ACTS: A Combinatorial Test Generation Tool

[†]Linbin Yu, [†]Yu Lei, [§]Raghu N. Kacker, [§]D. Richard Kuhn

[†]University of Texas at Arlington
[§]National Institute of Standards and Technology

Outline

- ACTS in Retrospect
  - The project, the NIST study, higher-strength testing, user distribution, release timeline, team
- Major Features
  - Test generation, test set extension, coverage verification
- Looking Forward
  - Input parameter modeling, industrial applications, advantages of CT, publication trend

The NIST ACTS Project

csrc.nist.gov/Projects/automated-combinatorial-testing-for-software

An official website of the United States government Here's how you know

NIST

Search CSRC | CSRC MENU

Information Technology Laboratory

COMPUTER SECURITY RESOURCE CENTER

NIST COMPUTER SECURITY RESOURCE CENTER CSRC

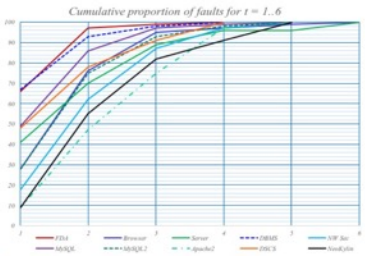
PROJECTS

Combinatorial Testing

Overview

Combinatorial methods can reduce costs for software testing, and *have significant applications in software engineering:*

- **Combinatorial or *t*-way testing** is a proven method for more effective testing at lower cost. The key insight underlying its 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, which means that combinatorial testing can provide more efficient fault detection than conventional methods. Multiple studies have shown fault detection equal to exhaustive testing with a 20X to 700X reduction in test set size. New algorithms compressing combinations into a small number of tests have made this method practical for industrial use, providing better testing at lower cost. See articles on high assurance software testing or security and reliability.



- **Assured autonomy and AI systems testing:** Input space coverage measurements are needed in life-critical assurance and verification of autonomous systems, because current methods for assurance of safety critical systems rely on measures of structural coverage, which do not apply to many autonomous systems. Combinatorial methods, including a theorem relating measures of input space coverage, offer a better approach for autonomous system verification and AI assurance.

NIST Tech reports on coverage measurement for assured autonomy:

- **Ordered t-way Combinations for Testing State-based Systems**
- **Combination Frequency Differencing**
- **Combinatorial Coverage Difference Measurement**

ACTS - award winning tool for combinatorial testing, used in thousands of organizations worldwide

PROJECT LINKS

Overview

FAQs

ADDITIONAL PAGES

Quick start

Downloadable Tools

Combinatorial Methods in Testing

- Why do Combinatorial Testing?
- Event Sequence Testing
- Oracle-free Testing and Test Automation
- Case Studies

Assured autonomy

- Explainable AI, Verification, and Validation
- Rule-based Expert Systems and Formal Methods
- AI and Assured Autonomy Papers
- Assured Autonomy - briefings and videos
- Case studies

