Combinatorial Interaction Testing

Justyna Petke

Centre for Research in Evolution, Search and Testing
University College London

CREST

Using materials from:

http://cse.unl.edu/~citportal/

http://csrc.nist.gov/groups/SNS/acts/ftfi.html

History of Combinatorial Interaction Testing

Written around 300 BC the Bhagabati Sutra was one of the earliest books to feature a problem dealing with combinatorics.

* image taken from https://en.wikipedia.org/wiki/Vy%C4%81khy%C4%81praj%C3%B1apti#/media/File:Kalpasutra.jpg

History of Combinatorial Interaction Testing

Given 6 spices, how many ways were there to select combinations of one, two and three spices ?

$$\binom{6}{1} + \binom{6}{2} + \binom{6}{3}$$

$$= 6 + 15 + 20 = 41$$

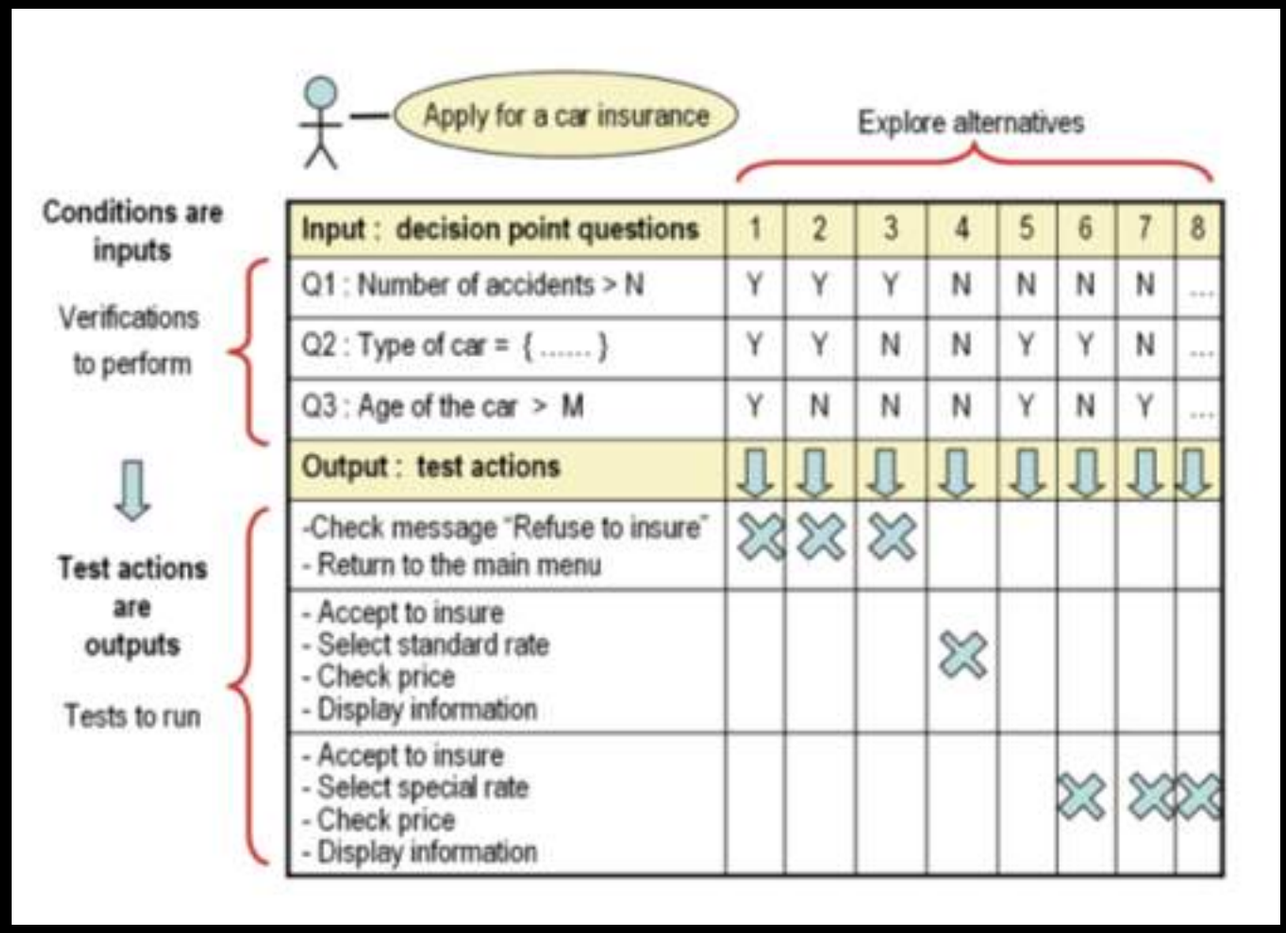
History of Combinatorial Interaction Testing

Ronald Fisher was an english statistician whose work included a book "Design of Experiements" (1935) that serves as the basis for many of the principals of CIT.

* image taken from https://en.wikipedia.org/wiki/The_Design_of_Experiments#/media/File:R._A._Fischer.jpg

History of Combinatorial Interaction Testing

Decision tables were initially developed by General Electric and Sutherland Corp. (independently in 1957) as a means to express product design logic, operational planning, management decision rules.



Input : decision point questions	1	2	3	4	5	6	7	8
Q1 : Number of accidents > N	Y	Y	Y	N	N	N	N	...
Q2 : Type of car = { ..... }	Y	Y	N	N	Y	Y	N	...
Q3 : Age of the car > M	Y	N	N	N	Y	N	Y	...
Output : test actions	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩
-Check message "Refuse to insure" - Return to the main menu	X	X	X					
- Accept to insure - Select standard rate - Check price - Display information				X				
- Accept to insure - Select special rate - Check price - Display information						X	X	X

* image taken from http://www.ibm.com/developerworks/rational/library/jun06/vauthier/

History of Combinatorial Interaction Testing

Latin Squares

An n x n array with n different characters occurring exactly once in each row and column.

* image taken from https://en.wikipedia.org/wiki/Latin_square#/media/File:Fisher-stainedglass-gonville-caius.jpg

History of Combinatorial Interaction Testing

In his 1985 paper Robert Mandl claimed that by using orthogonal Latin Squares you could get the equivalent of exhaustive testing at a fraction of the cost.

History of Combinatorial Interaction Testing

In 1992 R. Brownlie et al. published a paper in the AT&T technical journal on using Orthogonal Arrays to design tests in a real world setting.

1	1	1
2	2	1
1	2	2
2	1	2

History of Combinatorial Interaction Testing

Orthogonal Array

An N x k array A with entries from some set S with s levels, strength t within the range $0 \leq t \leq k$ and index λ where every N x t subarray of A contains each t-tuple based on S exactly λ times as a row.

1	1	1
2	2	1
1	2	2
2	1	2

History of Combinatorial Interaction Testing

Covering Array Tables for t=2,3,4,5,6

These tables are maintained by Charlie Colbourn on an irregular basis. Please report updates and corrections.

