ITL BULLETIN FOR MAY 2016

COMBINATORIAL TESTING FOR CYBERSECURITY AND RELIABILITY

Rick Kuhn, Raghu Kacker, Larry Feldman,[1] and Greg Witte,[1] Editors
Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
U.S. Department of Commerce

Combinatorial testing (CT) is a proven method for more effective software testing at lower cost. The key insight underlying combinatorial testing's effectiveness resulted from a series of studies by NIST from 1999 to 2004. NIST research showed that most software bugs and failures are caused by one or two parameters, with progressively fewer by three or more. This finding, referred to as the *interaction rule*, has important implications for software testing because it means that testing parameter combinations can provide more efficient fault detection than conventional methods. New algorithms compressing combinations into a small number of tests have made CT practical for industrial use, making it possible to do better testing at lower cost.

Background

Software developers often notice an interesting – though not surprising – phenomenon: when the number of system users increases significantly, components that had previously operated correctly will suddenly fail or display errors. For example, if there are many new transactions or accounts, some are likely to contain combinations of values that have not been seen before. Some of these rare combinations trigger faults that escaped previous testing and extensive use. Combinatorial testing can help detect problems early in the testing life cycle. [1]

Because system failures often result from the interaction of conditions that might be innocuous individually, testing combinations of parameter values can be nearly as effective as testing all possible combinations for many types of applications. Implementing this form of testing requires a *covering array*, a matrix that includes all *t*-way combinations of values for some specified interaction level, *t*. As noted, most failures are caused by one or two parameters; empirical data show that the number of failures triggered by three or more parameters is small, and no failures discovered by NIST or other researchers have involved more than six parameters. As a result of this phenomenon, covering arrays that include all 3-way to 6-way combinations can provide strong testing.

[1] Larry Feldman and Greg Witte are Guest Researchers from G2, Inc.

Case Studies and Practical Examples

Combinatorial testing is an extension of the established field of statistical Design of Experiments (DoE), endorsed by the Department of Defense and used by commercial firms with demonstrated success. One of the first organizations to make extensive use of this type of testing for software and systems is the U.S. Air Force test group at Eglin Air Force Base, and a growing number of organizations have reported cost and time savings using the approach. Most of its use has been in computer software and hardware, defense/aerospace, telecommunications, and financial applications, although users can be found in nearly every industry.

The most extensive, publicly documented analysis to date of an industrial application of combinatorial testing has been an eight-project study by Lockheed Martin, a large U.S. defense contractor. Lockheed Martin has a Cooperative Research and Development Agreement (CRADA) with NIST [2]. CRADAs are one of the ways in which NIST conducts joint research with U.S. industry, allowing federal laboratories to work with U.S. industry, providing flexibility in structuring projects and protecting industry-proprietary information and research results. Lockheed Martin and NIST entered into the agreement in 2010 to better understand applicability and effectiveness of the combinatorial testing approach for software testing to improve quality, safety and reliability of U.S. products and systems. Of particular interest was understanding the challenges in introducing a new approach for software testing in a large U.S. corporation. Projects evaluated the viability of the concept for achieving the following goals:

- Test process improvement in a variety of domains: system, software, and hardware testing;
- Make tests more effective in finding problems; and
- Reduce the cost of testing, or at least reduce test life-cycle cost by finding fewer errors late in development or in the field.

The pilot projects demonstrated that it was practical to incorporate the new methods, with testing cost reduction of approximately 20 %, with 20 % to 50 % improvement in test coverage.

Recent projects reported by other organizations include:

- A team of developers in banking and financial services reported, "Combinatorial Testing (CT) approach has greatly helped our projects from different domains to optimize testing effort without compromising on testing quality. We were able to achieve breakthrough business results. CT-based freeware tools such as All Pairs & ACTS [Automated Combinatorial Testing for Software] are of great help for testing professionals to optimize effort and reduce learning curve." [5]

- An application of combinatorial testing to automotive electronics by a large manufacturer found the method to be a significant advance over traditional test approaches. The authors reported

that they “observed a remarkable reduction in time to identify more likely defects and increased probability in detecting of less probable defects. This is especially crucial when the market release of product approaches.”[8]

- An “industry proof-of-concept demonstration used combinatorial testing approach to automate parts of the unit and integration testing of a highly complex avionics system. The goal was to see if it might cost-effectively reduce rework by reducing the number of software defects escaping into system test. The test would also determine if CT was adequately accurate, rigorous, thorough, and scalable. Overcoming scalability issues required moderate effort, but in general it was effective – e.g., generating 47,040 test cases (input vectors, expected outputs) in 75 seconds, executing and analyzing them in 2.6 hours. It subsequently detected all seeded defects, and achieved nearly 100 % structural coverage.”[3]

- Combinatorial methods provided an 84X efficiency improvement for testing conformance to a new video coding standard, HEVC (High Efficiency Video Coding). The original conformance testing spec included 1 000 182 coding tree units, but using a 3-way covering array plus two more tests, they provided better coverage with 13 712 units. Coverage was measured in 'syntax elements,' requiring more than 90 % coverage of the syntax elements. The authors say, "In the proposed method, the SE [syntax elements] coverage normalized by the number of CTUs [coding tree units] is 84 times higher compared to that in the HEVC conformance test suite. This means that we can verify the HEVC decoders 84 times faster with the test bitstream set obtained by the proposed method, compared to the HEVC conformance test suite." [4]