Input space measurement for autonomy and testing

- Input Space Coverage Measurement
- Coverage examples
- Case studies

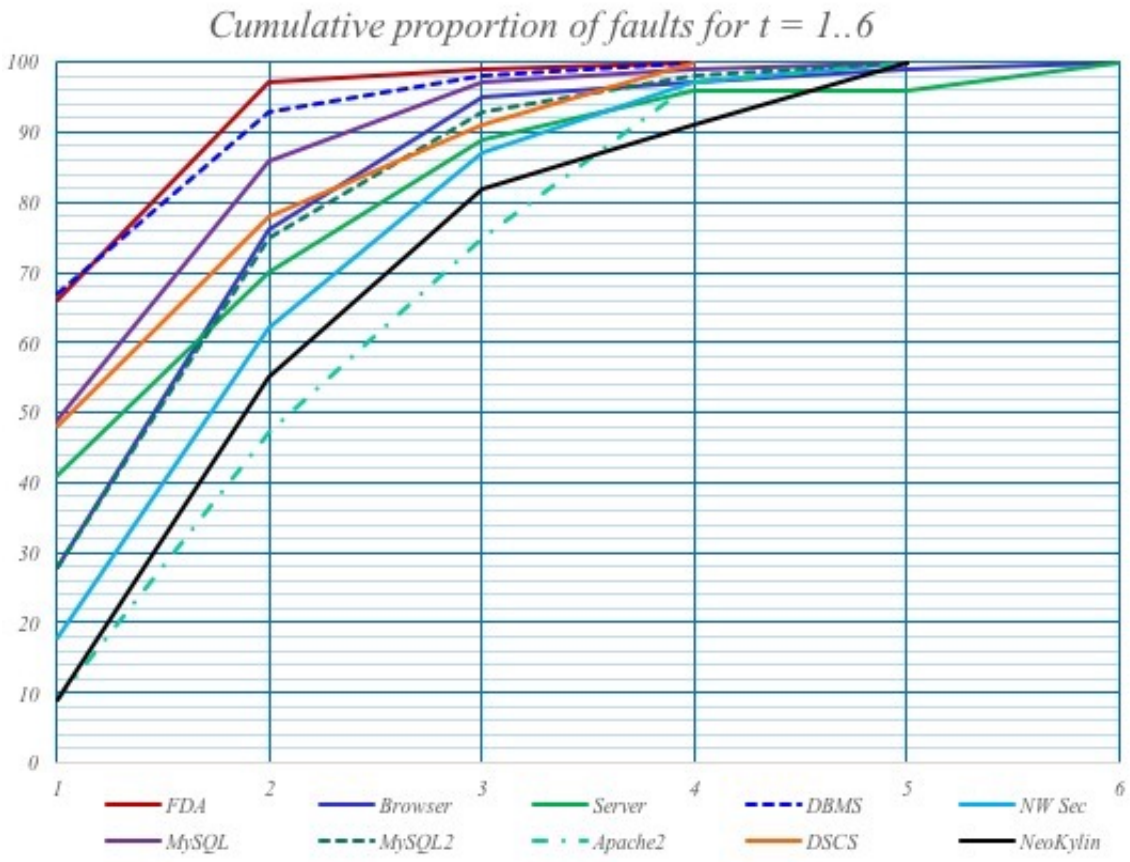
URL: https://csrc.nist.gov/acts

Foundational Papers

- D.R. Kuhn and M.J. Reilly. An investigation of the applicability of design of experiments to software testing. *27th Annual NASA Goddard/IEEE Software Engineering Workshop*, 2002. (Citations: 440)
- D.R. Kuhn, D.R. Wallace, A.M. Gallo, Jr. Software Fault Interactions and Implications for Software Testing, *IEEE Transactions on Software Engineering*, 2004. (Citations: 980)
- Y. Lei and K. C. Tai. In-parameter-order: A test generation strategy for pairwise testing." *IEEE International High-Assurance Systems Engineering Symposium*, 1998. (Citations: 503)
- 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'07), 2007. (Citations: 447)

