Author's Reply

April 6, 2017

Overview

I appreciate your review and address the last question below. Note that I mark all the changes made to this revision to respond to your concern are marked in blue. I hope that these changes helped you better read the flow of the paper.

Reviewer

<< Reviewer's comments to the author(s) >>

The manuscript is well revised.

However, I have a question how to estimate each daemon's influence as time. The estimation is important for accuracy of your proposed method.

For acceptance, please clarify the estimation method and its validity.

If I understood your question right, each daemon's influence is translated to its respective cutoff, which I proposed and derived in this paper. For the detailed cutoffs, please refer to Tab. 1 and for computation process (estimation method in your term), the second paragraph in the right column on page 3. As mentioned in the paper, the cutoffs are used to drop any execution (sample) involving daemon executions over their respective cutoffs. This drop indicates that such an execution was *significantly influenced* by infrequent, long-running daemons and thus we cannot include that execution in the computation of the execution time of a given program.

To assess the validity of my protocol using the cutoffs, I used the SPEC CPU2006 benchmark suite and observed that the protocol outperformed the extant method using elapsed time, as exhibited in Tab. 3 on page 4 (as already indicated by the previous revision). Note also that I took out Fig. 1 and instead moved Fig. 5 in the former revision (now Fig. 1) upfront (on page 1) in this revision, to early draw the attention of the protocol's effectiveness on two real programs (insertion sort and matrix multiplication) with increasing workloads.

Last but not least, there seems a confusion from the used program-under-test (PUT) (e.g., PUT128 and PUT16384) for not only defining but also evaluating the protocol. (In fact, the PUT is also an arbitrary compute-bound program, so there's no problem with applying the protocol to that. But I acknowledge that the dual use of the PUT potentially confuses readers including you.) To unravel the confusion, I have redesigned the protocol, divided into the three major steps, as shown in Fig. 2 on page 2. In the new design, I talk about the PUT, only in the context of protocol definition (Step B), but do not use it for evaluation. As described above, the evaluation was performed on those two real-world programs and the SPEC CPU benchmarks.

If I'm misunderstood, please kindly let me know. I'll re-address. Many thanks for your great comments.