For given t and v, the table (t,k,v) gives the current best known upper bound on CAN(t,k,v), the smallest number of rows in a uniform covering array having k factors each with v levels, with coverage at strength t. Covering array numbers are reported for each k up to 20000 for strength two, 10000 for strengths three through six. At present, the authorities are not given with references.

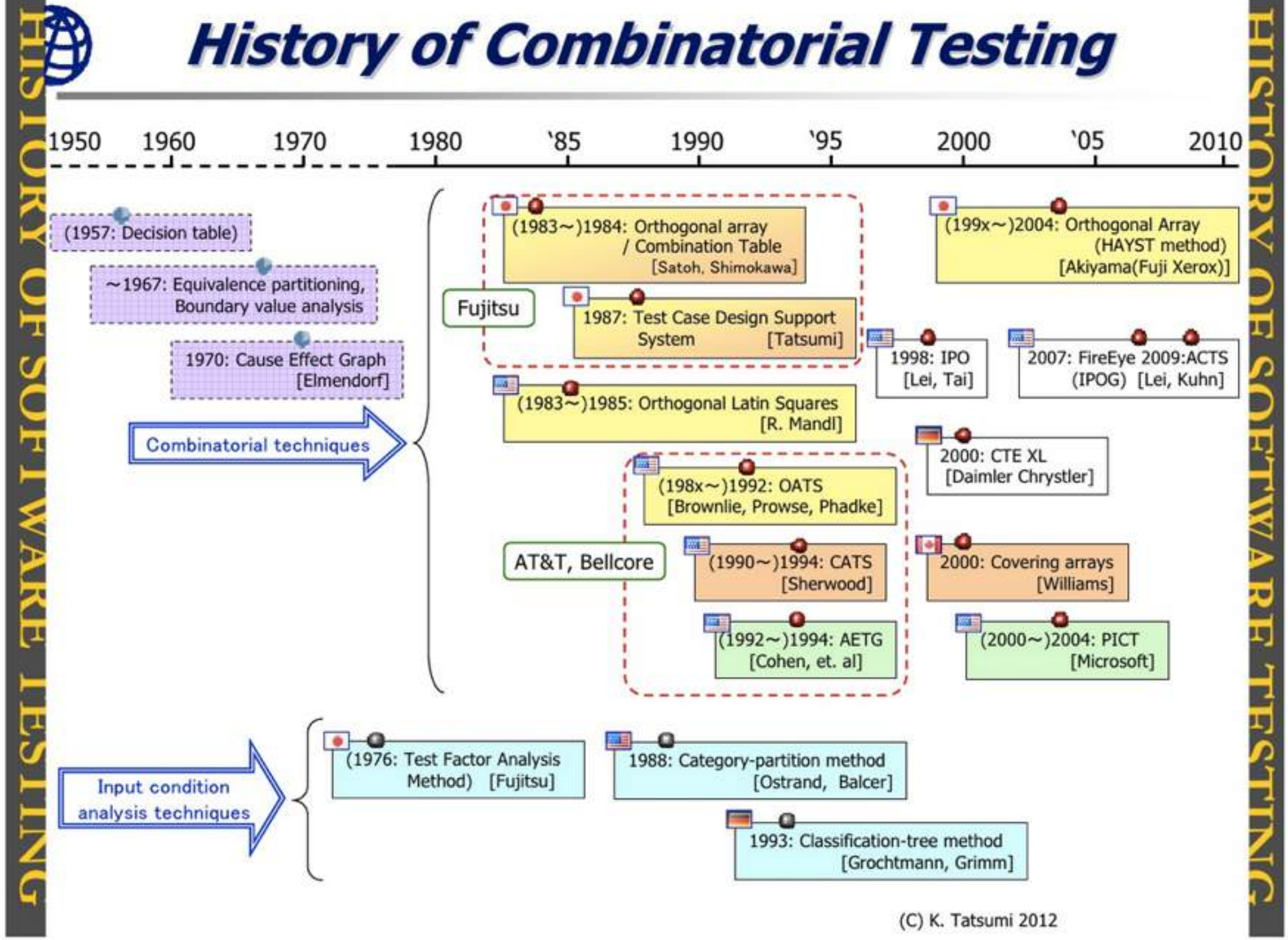
`Best known' means best reported in the literature, to me via email, or implied by a recursive construction. Sizes are reported when an explicit construction is known, not when a probabilistic argument guarantees existence. However, for certain values of v when t is 4, 5, or 6, a constructive conditional expectation algorithm yields better bounds than those implied by the direct and recursive methods -- in these cases, the accompanying graph shows two lines, of which the lower one shows the bounds from the conditional expectation method.

(2,k,2) (2,k,3) (2,k,4) (2,k,5) (2,k,6) (2,k,7) (2,k,8) (2,k,9) (2,k,10) (2,k,11) (2,k,12) (2,k,13) (2,k,14) (2,k,15) (2,k,16) (2,k,17) (2,k,18) (2,k,19) (2,k,20) (2,k,21) (2,k,22) (2,k,23) (2,k,24) (2,k,25)
(3,k,2) (3,k,3) (3,k,4) (3,k,5) (3,k,6) (3,k,7) (3,k,8) (3,k,9) (3,k,10) (3,k,11) (3,k,12) (3,k,13) (3,k,14) (3,k,15) (3,k,16) (3,k,17) (3,k,18) (3,k,19) (3,k,20) (3,k,21) (3,k,22) (3,k,23) (3,k,24) (3,k,25)
(4,k,2) (4,k,3) (4,k,4) (4,k,5) (4,k,6) (4,k,7) (4,k,8) (4,k,9) (4,k,10) (4,k,11) (4,k,12) (4,k,13) (4,k,14) (4,k,15) (4,k,16) (4,k,17) (4,k,18) (4,k,19) (4,k,20) (4,k,21) (4,k,22) (4,k,23) (4,k,24) (4,k,25)
(5,k,2) (5,k,3) (5,k,4) (5,k,5) (5,k,6) (5,k,7) (5,k,8) (5,k,9) (5,k,10) (5,k,11) (5,k,12) (5,k,13) (5,k,14) (5,k,15) (5,k,16) (5,k,17) (5,k,18) (5,k,19) (5,k,20) (5,k,21) (5,k,22) (5,k,23) (5,k,24) (5,k,25)
(6,k,2) (6,k,3) (6,k,4) (6,k,5) (6,k,6) (6,k,7) (6,k,8) (6,k,9) (6,k,10) (6,k,11) (6,k,12) (6,k,13) (6,k,14) (6,k,15) (6,k,16) (6,k,17) (6,k,18) (6,k,19) (6,k,20) (6,k,21) (6,k,22) (6,k,23) (6,k,24) (6,k,25)

If you are interested in explicit presentations of covering arrays, which are not necessarily the best known, a good place to start is at the NIST Covering Array Tables. Some explicit solutions are also available from Jose Torres Jimenez here -- click on Covering Arrays.

In 1994 the first paper on the AETG system was published by D.M. Cohen et al. It showed how earlier works in software testing could be improved by using Covering Arrays instead of Orthogonal Arrays.