Citations are accessed from Google Scholar on April 11, 2023

The NIST Study



URL: https://csrc.nist.gov/acts

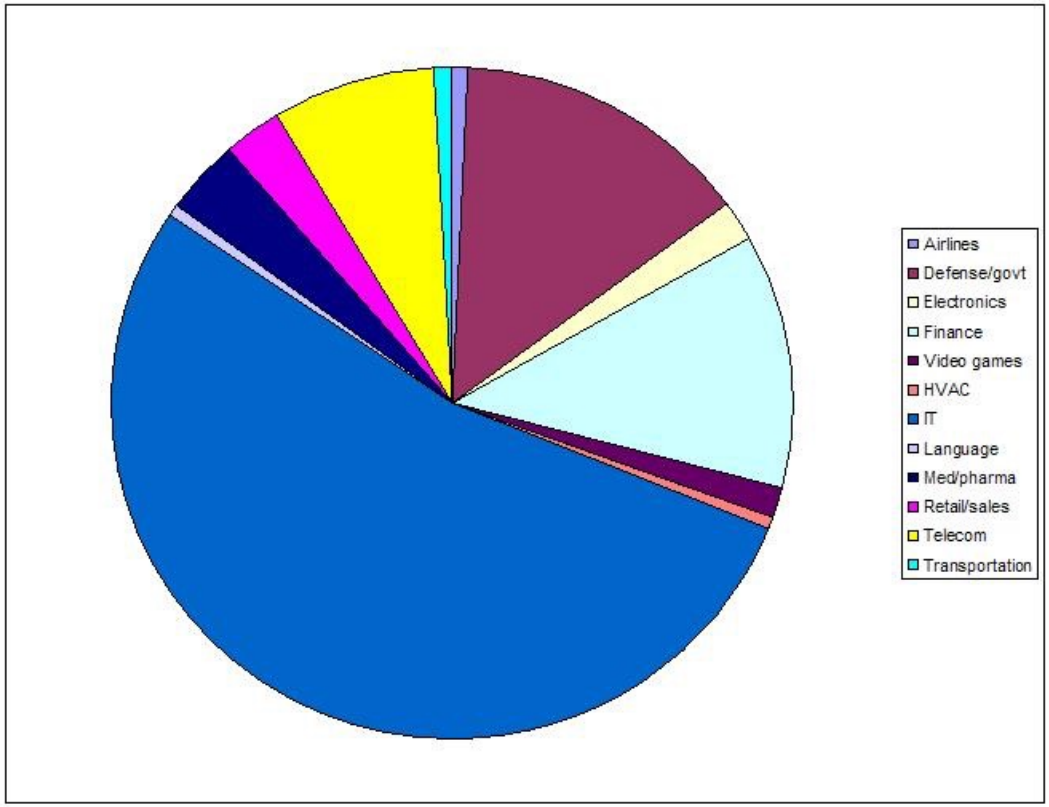
Higher-Strength Testing

- Interaction Rule: Most software bugs and failures are caused by one or two parameters, with progressively fewer by three or more.
- Higher-strength testing is needed to detect faults that involve more than two factors.
  - Start with pairwise testing, and then increase test strength as allowed by the available resources.
- In practice, a test set should be checked, and made complete if resources are available, at least for pairwise coverage.

The NIST ACTS Tool

- A research tool for combinatorial test generation.
  - written in Java
  - has been downloaded by more than **4000** individual and/or institutions

User Distribution



* The chart is made based on data up to 2015.

Release Timeline

- 2008: FireEye Beta
- 2010: ACTS 1.0 (FireEye changed to ACTS)
- 2012: ACTS 2.0 (Constraint support)
- 2017: ACTS 3.0 (Optimization)
- 2019: ACTS 3.2 (latest version)

The Tool Team

- PIs
  - Rick Kuhn (NIST)
  - Raghu Kacker (NIST)
  - Yu Lei (UTA)
- Students
  - Linbin Yu (Graduated in 2013, Facebook)
  - Feng Duan (Graduated in 2020, Jiangxi Univ. of Finance and Economics)
  - Tony Opara (Graduated in 2010, Salesforce)
  - Kiran Karnam (Graduated in 2010, Nvidia)

Outline

- ACTS in Retrospect
  - The project, the NIST study, higher-strength testing, user distribution, release timeline, team
- Major Features
  - Test generation, test set extension, coverage verification
- Looking Forward
  - Input parameter modeling, industrial applications, advantages of CT, publication trend

T-Way Test Generation

- Support from 1-way up to 6-way coverage
  - Base-choice coverage for 1-way testing
- Constraint handling
  - support Boolean, relational and arithmetic operators
  - e.g.: P1 > P2 + P3, ! (A || B), X > 3 => Z = false
- Mixed Strength
  - allows different parameter groups to be created and covered with different strengths

An Example 2-Way Test Set

Consider a system that has three parameters, each having two values 0 and 1.

P1	P2	P3
0	0	0
0	1	1
1	0	1
1	1	0

Pick ANY two parameters, all combinations 00, 01, 10, 11 are covered.

