

Combinatorial Testing in Five Minutes

Robert V. Binder

August 4, 2020

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Notices

Copyright 2020 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

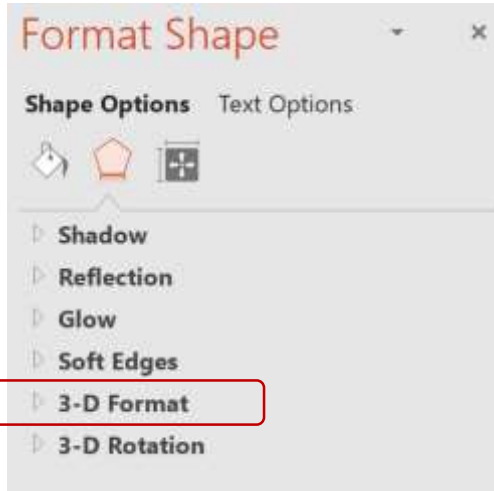
NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

DM20-0623

Example of a combinatorial testing challenge

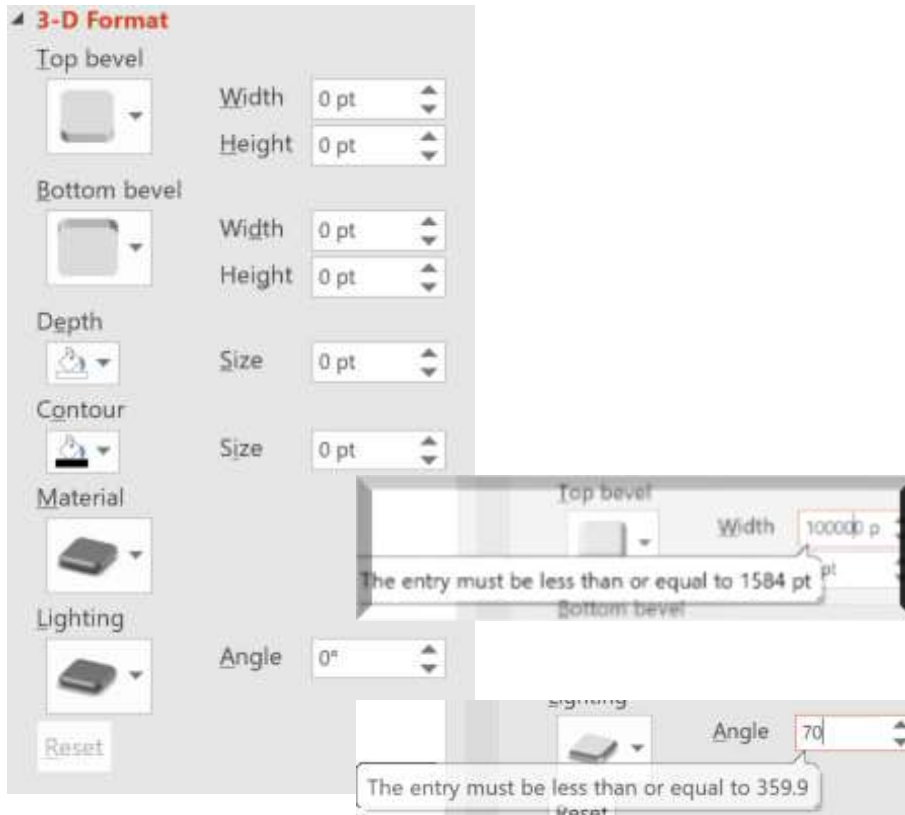


Consider the 3-D Format panel found in PowerPoint 2019

Suppose we want to test it

- How many tests are needed?
- Which ones?
- What settings/values to use?

Example of a combinatorial testing challenge



The controls accept the following

- Top Bevel: 13 types
- Bottom Bevel: 13 types
- Depth: 256 standard colors
- Contour: 256 standard colors
- Material: 11 settings
- Lighting: 15 settings
- Widths: 0..1584
- Angles: 0..359.9
- Sizes: 0..1584

How many combinations are there?

We need to choose specific inputs (“levels”) for each control (a “factor”). For this example, I use boundary values, all values of enumerated types, at least one randomly selected in-range value and one fuzzed value.

Factor	Level Values	# of Levels
Height, Width, Size	-1,0,1, RandInRange(), 1584, 1585, Fuzz()	7
Angle	-1, -0.1, 0, 0.1, 1.0, RandInRange(), 359.9, 360, Fuzz()	9
Top Bevel, Bottom Bevel	B1..B13	13
Depth and Contour:	None, white, black, red, green, RandInRange()	6
Material	M1..M11	11
Lighting	L1..L15	15

All possible combinations of level values:

$$7 \times 7 \times 7 \times 9 \times 13 \times 13 \times 6 \times 6 \times 11 \times 15 = 3,098,915,820$$

How to select a practical and comprehensive set of tests?