History of Combinatorial Testing



Definition

"Combinatorial Interaction Testing (CIT)
is a black-box system testing technique
that samples inputs, configurations and parameters
and combines them in a systematic fashion."

Overview of CIT http://cse.unl.edu/~citportal/tutorials.php

Definition

Black-box testing is a method of software testing that tests program functionality without modifying it.

Definition

White-box testing is a method of software testing that tests internal program structures.

examples: branch/statement coverage testing

Example

Booking a flight from London:

Country
City
Date
Return Date

Parameters and Values

Booking a flight from London:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
	Madrid	21 Aug	26 Aug

Example

Booking a flight from London:

Spain	→ Barcelona	→ 20 Aug	→ 27 Aug
			→ 26 Aug
		→ 21 Aug	→ 27 Aug
			→ 26 Aug
	→ Madrid	→ 20 Aug	→ 27 Aug
			→ 26 Aug
		→ 21 Aug	→ 27 Aug
			→ 26 Aug

1 choice x 2 choices x 2 choices x 2 choices

Parameters and Values

Booking a flight from London:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
	Madrid	21 Aug	26 Aug

How many possible inputs?

Example

All possible parameter-value combinations:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Example

Test suite for the web form:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Example 2

Booking a flight from London-Heathrow:

Country	City	Date	Return Date
80 countries	184 cities	365 days	365 days

$\sim 2*10^9$ combinations !

Example 2

Booking a flight from London-Heathrow:

Country	City	Date	Return Date
80 countries	184 cities	365 days	365 days

Combinatorial Explosion problem

Combinatorial Interaction Testing

Problem: Testing all combinations is too expensive.

Solution: Test all interactions between any pair of parameters.

Combinatorial Interaction Testing

Problem: Testing all combinations is too expensive.

Solution: pairwise testing

Combinatorial Interaction Testing

A pairwise test suite covers all 2-way interactions between any 2 parameters.

Pairwise (2-way) testing is the most widely studied CIT technique.

Pairwise testing example

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Pairwise testing example

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Pairwise testing example

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Pairwise testing example

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Pairwise testing example

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Pairwise testing example

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Pairwise testing example

Pairwise (2-way) interaction test suite:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug

Combinatorial Interaction Testing

Problem: Testing all combinations is too expensive.

Solution: Test all interactions between any set of t parameters.

Combinatorial Interaction Testing

Problem: Testing all combinations is too expensive.

Solution: t-way testing

Combinatorial Interaction Testing

A CIT test suite covers all t-way interactions between any t parameters.

Such a test suite is also known as a covering array.

Covering Array

Definition: A covering array $CA(t, k, v)$ of size N is a table with N rows and k columns. Each field of CA contains a value in the range $0, \cdots, v-1$. CA has the following property: every combination of t values between any t parameters occurs in at least one row.

t is called the strength of a covering array.

Combinatorial Interaction Testing

CIT problem:

Find a minimal test suite that covers all t-way interactions.

$CAN(t, k, v)$ denotes the size of the smallest covering array.

Covering Array Tables

www.public.asu.edu/~ccolbou/src/tabby/catable.html

Covering Array Tables for t=2,3,4,5,6

These tables are maintained by Charlie Colbourn on an irregular basis. Please report updates and corrections.

For given t and v, the table (t,k,v) gives the current best known upper bound on CAN(t,k,v), the smallest number of rows in a uniform covering array having k factors each with v levels, with coverage at strength t. Covering array numbers are reported for each k up to 20000 for strength two, 10000 for strengths three through six. At present, the authorities are not given with references.

`Best known' means best reported in the literature, to me via email, or implied by a recursive construction. Sizes are reported when an explicit construction is known, not when a probabilistic argument guarantees existence. However, for certain values of v when t is 4, 5, or 6, a constructive conditional expectation algorithm yields better bounds than those implied by the direct and recursive methods -- in these cases, the accompanying graph shows two lines, of which the lower one shows the bounds from the conditional expectation method.

(2,k,2) (2,k,3) (2,k,4) (2,k,5) (2,k,6) (2,k,7) (2,k,8) (2,k,9) (2,k,10) (2,k,11) (2,k,12) (2,k,13) (2,k,14) (2,k,15) (2,k,16) (2,k,17) (2,k,18) (2,k,19) (2,k,20) (2,k,21) (2,k,22) (2,k,23) (2,k,24) (2,k,25)
(3,k,2) (3,k,3) (3,k,4) (3,k,5) (3,k,6) (3,k,7) (3,k,8) (3,k,9) (3,k,10) (3,k,11) (3,k,12) (3,k,13) (3,k,14) (3,k,15) (3,k,16) (3,k,17) (3,k,18) (3,k,19) (3,k,20) (3,k,21) (3,k,22) (3,k,23) (3,k,24) (3,k,25)
(4,k,2) (4,k,3) (4,k,4) (4,k,5) (4,k,6) (4,k,7) (4,k,8) (4,k,9) (4,k,10) (4,k,11) (4,k,12) (4,k,13) (4,k,14) (4,k,15) (4,k,16) (4,k,17) (4,k,18) (4,k,19) (4,k,20) (4,k,21) (4,k,22) (4,k,23) (4,k,24) (4,k,25)
(5,k,2) (5,k,3) (5,k,4) (5,k,5) (5,k,6) (5,k,7) (5,k,8) (5,k,9) (5,k,10) (5,k,11) (5,k,12) (5,k,13) (5,k,14) (5,k,15) (5,k,16) (5,k,17) (5,k,18) (5,k,19) (5,k,20) (5,k,21) (5,k,22) (5,k,23) (5,k,24) (5,k,25)
(6,k,2) (6,k,3) (6,k,4) (6,k,5) (6,k,6) (6,k,7) (6,k,8) (6,k,9) (6,k,10) (6,k,11) (6,k,12) (6,k,13) (6,k,14) (6,k,15) (6,k,16) (6,k,17) (6,k,18) (6,k,19) (6,k,20) (6,k,21) (6,k,22) (6,k,23) (6,k,24) (6,k,25)

If you are interested in explicit presentations of covering arrays, which are not necessarily the best known, a good place to start is at the NIST Covering Array Tables. Some explicit solutions are also available from Jose Torres Jimenez here -- click on Covering Arrays.

Pairwise testing example

Pairwise (2-way) interaction test suite:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug

Are all 3-way interactions covered?

