

Problem 1: A manufacturer develops an ISA extension which can dramatically improve the performance of certain benchmarks. The extension includes new instructions which work well with small integers. In what the manufacturer calls well-formed C programs the compiler will find all opportunities where the new instructions can be used and so the dramatic improvement will be realized. On other programs in which the new instructions could be used the compiler won't use them because it can't tell if the resulting machine code would be correct (perhaps because its not sure if values in registers would be small). In such cases the compiler will provide a message for the programmer indicating a list of regions in which there was the possibility of using the instructions. The programmer can then recompile with a special option indicating which of those regions the new instructions can safely be used in. The resulting code would be sped up.

Suppose this all works out very well for developers. They have no problems indicating which regions are safe for the new instructions and their resulting executables are fast and run correctly.

The manufacturer would like to run the SPECcpu2006 benchmarks on their new implementation. Most of the SPECcpu2006 benchmarks are not well formed.

(a) Why couldn't the compiler options (flags) for the SPEC run (base or peak) indicate the safe regions under a reasonable interpretation of the rules? In your answer refer to specific parts of the SPECcpu2006 run and reporting rules,

<http://www.spec.org/cpu2006/Docs/runrules.html>.

The rules state that compiler flags cannot use names. Strictly speaking, the list of regions provided by the compiler is probably not a list of names but what is being asked of the programmer is similar to what spec rule 2.1.1 forbids: the use of variable or subroutine names in optimization flags. Flags with variable or subroutine names might be used to tell the compiler to apply a dangerous optimization only to that code, such flags are probably forbidden because few programmers would make the effort to use them properly (especially when using the flags inappropriately would lead to incorrect execution). The compiler described in the problem provides a list of regions and asks the programmer to make the same kind of decision, one that would require familiarity with both the optimization and with the code. For that reason the safe regions flags could arguably be forbidden under SPEC rules.

(b) Keeping in mind the goals of the SPECcpu benchmarks argue either that the SPECcpu rules should be changed (perhaps for a future version of the benchmark) or argue that the rules should remain as they are.

Argument for changing the rules:

SPECcpu is supposed to show the performance potential of new systems, including CPU, memory, and compilers. By providing a short list of regions the programmer is not burdened with scouring the code for special optimization opportunities, and so many programmers would use it. In fact, many programmers do use it and it would make no sense to forbidding optimization techniques that are becoming common practice in production environments.

Solve the problems below. Then look at the solutions and assign yourself a grade.

Problem 2: Without looking at the solution solve Spring 2006 Midterm Exam Problem 1.

Problem 3: Without looking at the solution solve Spring 2006 Midterm Exam Problem 2.