APPENDIX

In our experiments for RQ1, we first generate testing programs and optimization sequences for CTOS and all baselines. Thus, the testing period does not include the time to obtain testing programs and optimization sequences for CTOS and the baselines. This may be threats to the validity of CTOS since this may cause unfair comparisons between CTOS and the baselines. For example, CTOS takes 3 to 6 hours to generate testing programs and optimizations, while we can quickly generate random testing programs and optimization sequences. To investigate the potential impact of the time to obtain testing programs and optimization sequences on the results for RQ1, we analyse the detected bugs within the first 80 hours out of the testing period 90 hours. That is, we assume that 10 hours are used to obtain testing programs and optimization sequences for CTOS.

Table 1 shows the results for the 10 runs of CTOS. From Table 1, we can see that the majority of bugs can be detected by CTOS within the first 80 hours. Particularly, CTOS can detect 11.7 bugs on average within 80 hours, while it is 13.1 bugs within the testing period 90 hours. Although the detected bugs within 80 hours are less than those in 90 hours, the results also outperforms all the baselines (see Table 2 in the paper). For example, compared with CTOS(*RP*+*RS*) that detects 9.8 bugs on average within 90 hours, CTOS also find 19.39% more bugs within 80 hours. Also, the time to obtain testing programs and optimization sequences is less compared with the total testing period. Thus, we believe that the results for RQ1 cannot be affected dramatically by the time to obtain testing programs and optimization sequences.

TABLE 1
Results of CTOS within the first 80 hours.

ID.	TP.	Crash	WC.	Inv. IR	Perf.	CGB	Total bugs
1	92	7	3	2	1	0	13
2	100	6	4	1	1	0	12
3	97	10	3	0	1	0	14
4	94	6	3	1	1	0	11
5	102	7	3	0	1	0	11
6	101	7	2	0	1	0	10
7	104	7	2	1	1	0	11
8	107	7	2	1	1	0	11
9	90	6	4	2	1	0	13
10	97	7	3	0	1.0	0	11
Average	98.4	7.0	2.9	0.8	1.0	0	11.7

TP.: Testing Programs, Inv. IR: Invalid IR, WC.: Wrong Code, CGB.: Code Generator Bug.