3-way testing example

3-way interaction test suite:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Applications

Web Form Example

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
	Madrid	21 Aug	26 Aug

Web Browser Example

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install	Remember downloads
Allow Restrict Block	Yes No	Allow Restrict Block	Yes No	Yes No

Car Model Example

Automated Driving Controller	Collision Avoidance Braking	Parallel Parking	Lateral Range Finder	Forward Range Finder
Included	StandardAvoidance	Included	Included	Included
None	EnhancedAvoidance	None	None	None
	None			

Cellphone Example

MMS	WLAN	Bluetooth	MP3	Camera
Included None	Included None	Included None	Included None	Black&White Colour None

Applications

Web Forms

Web Browsers

Automotive Industry

Cellphone Industry

.. and many other configurable systems

Fault detection

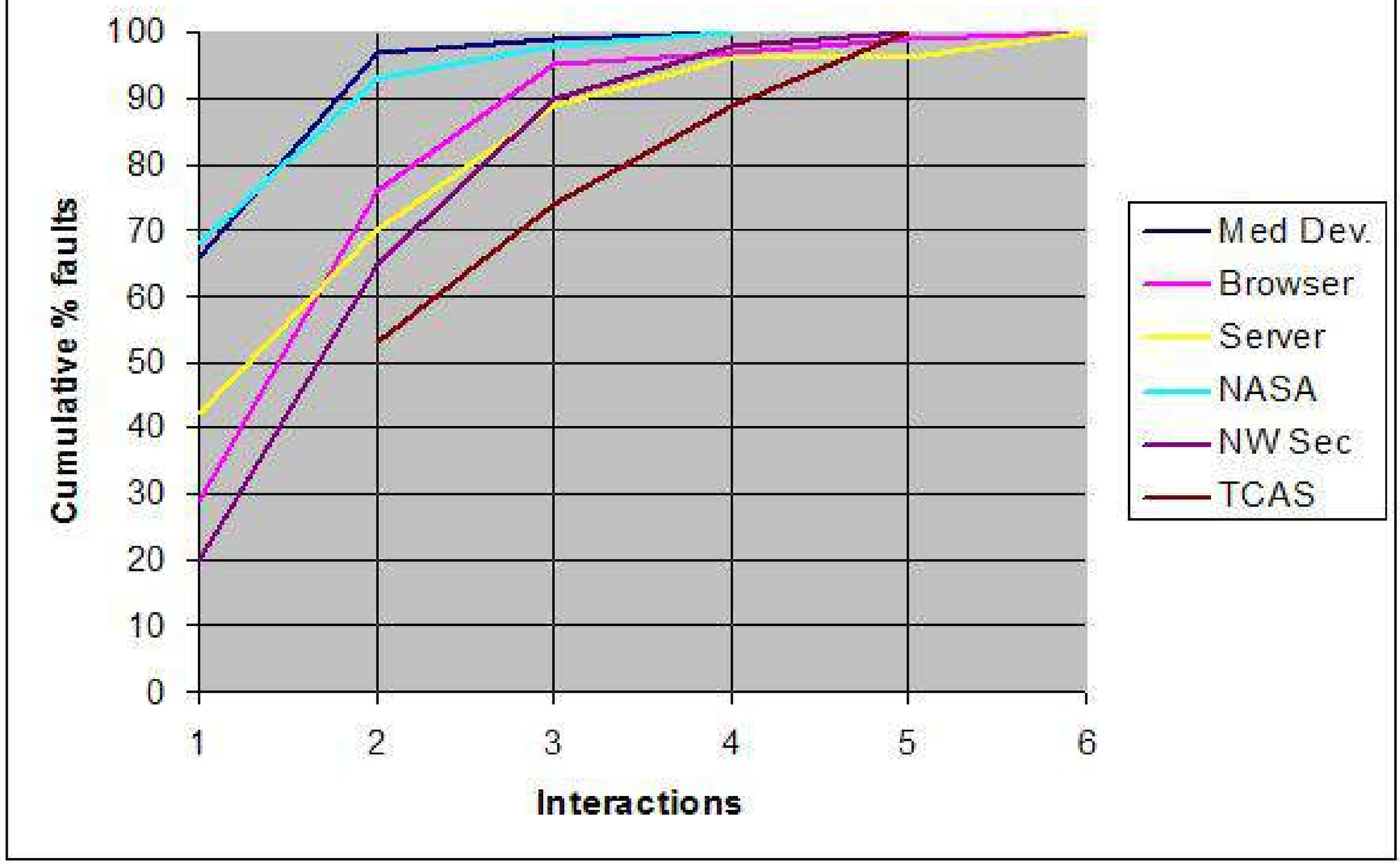
Will a pairwise test suite discover a fault triggered by interactions of 3 or more parameters?

Fault detection

Pairwise testing discovers at least 53% of the known faults.

6-way testing discovers 100% of the known faults.

Fault detection



*

results avilable at `http://csrc.nist.gov/groups/SNS/acts/ftfi.html`

Example

3-way interaction test suite:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

What can we do to discover a pairwise interaction fault more quickly?

Example

3-way interaction test suite:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Test case 5 is the first to discover the fault.

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	1 new pair
Spain	Barcelona	21 Aug	26 Aug	
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	2 new pairs
Spain	Barcelona	21 Aug	26 Aug	
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	3 new pairs
Spain	Barcelona	21 Aug	26 Aug	
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	4 new pairs
Spain	Barcelona	21 Aug	26 Aug	
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	5 new pairs
Spain	Barcelona	21 Aug	26 Aug	
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	0 new pairs
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	1 new pair
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	2 new pairs
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	3 new pairs
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	4 new pairs
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	5 new pairs
Spain	Barcelona	20 Aug	26 Aug	
Spain	Barcelona	21 Aug	27 Aug	
Spain	Madrid	21 Aug	26 Aug	
Spain	Madrid	20 Aug	27 Aug	
Spain	Madrid	20 Aug	26 Aug	
Spain	Madrid	21 Aug	27 Aug	

How many new pairs does each test case cover?

Example

3-way interaction test suite:

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	5 new pairs
Spain	Barcelona	20 Aug	26 Aug	1 new pair
Spain	Barcelona	21 Aug	27 Aug	1 new pair
Spain	Madrid	21 Aug	26 Aug	3 new pairs
Spain	Madrid	20 Aug	27 Aug	2 new pairs
Spain	Madrid	20 Aug	26 Aug	0 new pairs
Spain	Madrid	21 Aug	27 Aug	0 new pairs

How many new pairs does each test case cover?

Prioritisation

2-way prioritised 3-way interaction test suite

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
Spain	Madrid	21 Aug	26 Aug
Spain	Barcelona	21 Aug	26 Aug
Spain	Madrid	20 Aug	27 Aug
Spain	Barcelona	20 Aug	26 Aug
Spain	Barcelona	21 Aug	27 Aug
Spain	Madrid	20 Aug	26 Aug
Spain	Madrid	21 Aug	27 Aug

Prioritisation

2-way prioritised 3-way interaction test suite

Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Madrid	21 Aug	26 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	2 new pairs
Spain	Madrid	20 Aug	27 Aug	2 new pairs
Spain	Barcelona	20 Aug	26 Aug	1 new pair
Spain	Barcelona	21 Aug	27 Aug	1 new pair
Spain	Madrid	20 Aug	26 Aug	0 new pairs
Spain	Madrid	21 Aug	27 Aug	0 new pairs

How quickly will a fault caused by Spain-Madrid interaction be discovered?

