Checking Safety Properties of Sequential Programs via Static Analysis

Thomas Ball
Microsoft Research, Redmond, USA

At Microsoft, we now regularly apply a new generation of static analysis tools that can automatically identify serious defects in programs. These tools examine millions of lines of code every day, long before the software is released for general use. With these tools, we catch more defects earlier in the software process, enabling Microsoft to deliver more reliable systems. A number of these tools have been released for general use through Microsoft's Visual Studio integrated development environment as well as freely available development kits. In my lectures I will address the question: "How does one design and implement a static analysis tool chain to help people effectively address a software reliability problem?" In particular, I will identify a set of basic techniques that have proven very useful in constructing static analysis tools and have shown their worth through numerous applications. Experience with these techniques suggests we are approaching an exciting time when more people can contribute to the design and implementation of static analysis tools.

References

1. ASTREE:
   Blanchet, B., Cousot, P., Cousot, R., Feret, J., Mauborgne, L., Miné, A., Monniaux, D., and Rival, X. *A Static Analyzer for Large Safety-Critical Software.* In Proc of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation, ACM, New York, NY; pp 196–207; 2003. `http://doi.acm.org/10.1145/781131.781153`

2. ESPX:
   Hackett, B., Das, M., Wang, D., and Yang, Z. *Modular Checking for Buffer Overflows in the Large.* In Proc of the 28th International Conference on Software Engineering, ACM, New York, NY; pp 232–241; 2006. `http://doi.acm.org/10.1145/1134285.1134319`

3. FINDBUGS:
   Hovemeyer, D., Spacco, J., and Pugh, W. *Evaluating and Tuning a Static Analysis to find Null Pointer Bugs.* In Proc of the 6th ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering, ACM, New York, NY; pp 13–19; 2005.
   `http://doi.acm.org/10.1145/1108792.1108798`

4. STATIC DRIVER VERIFIER:
   Ball, T., Bounimova, E., Cook, B., Levin, V., Lichtenberg, J., McGarvey, C., Ondrusek, B., Rajamani, S. K., and Ustuner, A. *Thorough Static Analysis of Device Drivers.* In Proc of the 1st ACM Sigops/Eurosys European Conference on Computer Systems 2006, ACM, New York, NY; pp 73–85; 2006. `http://doi.acm.org/10.1145/1217935.1217943`

5. TYPESTATE VERIFICATION FOR JAVA:
   Fink, S., Yahav, E., Dor, N., Ramalingam, G., and Geay, E. *Effective Typestate Verification in the Presence of Aliasing.* In Proc of the 2006 International Symposium on Software Testing and Analysis. ACM, New York, NY; pp 133–144; 2006.
   `http://doi.acm.org/10.1145/1146238.1146254`