The IPOG Strategy

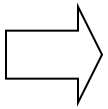
- First generate a t-way test set for the first t parameters, then for the first t + 1 parameters, and so on
- A pairwise test set for the first k parameters is built by extending the test set for the first k – 1 parameters
  - Horizontal growth: Extend each existing test case by adding one value of the new parameter
  - Vertical growth: Adds new tests, if necessary

Illustration of IPOG

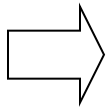
Three parameters A, B, and C

A: A1, A2; B: B1, B2; C: C1, C2 and C3

A	B
A1	B1
A1	B2
A2	B1
A2	B2



A	B	C
A1	B1	C1
A1	B2	C2
A2	B1	C3
A2	B2	C1



A	B	C
A1	B1	C1
A1	B2	C2
A2	B1	C3
A2	B2	C1
A2	B1	C2
A1	B2	C3

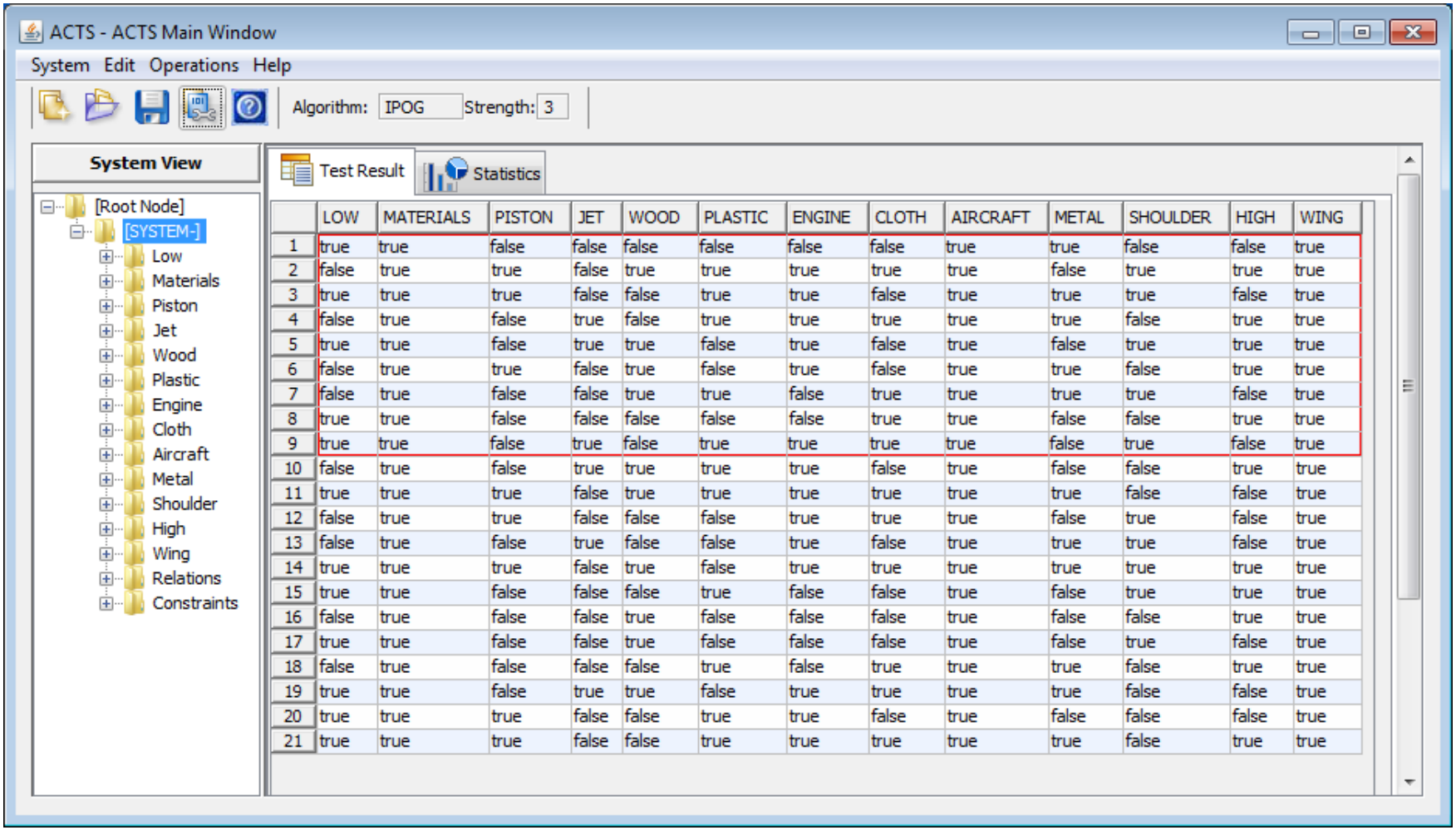
Horizontal Growth

Vertical Growth

Test Set Extension

- Build a t-way test set by extending an existing, incomplete test set
  - Can save earlier effort that has already been spent in the testing process
- An incomplete test set may be given by the user or generated by ACTS, e.g., when there were a smaller number of parameters and/or values or a lower strength

Test Set Extension (2)