Prioritisation

2-way prioritised 3-way interaction test suite

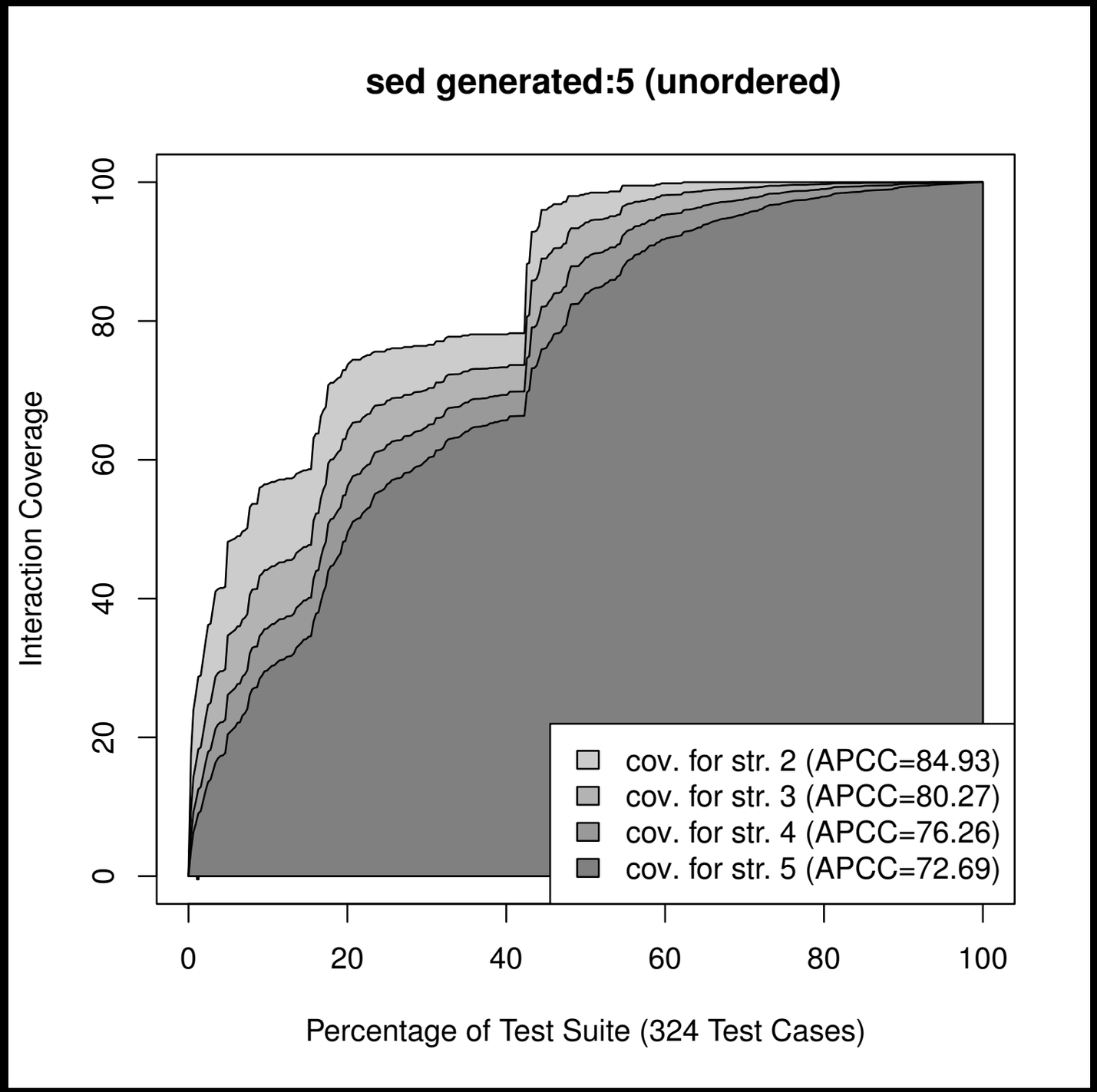
Country	City	Date	Return Date	
Spain	Barcelona	20 Aug	27 Aug	6 new pairs
Spain	Madrid	21 Aug	26 Aug	6 new pairs
Spain	Barcelona	21 Aug	26 Aug	2 new pairs
Spain	Madrid	20 Aug	27 Aug	2 new pairs
Spain	Barcelona	20 Aug	26 Aug	1 new pair
Spain	Barcelona	21 Aug	27 Aug	1 new pair
Spain	Madrid	20 Aug	26 Aug	0 new pairs
Spain	Madrid	21 Aug	27 Aug	0 new pairs

Test case 2 is the first to discover the fault.

