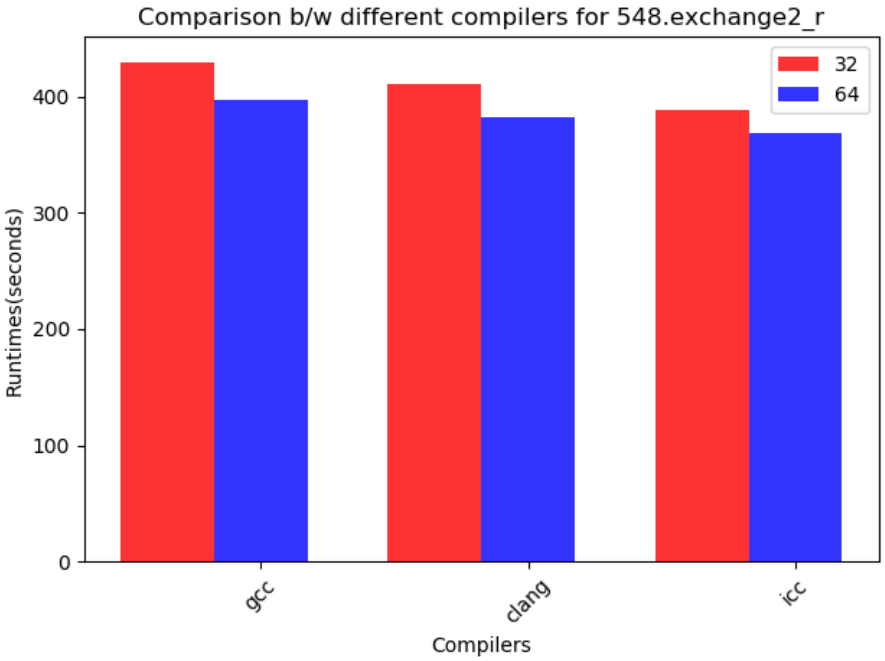

Figure 11

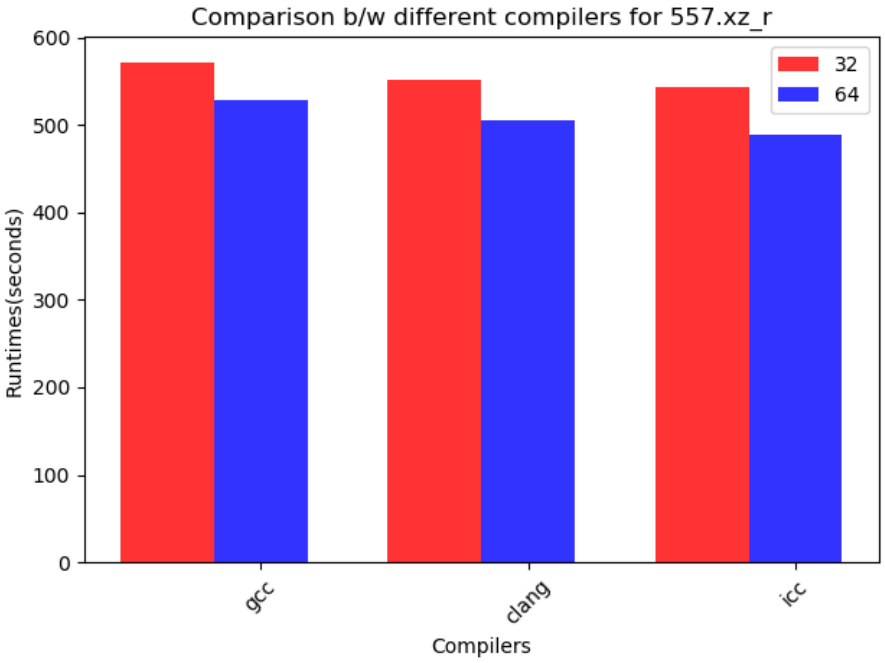

Figure 12

3 Optimization levels

The following graph shows the performances for different optimization levels in gcc.

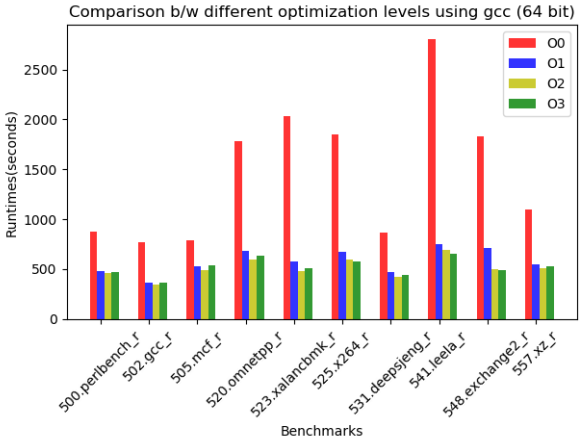

Figure 13: Comparison b/w different optimization levels in gcc

We see that -O1, -O2, -O3 are all better than -O0 (unoptimized) as to be expected. Also, the following rule is observed in most of the benchmarks -

$$-O2 > -O3 > -O1$$

This is to be expected. -O1 only does moderate optimization so as to not degrade the compile time too much whereas -O2 generates highly optimized code at the cost of compilation time. -O3 also does full optimization like -O2 but also use aggressive inlining of subprograms within a unit and attempts to vectorize loops. This maked the program not fit in the instruction cache and makes the execution time slower for -O3. For some benchmarks, -O3 outperforms -O2. This is probably because these benchmarks contain critical loops that actually benefit from the loop unrolling done by -O3.

4 Influence of benchmarks on results between compilers

It was observed that the benchmarks do affect the results between compilers.

1. 531.deepsjeng_r and 525.x264_r had clang outperforming even icc.
2. For 523.xalancbmk_r, the 32-bit executables were better than the 64-bit ones.
3. In 505.mcf_r, gcc performs slightly better than clang.