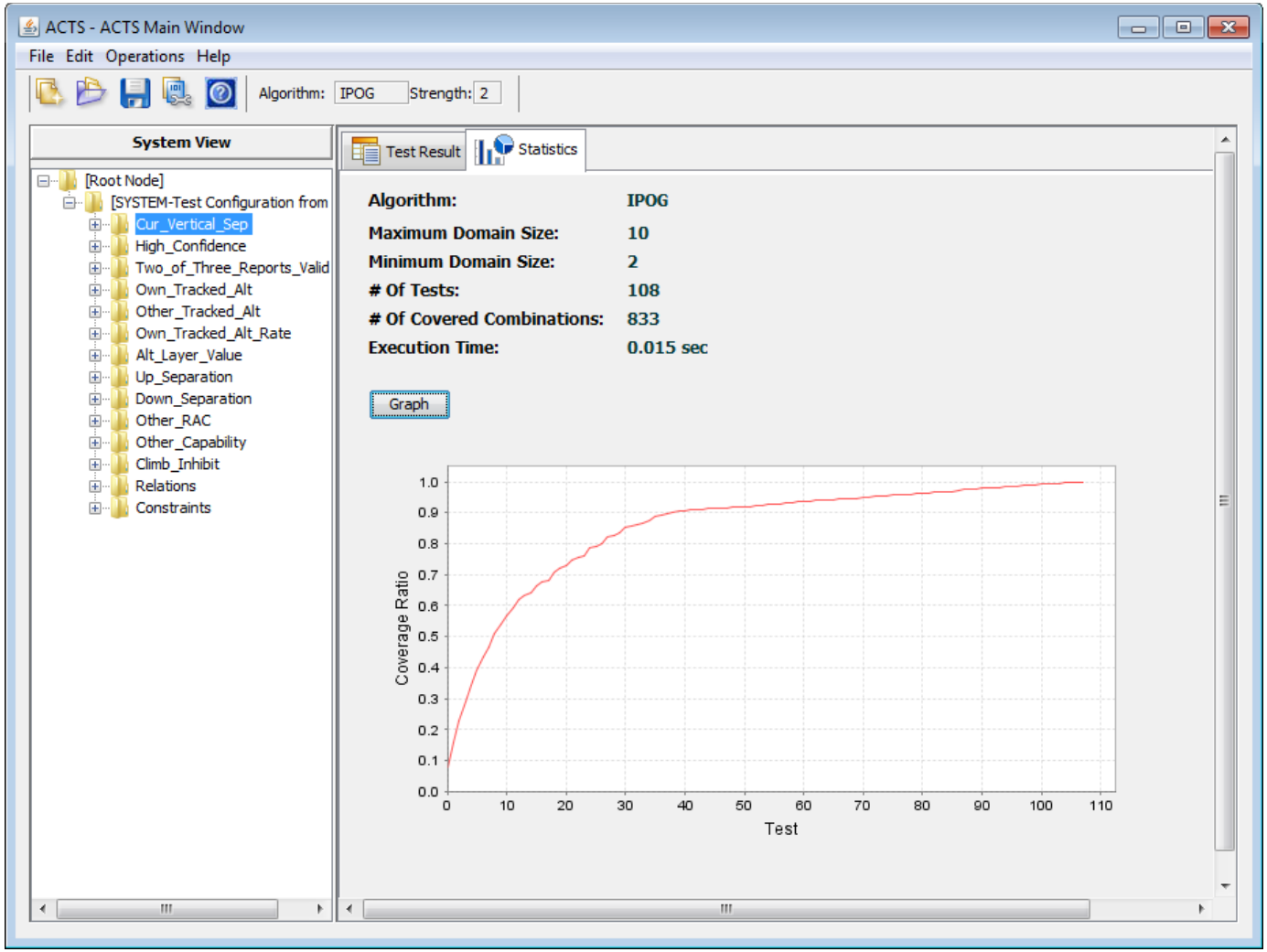


	LOW	MATERIALS	PISTON	JET	WOOD	PLASTIC	ENGINE	CLOTH	AIRCRAFT	METAL	SHOULDER	HIGH	WING
1	true	true	false	false	false	false	false	false	true	true	false	false	true
2	false	true	true	false	true	true	true	true	true	false	true	true	true
3	true	true	true	false	false	true	true	false	true	true	true	false	true
4	false	true	false	true	false	true	true	true	true	true	false	true	true
5	true	true	false	true	true	false	true	false	true	false	true	true	true
6	false	true	true	false	true	false	true	false	true	true	false	true	true
7	false	true	false	false	true	true	false	true	true	true	true	false	true
8	true	true	false	false	false	false	false	true	true	false	false	true	true
9	true	true	false	true	false	true	true	true	true	false	true	false	true
10	false	true	false	true	true	true	true	false	true	false	false	true	true
11	true	true	true	false	true	true	true	true	true	true	false	false	true
12	false	true	true	false	false	false	true	true	true	false	true	false	true
13	false	true	false	true	false	false	true	false	true	true	true	false	true
14	true	true	true	false	true	false	true	true	true	true	true	true	true
15	true	true	false	false	false	true	false	false	true	false	true	true	true
16	false	true	false	false	true	false	false	false	true	false	false	true	true
17	true	true	false	false	true	false	false	false	true	false	true	false	true
18	false	true	false	false	false	true	false	true	true	true	false	true	true
19	true	true	false	true	true	false	true	true	true	true	false	false	true
20	true	true	true	false	false	true	true	false	true	false	false	false	true
21	true	true	true	false	false	true	true	true	true	true	false	true	true

Coverage Verification

- Verify whether a test set satisfies t-way coverage or not, and plot the curve of accumulative coverage
- Implemented independently from the test generation process

Coverage Graph



Outline

- ACTS in Retrospect
  - The project, the NIST study, higher-strength testing, user distribution, release timeline, team
- Major Features
  - Test generation, test set extension, coverage verification
- Looking Forward
  - Input parameter modeling, industrial applications, advantages of CT, publication trend

Input Parameter Modeling

- Can we automatically identify parameters, values, relations, and constraints?
  - Analyze requirements or source code using machine learning techniques like NLP
  - How to get a large amount of training data?