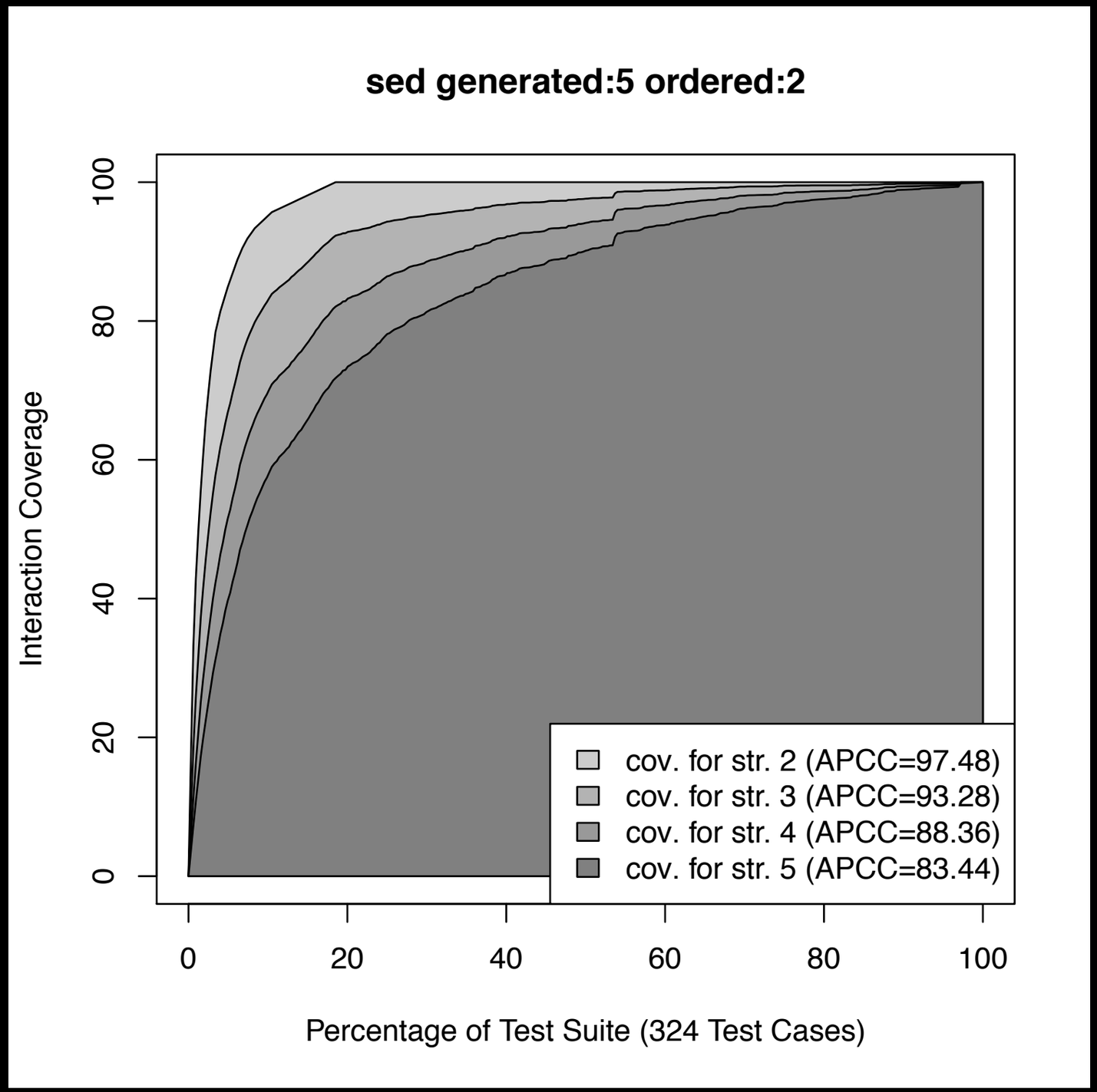
Higher-strength prioritisation

Justyna Petke, Myra B. Cohen, Mark Harman, and Shin Yoo.
Practical Combinatorial Interaction Testing: Empirical Findings on Efficiency and Early Fault Detection.
IEEE Transactions on Software Engineering (TSE) volume 99: 1-26 (2015) Justyna Petke, Shin Yoo, Myra B. Cohen and Mark Harman.
Efficiency and Early Fault Detection with Lower and Higher Strength Combinatorial Interaction Testing.
The 9th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE 2013)

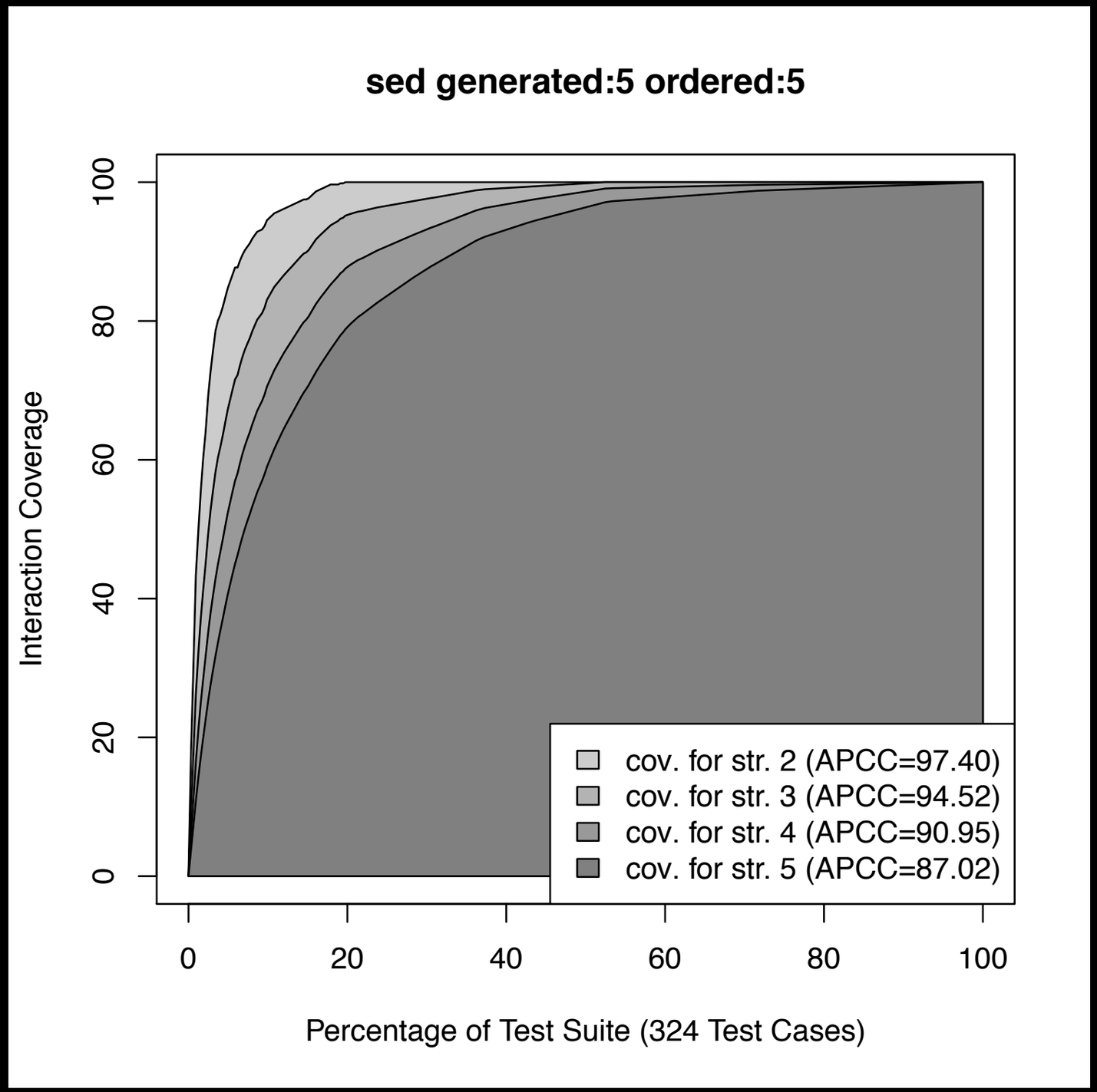
sed 5-way Example



sed 5-way Example



sed 5-way Example



Prioritisation

Pairwise prioritisation is no worse than higher-strength prioritisation in terms of interaction coverage and early fault detection.

Combinatorial Interaction Testing

CIT is a black-box testing technique.

A CIT test suite covers all t-way interactions between any t parameters.

Pairwise (2-way) testing is the most widely studied CIT technique.

Pairwise prioritisation is no worse than higher-strength prioritisation in terms of interaction coverage and early fault detection.

Algorithms

Approaches for generating t-way interaction test suites:

Greedy

Meta-heuristic search

Greedy approach - example

Three parameters: P_1, P_2, P_3

Values for parameter P_1: 0,1
Values for parameter P_2: 0,1
Values for parameter P_3: 0,1,2

Objective: find a pairwise interaction test suite

Greedy algorithm for CIT

IPOG (In-Parameter-Order-General)

IPOG: A General Strategy for T-Way Software Testing Yu Lei et al., 2007

Greedy algorithm IPOG-Test (int t, ParameterSet ps)

1. initialize test set ts to be an empty set
2. denote the parameters in ps, in an arbitrary order, as P_1, P_2,$\cdots$, and P_n
3. add into ts a test for each combination of values of the first t parameters
4.
5.
6.
7.
8.
9.
10.
11.
12.
13.
14.

