

## Editorial: Focus Section on Program Debugging

Software systems today are large and complex. At the same time, the time to market is extremely short because of competition. As a result, program debugging for real-life systems is very difficult. In general, the debugging process consists of three tasks, namely, fault localization, fault repair, and retesting. In particular, fault localization is generally considered to be the most challenging. It is recognized as time-consuming and tedious if conducted manually. On the other hand, formal methods suffer from scalability problems, and static techniques are imprecise. Automatic statistical fault localization techniques are regarded as the most promising option. They compare passed and failed executions of a faulty program and produce a suspiciousness ranking of program entities (such as statements or predicates). Developers may then follow up with the list sequentially to identify program faults. Unfortunately, although a large number of statistical fault localization techniques are available, they have not reached the maturity to pinpoint accurately and precisely the locations of faults. Also, the recording and replaying of passed and failed executions, as well as fault repair without introducing new bugs remain unresolved issues. Furthermore, researchers often make unrealistic assumptions, and software subjects under study do not necessarily reflect the fault characteristics of large industrial applications. There is plenty of room for improvement.

The 2nd International Workshop on Program Debugging (IWPD 2011) was a full day workshop held in conjunction with the 35th Annual International Computer Software and Applications Conference (COMPSAC 2011) in Munich, Germany in July 2011. It serves as a platform for researchers and practitioners to exchange ideas, present new advancements, and identify further challenges in program debugging. It brings to light the latest challenges and advances in research and practice in program debugging, with a special emphasis on methodology, technology, and environment. Two keynote speeches were given by internationally renowned researchers — T.Y. Chen of Swinburne University of Technology, Australia and W.K. Chan of City University of Hong Kong, Hong Kong. There were also sessions for paper presentations and panel discussions.

We shortlisted three papers from the workshop and invited the authors to submit an extended version to *Software: Practice and Experience*. Two papers were accepted for this focus section after going through two rounds of rigorous reviews involving two to three anonymous reviewers for each article. Both accepted papers address the important area of statistical fault location.

The first paper, entitled “In Quest of the Science in Statistical Fault Localization” by W. K. Chan and Yan Cai, is an extended version of the keynote speech delivered by the first author in IWPD 2011. A vital element in research is to know the shortcomings of the current state of the art. In this paper, the authors conduct a critical review of existing work on statistical fault localization (including their own), highlight misconceptions and unnecessary assumptions, and provide remedial measures to rectify such malpractices. The authors point out that a lot of current research in statistical fault localization does not consider coincidental correctness, which means that the execution of a faulty statement may not necessarily lead to a program failure, even though this important concept has been known to software testers for decades. Also, existing fault localization techniques compare the similarities and dissimilarities between passed and failed executions to locate faults. These similarity coefficients estimate the probability that a particular program entity causes a failure, but ignore the noise caused by other entities. The authors point out the importance of a noise-reduction mechanism for the similarity coefficients. Another issue is that existing researchers often assume that they are dealing with large samples, where the central limit theorem applies. Empirical studies by the authors show that this assumption is often invalid. It is unrealistic to expect the availability of execution profiles with thousands of test verdicts for the average programs. A developer needs to debug a program even if a small number of failures have been revealed. When the number of samples is small, nonparametric statistical techniques should be applied. The authors conclude the paper by giving an insightful summary of the challenges in statistical fault localization that may benefit researchers in software engineering and related software areas.

The second paper is entitled “A Consensus-based Strategy to Improve the Quality of Fault Localization” by Vidroha Debroy and W. Eric Wong. Quite a number of statistical fault localization techniques have been proposed. Each of them claims to be superior to others in one aspect or another using different data sets. There is, however, no single technique that is definitely better than others in all aspects. In this paper, the authors put forward an integrated approach to address the issue. Rather than proposing yet another new technique that captures the more promising features of existing techniques, the authors propose a consensus-based strategy, which combines the rankings of several techniques. Using the Borda method, a consolidated ranking is produced by integrating various statement rankings that result from individual techniques. The scale of the proposed approach can be easily extended or retracted because new fault localization techniques can be added by the

inclusion of their rankings, or existing techniques can be excluded by the removal of their rankings. Also, since different techniques operate on the same input data set, the overhead of the consensus is minimal. The overall ranking can be determined in linear time. The effectiveness of the consensus-based approach has been validated using three popular fault localization techniques (Tarantula, Ochiai, and H3) on the Siemens suite of programs as well as the Ant, grep, gzip, make, and space programs. The empirical study shows that the performance of the proposed approach is close to the best results of the techniques under study.

Finally, I would like to thank Professor Nigel Horspool and Professor Andy Wellings, Editors of *Software: Practice and Experience*, for kindly agreeing to publish this focus section.

T.H. TSE  
*Department of Computer Science*  
*The University of Hong Kong*  
*Pokfulam, Hong Kong*