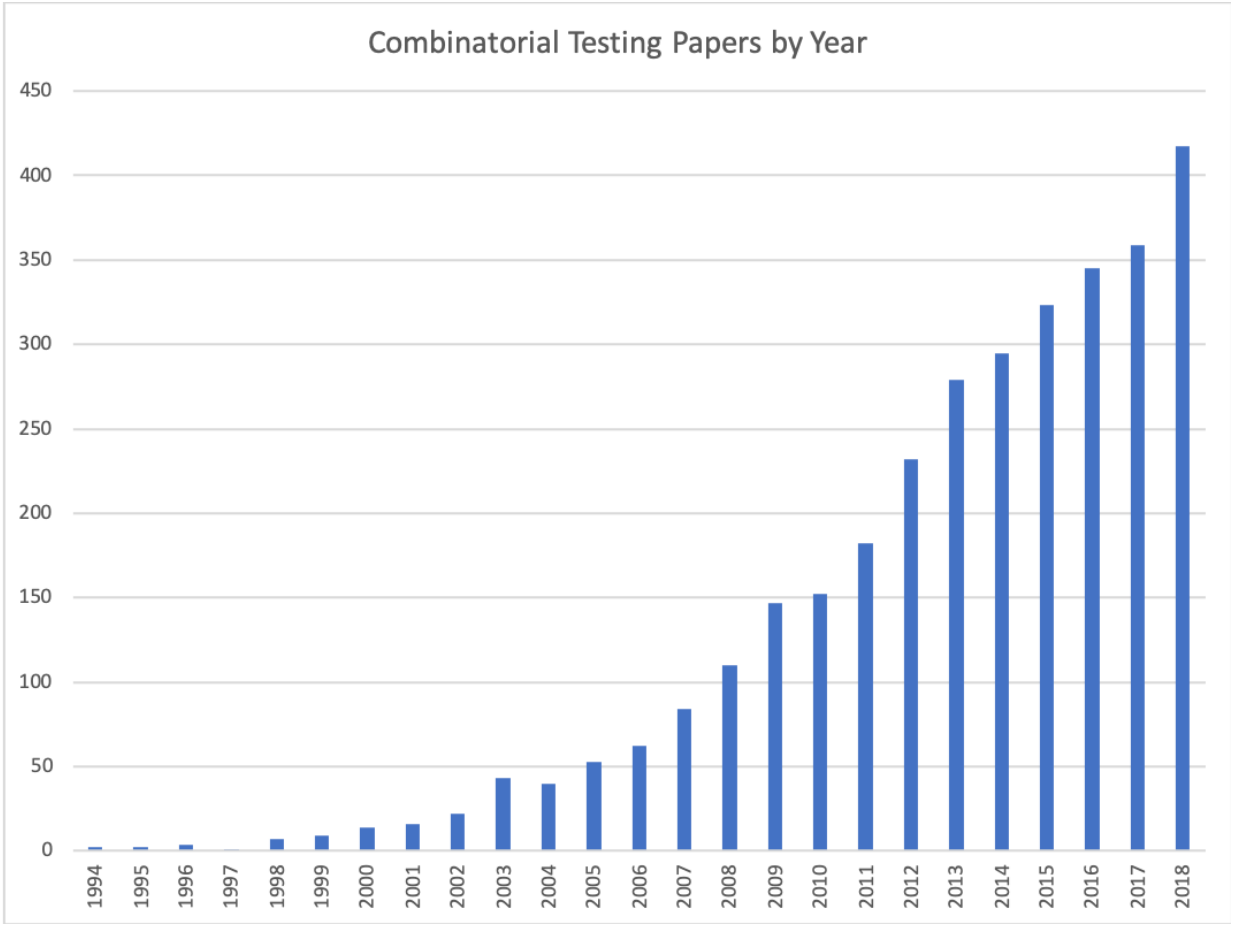
Industrial Applications

- Rigorous use of CT in industrial settings
  - How effective is CT in detecting faults in real-world applications, especially in terms of comparative evaluation?
  - What are the major challenges, technical or otherwise, for an organization to adopt CT?
  - What are the best practices?

Advantages of CT

- General applicability: The interaction rule is fundamental and general, making CT applicable in numerous domains.
- Rigorous foundation: A t-way test set is a well-defined, mathematic structure (i.e., covering array).
- Effectiveness: CT can significantly reduce the number of tests while preserving fault detection effectiveness.
- Ease of Use: Given an IPM (a light-weight specification), the test generation process is fully automated.

Publication Trend



URL: https://csrc.nist.gov/acts

Resources

- The NIST ACTS project webpage is available at:
  - https://csrc.nist.gov/acts
  - Or just google “NIST ACTS”
- To request a copy of ACTS, please contact:
  - Rick Kuhn: kuhn@nist.gov

Thank you