Greedy approach - example

Adding all combinations of values between the first 2 parameters:

P_1	P_2
0	0
0	1
1	0
1	1

Greedy algorithm IPOG-Test (int t, ParameterSet ps)

1. initialize test set ts to be an empty set
2. denote the parameters in ps, in an arbitrary order, as P_1, P_2,$\cdots$, and P_n
3. add into ts a test for each combination of values of the first t parameters
4. for (int i = t + 1; $i \leq n$; $i++$){
5. let π be the set of t-way combinations of values involving parameter P_i and t-1 parameters among the first i-1 parameters
6.
7.
8.
9.
10.
11.
12.
13.
14.

Greedy approach - example

Set π = pairs to cover involving P_3:

P_1	P_2	P_3
0		0
	0	0
0		1
	0	1
0		2
	0	2
1		0
	1	0
1		1
	1	1
1		2
	1	2

Greedy algorithm IPOG-Test (int t, ParameterSet ps)

1. initialize test set ts to be an empty set
2. denote the parameters in ps, in an arbitrary order, as P_1, P_2,$\cdots$, and P_n
3. add into ts a test for each combination of values of the first t parameters
4. for (int i = t + 1; $i \leq n$; $i++$){
5. let π be the set of t-way combinations of values involving parameter P_i and t-1 parameters among the first i-1 parameters
6. for (each test γ = ($v_1, v_2, \cdots, v_{i-1}$) in test set ts) {
7. choose a value v_i of P_i and replace γ with γ' = ($v_1, v_2, \cdots, v_{i-1}, v_i$) so that γ' covers the most number of combinations of values in π
8. remove from π the combinations of values covered by γ' }
9.
10.
11.
12.
13.
14.

Greedy approach - example

Adding values for P_3 in ts:

P_1	P_2	P_3
0	0	0
0	1	
1	0	
1	1	

P_1	P_2	P_3
0		0
	0	0
0		1
	0	1
0		2
	0	2
1		0
	1	0
1		1
	1	1
1		2
	1	2

Greedy approach - example

Adding values for P_3 in ts:

P_1	P_2	P_3
0	0	0
0	1	1
1	0	
1	1	

P_1	P_2	P_3
0		0
	0	0
0		1
	0	1
0		2
	0	2
1		0
	1	0
1		1
	1	1
1		2
	1	2

Greedy approach - example

Adding values for P_3 in ts:

P_1	P_2	P_3
0	0	0
0	1	1
1	0	1
1	1	

P_1	P_2	P_3
0		0
	0	0
0		1
	0	1
0		2
	0	2
1		0
	1	0
1		1
	1	1
1		2
	1	2

Greedy approach - example

Adding values for P_3 in ts:

P_1	P_2	P_3
0	0	0
0	1	1
1	0	1
1	1	0

P_1	P_2	P_3
0		0
	0	0
0		1
	0	1
0		2
	0	2
1		0
	1	0
1		1
	1	1
1		2
	1	2

Greedy algorithm IPOG-Test (int t, ParameterSet ps)

1. initialize test set ts to be an empty set
2. denote the parameters in ps, in an arbitrary order, as P_1, P_2,$\cdots$, and P_n
3. add into ts a test for each combination of values of the first t parameters
4. for (int i = t + 1; $i \leq n$; $i++$){
5. let π be the set of t-way combinations of values involving parameter P_i and t-1 parameters among the first i-1 parameters
6. for (each test γ = ($v_1, v_2, \cdots, v_{i-1}$) in test set ts) {
7. choose a value v_i of P_i and replace γ with γ' = ($v_1, v_2, \cdots, v_{i-1}, v_i$) so that γ' covers the most number of combinations of values in π
8. remove from π the combinations of values covered by γ' }
9. for (each combination α in set π){
10. if (there exists a test that already covers α) {
11. remove α from π
12. } else {
13. change an existing test, if possible, or otherwise add a new test to cover α and remove it from π
} } }
14. return ts;

Greedy approach - example

Extending ts:

P_1	P_2	P_3
0	0	0
0	1	1
1	0	1
1	1	0

P_1	P_2	P_3
0		0
	0	0
0		1
	0	1
0		2
	0	2
1		0
	1	0
1		1
	1	1
1		2
	1	2

Greedy approach - example

Extending ts:

P_1	P_2	P_3
0	0	0
0	1	1
1	0	1
1	1	0
0	0	2

P_1	P_2	P_3
0		0
	0	0
0		1
	0	1
0		2
	0	2
1		0
	1	0
1		1
	1	1
1		2
	1	2

Greedy approach - example

Extending ts:

P_1	P_2	P_3
0	0	0
0	1	1
1	0	1
1	1	0
0	0	2
1	1	2

P_1	P_2	P_3
0		0
	0	0
0		1
	0	1
0		2
	0	2
1		0
	1	0
1		1
	1	1
1		2
	1	2

Greedy algorithm & prioritisation

In a greedy algorithm values are added to cover the largest amount of t-way interactions.

Thus it already prioritises the test suite!

Meta-heuristics

Meta-heuristics are strategies that guide the search process.

They efficiently explore the search space in order to find near-optimal solutions.

They are not problem-specific.

Meta-heuristic search for CIT

Outer Search (guess test suite size)

Inner Search (populates test suite)

Meta-heuristic search for CIT

Outer Search (e.g. binary)

Inner Search (e.g. simulated-annealing)

Meta-heuristic search for CIT

Evaluating improvements to a meta-heuristic search for constrained interaction testing. Brady J. Garvin et. al, 2011

Meta-heuristic search for CIT

Bluetooth	MP3	Camera
Included None	Included None	Black&White None Colour

Meta-heuristic search for CIT

pick test suite size : 6

Bluetooth	MP3	Camera

Meta-heuristic search for CIT

randomly generate test suite of size 6

Bluetooth	MP3	Camera
Included	Included	Black&White
None	None	None
None	None	Colour
None	None	Black&White
Included	Included	None
None	None	Colour

Meta-heuristic search for CIT

evaluate fitness : 4 pairs missing

Bluetooth	MP3	Camera
Included	Included	Black&White
None	None	None
None	None	Colour
None	None	Black&White
Included	Included	None
None	None	Colour

Meta-heuristic search for CIT

apply mutation and pick the better solution (or worse one with certain probability)
fitness : 3 pairs missing

Bluetooth	MP3	Camera
Included	Included	Black&White
None	None	None
None	None	Colour
None	Included	Black&White
Included	Included	None
None	None	Colour

Meta-heuristic search for CIT

repeat until solution found
or a fixed number of times and then increase test suite size

Bluetooth	MP3	Camera
Included	Included	Black&White
None	Included	None
Included	None	Colour
None	None	Black&White
Included	None	None
None	Included	Colour

Minimality

Is this test suite minimal?

Bluetooth	MP3	Camera
Included	Included	Black&White
None	Included	None
Included	None	Colour
None	None	Black&White
Included	Included	None
None	None	Colour

Minimality

4 pairs to cover:

Bluetooth	MP3	Camera
Included	Included	
Included	None	
None	Included	
None	None	

Minimality

6 pairs to cover:

Bluetooth	MP3	Camera
Included		Black&White
None		Black&White
Included		Colour
None		Colour
Included		None
None		None

Minimality

6 pairs to cover:

Bluetooth	MP3	Camera
	Included	Black&White
	None	Black&White
	Included	Colour
	None	Colour
	Included	None
	None	None

Minimality

Yes! We need at least 6 tests.