Tools

NIST has developed two research tools for combinatorial testing:

- Automated Combinatorial Testing for Software (ACTS), developed by NIST and the University of Texas Arlington, generates tests. The ACTS distribution includes both a command line version for integration with shell scripts or other testing tools, and an interactive version with an easy-to-use graphical user interface. It has strong support for setting constraints between parameters, an essential feature for real-world testing. The test arrays produced by ACTS are among the smallest of any known algorithms, allowing for highly efficient testing [6].

- Combinatorial Coverage Measurement (CCM), developed by NIST and the Centro Nacional de Metrologia of Mexico, for determining the combinatorial coverage of any test suite (not necessarily made using combinatorial methods) [7]. CCM computes measures of combinatorial coverage that can be used in evaluating the degree of *t*-way coverage of any test suite,

regardless of whether it was initially constructed for combinatorial coverage. CCM has an intuitive user interface, support for constraints using the same syntax as ACTS, and produces detailed reports on coverage distribution.

Future Directions

An extensive body of experience from both industry and government shows that combinatorial testing is highly effective, producing significant improvements in the cost/benefit ratio for software assurance. Future research will improve integration of this method with industrial practice, focusing on two themes: (1) development of supporting materials; and (2) evolution of the combinatorial approach and tools. NIST will continue to work with the community to:

- Develop additional educational materials and guidance including case studies and industry experience reports;
- Enhance CT tools. A recent project with Carnegie Mellon University developed tools for easily creating and editing input models, to define parameters, values, and relationships among them. The input model is then automatically processed using ACTS to generate a covering array of tests; and
- Adapt CT methods for use with established testing infrastructures. Because established or regulated organizational test processes may be difficult to change, the CCM tool has been developed to make it possible to extend tests created with traditional methods to provide combinatorial testing.

Additional Resources

[1] D. R. Kuhn, R. N. Kacker, and Y. Lei, *Practical Combinatorial Testing*, National Institute of Standards and Technology (NIST) Special Publication (SP) 800-142, October 2010, 82 pp. http://dx.doi.org/10.6028/NIST.SP.800-142.

[2] J. D. Hagar, T. L. Wissink, D. R. Kuhn, and R. N. Kacker, "Introducing combinatorial testing in a large organization," *Computer (IEEE)*, vol. 48, no. 4 (April 2015), pp. 64-72. http://dx.doi.org/10.1109/MC.2015.114.

[3] R. Bartholomew, "An Industry Proof-of-Concept Demonstration of Automated Combinatorial Test," *25th Annual IEEE Software Technology Conference*, Salt Lake City, Utah, April 8-10, 2013.

[4] D. Hong and S.-I. Chae, "Efficient test bitstream generation method for verification of HEVC decoders," *18th IEEE International Symposium on Consumer Electronics (ISCE 2014)*, JeJu Island, South Korea, June 22-25, 2014, 2 pp. http://dx.doi.org/10.1109/ISCE.2014.6884404.

[5] M. Mehta and R. Philip, "Applications of Combinatorial Testing Methods for Breakthrough Results in Software Testing," *2nd International Workshop on Combinatorial Testing (IWCT 2013)*, Luxembourg, March 22, 2013, in *Proceedings, IEEE Sixth International Conference on Software, Testing,*

Verification and Validation Workshops, Piscataway, New Jersey: IEEE Computer Society, 2013, pp. 348-351. http://dx.doi.org/10.1109/ICSTW.2013.46.

[6] Y. Lei, R. Kacker, D. R. Kuhn, V. Okun, and J. Lawrence, "IPOG: a General Strategy for *t*-way Software Testing," *14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS 2007)*, Tucson, Arizona, United States, March 26-29, 2007, pp. 549–556. http://dx.doi.org/10.1109/ECBS.2007.47.

[7] D. R. Kuhn, I. Dominguez, R. N. Kacker, and Y. Lei, "Combinatorial Coverage Measurement Concepts and Applications," *2nd International Workshop on Combinatorial Testing (IWCT 2013)*, Luxembourg, March 22, 2013, in *Proceedings, IEEE Sixth International Conference on Software, Testing, Verification and Validation Workshops*, Piscataway, New Jersey: IEEE Computer Society, 2013, pp. 352-361. http://dx.doi.org/10.1109/ICSTW.2013.77.

[8] S. Züfle and V. Krishnamoorthy, "A process for nonfunctional combinatorial testing: Selection of parameter values from a nondiscrete domain space," *2015 IEEE 8th International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, Graz, Austria, April 13-17, 2015, 4 pp. http://dx.doi.org/10.1109/ICSTW.2015.7107437.

ITL Bulletin Publisher: Elizabeth B. Lennon
Information Technology Laboratory
National Institute of Standards and Technology
elizabeth.lennon@nist.gov