3 billion is too many.

A few ad hoc and/or random selections is not enough.

How about testing every possible pair of levels at least once?

How many unique tests would that take?

How to select a practical and comprehensive set of tests?

All pairs of levels can be covered with 202 test cases

ACTS - ACTS Main Window
System Edit Operations Help
Algorithm: IPOG Strength: 2

System View

- Root Node
- SYSTEM_3D_Paramet
- Height
- Width
- Size
- Angle
- Toplevel
- Bottomlevel
- Material
- Lighting
- Depth
- Contour

Test Result

	HEIGHT	WIDTH	SIZE	ANGLE	TOPLEVEL	BOTTOMLEVEL	MATERIAL	LIGHTING	DEPTH	CONTOUR
1	0	0	0	-0.009	1	2	2	1	White	White
2	1	1	1	0.0	2	3	3	1	Black	Black
3	R	R	R	0.009	3	4	4	1	Red	Red
4	1584	1584	1584	0.1	4	5	5	1	Green	Green
5	1585	1585	1585	R	5	6	6	1	R	R
6	F	F	F	359.8	6	7	7	1	None	None
7	-1	-1	-1	359.9	7	8	8	1	White	Black
8	0	1	R	360.0	8	9	9	1	Green	R
9	1	R	1584	-1.0	9	10	10	1	R	None
10	R	1584	1585	-0.009	10	11	11	1	None	Black
11	1584	1585	F	0.0	11	12	1	1	Red	White
12	1585	F	-1	0.009	12	13	3	1	Black	Green
13	F	-1	0	0.1	13	1	3	1	Red	R
14	-1	1584	F	R	1	3	4	1	Green	Red
15	0	1585	-1	359.8	2	4	5	2	R	Green
16	1	0	1	359.9	3	5	6	2	None	White
17	R	-1	1584	360.0	4	6	7	2	Black	White
18	1584	1	0	-1.0	5	7	8	2	White	Red
19	1585	R	1	-0.009	6	8	9	2	Red	None
20	F	F	1585	0.0	7	9	10	2	Green	Red
21	-1	0	R	0.009	8	10	11	2	R	Black
22	0	F	1	0.1	9	11	1	2	White	R
23	1	-1	R	R	10	12	2	2	None	Green
24	R	0	1584	359.8	11	13	3	2	White	None
25	1584	1	1585	359.9	12	1	4	2	R	None
26	1585	R	F	360.0	13	2	5	2	None	Black
27	F	1584	-1	-1.0	1	4	6	3	Black	R
28	-1	1585	0	-0.009	2	5	7	3	Green	None
29	0	R	1584	0.0	3	6	8	3	None	Green
30	1	1585	1585	0.009	4	7	9	3	White	White
31	R	0	F	0.1	5	8	10	3	Black	Green
32	1584	-1	1	R	6	9	11	3	R	Red
33	1585	1	R	359.8	7	10	1	3	Red	White
34	F	R	1584	359.9	8	11	2	3	Red	Black
35	-1	F	1585	360.0	9	12	3	3	Red	Green
36	0	-1	F	-1.0	10	13	4	3	Green	White
37	1	F	R	-0.009	11	1	5	3	Black	Red
38	R	-1	0	0.0	12	2	6	3	Green	R
39	1584	0	-1	0.009	13	3	7	3	None	R
40	1585	-1	1585	359.8	1	5	8	4	Black	None
41	F	1584	R	359.9	2	6	9	4	White	Red
42	-1	1585	1	360.0	3	7	10	4	R	Black

Impact of Adding Factors and Levels

There are two flavors of width and height (top and bottom bevel).

There are two flavors of size (depth and contour).

The first model used only one of each.

What is the effect of adding 3 more factors?

- Bottom Bevel Height
- Bottom Bevel Width
- Contour Size

The screenshot shows the ACTS software interface. The 'System View' pane on the left displays a hierarchical tree structure starting with '[Root Node]', followed by '[SYSTEM-3D-Format]' (highlighted in blue), and '[SYSTEM-3D-Format_Cop]'. Below these are several factors: Height, Width, Size, Angle, TopBevel, BottomBevel, Material, Lighting, Depth, Contour, BBheight, BBwidth, and ContourSize. The right pane shows 'Test Result' and 'Statistics' tabs. The 'Statistics' tab displays the following data:

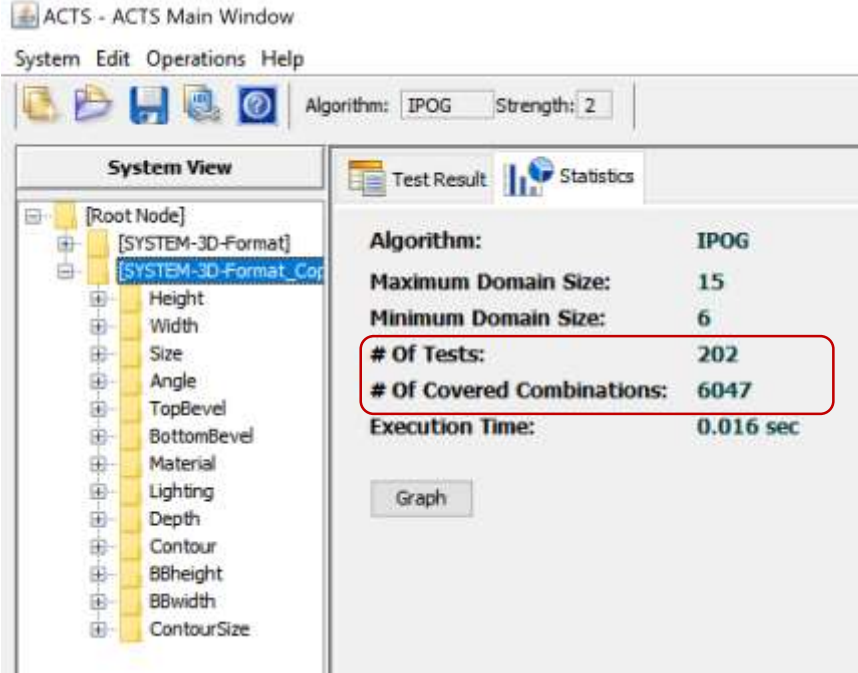
Algorithm:	IPOG
Maximum Domain Size:	15
Minimum Domain Size:	6
# Of Tests:	202
# Of Covered Combinations:	3926
Execution Time:	0.032 sec

The '# Of Tests' and '# Of Covered Combinations' values are highlighted with a red box. A 'Graph' button is visible at the bottom of the statistics pane.

Baseline Model

Impact of Adding Factors and Levels

No increase in the number of tests!



The screenshot shows the ACTS Main Window. The System View pane on the left displays a tree structure of factors and levels. The right pane shows the Test Result and Statistics for the selected system. The statistics are as follows:

Parameter	Value
Algorithm:	IPOG
Maximum Domain Size:	15
Minimum Domain Size:	6
# Of Tests:	202
# Of Covered Combinations:	6047
Execution Time:	0.016 sec

A red box highlights the # Of Tests and # Of Covered Combinations values.

Model with 3 more Factors

Considerations

There are many ways to select level values besides those shown here.

The generated test cases typically do not include the expected result of the test - the tester must develop expected results or judge the response.

ACTS and other tools support constraints to automatically exclude infeasible combinations.

ACTS and other tools provide generic data export (csv, xls, etc.) that can be used for automated data-driven testing.

Besides selecting inputs for test cases, combinatorial design can generate:

- Configuration combos to be tested (OS versions, RAM, CPU, etc.)
- Test environment combos (temperature, operational mode, traffic volume, time of day, etc.)

Can be used at any scope (unit, integration, system) for any kind of interface (API, GUI, etc.), manual or automated execution, and for functional, regression, or performance testing.

Resources

ACTS tool from NIST

<https://www.nist.gov/programs-projects/automated-combinatorial-testing-software-acts>

Experience Reports about using combinatorial Testing

<http://www.pairwise.org/results.asp>

<https://csrc.nist.gov/CSRC/media/Presentations/Introducing-Combinatorial-Testing-in-a-Large-Organ/images-media/poster-iwct14-lm2.pdf>

<https://app.hexawise.com/Combinatorial-Software-Testing-Case-Studies-IEEE-Computer-Kuhn-Kacker-Lei-Hunter.pdf>

<https://docs.hexawise.com/en/articles/2016946-does-pairwise-testing-really-work-evidence-data-and-case-studies>

Collection of many resources

<http://www.pairwise.org/index.asp>

Contact Information

Organization

Carnegie Mellon University
Software Engineering Institute
Software Solutions Division

Robert V. Binder

rvbinder@sei.cmu.edu

+1 412-268-1549

U.S. Mail

Software Engineering Institute
Customer Relations
4500 Fifth Avenue
Pittsburgh, PA 15213-2612, USA

Web

www.sei.cmu.edu
www.sei.cmu.edu/contact.cfm

Customer Relations

Email: info@sei.cmu.edu
Telephone: +1 412-268-5800
SEI Phone: +1 412-268-5800
SEI Fax: +1 412-268-6257