Bluetooth	MP3	Camera
Included	Included	Black&White
None	Included	None
Included	None	Colour
None	None	Black&White
Included	None	None
None	Included	Colour

Minimality

However, a greedy approach for CIT test suite generation does not guarantee the minimality of the resultant test suite.

Greedy vs. Meta-heuristics

Size comparison (average over 50 runs).

Subject	Greedy	Meta-heuristics
SPIN-S	27	19
SPIN-V	42	36
GCC	24	21
Apache	42	32
Bugzilla	21	16

*results from: Evaluating improvements to a meta-heuristic search for constrained interaction testing. Brady J. Garvin et. al, 2011

Greedy vs. Meta-heuristics

Time (sec.) comparison (average over 50 runs).

Subject	Greedy	Meta-heuristics
SPIN-S	0.2	8.6
SPIN-V	11.3	102.1
GCC	204	1902.0
Apache	76.4	109.1
Bugzilla	1.9	9.1

*results from: Evaluating improvements to a meta-heuristic search for constrained interaction testing. Brady J. Garvin et. al, 2011

Greedy vs. Meta-heuristics

Meta-heuristic search usually generates smaller test suites, but takes more time than a greedy approach for CIT test suite generation.

Combinatorial Explosion problem

We can reduce the test suite size by generating a t-way interaction test suite. Can we reduce a CIT test suite even further?

Example 3

Booking a flight from London:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
France	Paris		

Example 3

Pairwise interaction test suite:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
France	Paris	20 Aug	27 Aug
Spain	Paris	20 Aug	27 Aug
France	Barcelona	20 Aug	27 Aug

What is wrong with these input values?

Hard Constraints

Pairwise interaction test suite:

Country	City	Date	Return Date
Spain	Barcelona	20 Aug	27 Aug
France	Paris	20 Aug	27 Aug
Spain	Paris	20 Aug	27 Aug
France	Barcelona	20 Aug	27 Aug

Example 4

Booking a flight from London:

City	via City	Date	Return Date
Barcelona (Spain)	Paris (France) Nuuk (Greenland)	20 Aug	27 Aug

Example 4

Pairwise intertaction test suite:

City	via City	Date	Return Date
Barcelona (Spain)	Paris (France)	20 Aug	27 Aug
Barcelona (Spain)	Nuuk (Greenland)	20 Aug	27 Aug

What is wrong with these input values?

Soft Constraints

Booking a flight from London:

City	via City	Date	Return Date
Barcelona (Spain)	Paris (France)	20 Aug	27 Aug
Barcelona (Spain)	Nuuk (Greenland)	20 Aug	27 Aug

Soft Constraints

find <pattern> <empty file>

throws file doesn't exist error

testing with other parameter-value combinations is not required

Constraints in CIT

Hard constraints prohibit certain parameter-value combinations (i.e. interactions).

Soft constraints are usually imposed by the tester to exclude interactions that don't need to be tested.

Web Browser Example

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow Restrict Block	Yes No	Allow Restrict Block	Yes No

Generate a 3-way interaction test suite.

3-way interactions to cover:

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Allow	
Allow	Yes	Restrict	
Allow	Yes	Block	
Allow	No	Allow	
Allow	No	Restrict	
Allow	No	Block	
Restrict	Yes	Allow	
Restrict	Yes	Restrict	
Restrict	Yes	Block	
Restrict	No	Allow	
Restrict	No	Restrict	
Restrict	No	Block	
Block	Yes	Allow	
Block	Yes	Restrict	
Block	Yes	Block	
Block	No	Allow	
Block	No	Restrict	
Block	No	Block	

Web Browser Example

3-way interactions to cover:

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
3x2x3=18 interactions			

Web Browser Example

3-way interactions to cover:

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes		Yes
Restrict	No		No
Block			

Web Browser Example

3-way interactions to cover:

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
3 x 2			x 2 =12 interactions

Web Browser Example

3-way interactions to cover:

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
	2x3x2=12 interactions		

Web Browser Example

3-way interactions to cover:

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
3		x 3 x 2 =18 interactions	

Web Browser Example

Overall: 18+12+12+18 = 60 3-way interactions to cover.

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow Restrict Block	Yes No	Allow Restrict Block	Yes No

What is the lower bound on the size of the smallest 3-way test suite?

Minimal 3-way interaction test suite (18 test cases):

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Restrict	No	Allow	Yes
Block	No	Block	Yes
Block	Yes	Block	No
Allow	No	Block	No
Allow	Yes	Block	Yes
Restrict	Yes	Allow	No
Restrict	No	Restrict	No
Allow	No	Restrict	Yes
Restrict	Yes	Restrict	Yes
Allow	Yes	Restrict	No
Restrict	Yes	Block	No
Block	Yes	Restrict	Yes
Block	Yes	Allow	Yes
Block	No	Restrict	No
Allow	Yes	Allow	Yes
Allow	No	Allow	No
Restrict	No	Block	Yes
Block	No	Allow	No

Web Browser Example

Generate 3-way interaction test suite:

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow Restrict Block	Yes No	Allow Restrict Block	Yes No

Constraints:

1. if ‘Load content’=‘Restrict’ then ‘Notify pop-up blocked’=‘Yes’

Which test cases violate the constraint:

if ‘Load content’=‘Restrict’ then ‘Notify pop-up blocked’=‘Yes’ ?

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
1	Restrict	No	Allow	Yes
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
7	Restrict	No	Restrict	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
17	Restrict	No	Block	Yes
18	Block	No	Allow	No

Solution: Remove all invalid test cases?

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No

Is the following 3-way interaction covered? :

'Load content'='Restrict', 'Cookies=Allow', 'Warn before (..)'='Yes'

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No

Solution: change violating values, e.g. ‘No’ to ‘Yes’?

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
1	Restrict	Yes	Allow	Yes
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
7	Restrict	Yes	Restrict	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
17	Restrict	Yes	Block	Yes
18	Block	No	Allow	No

Is the following 3-way interaction covered? :

'Notify pop-up (..)'='No', 'Cookies=Allow', 'Warn before (..)'='Yes'

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
1	Restrict	Yes	Allow	Yes
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
7	Restrict	Yes	Restrict	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
17	Restrict	Yes	Block	Yes
18	Block	No	Allow	No

Which interactions are missing when tcs 1,7 and 17 removed?

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
1	Restrict	No	Allow	Yes
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
7	Restrict	No	Restrict	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
17	Restrict	No	Block	Yes
18	Block	No	Allow	No

Web Browser Example

Which interactions are missing when tcs 1,7 and 17 removed?

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
1	Restrict	No	Allow	Yes
	Restrict		Allow	Yes
		No	Allow	Yes
7	Restrict	No	Restrict	No
	Restrict		Restrict	No
		No	Restrict	No
17	Restrict	No	Block	Yes
	Restrict		Block	Yes
		No	Block	Yes

Web Browser Example

3-way interactions to cover:

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
	Restrict		Allow	Yes
		No	Allow	Yes
	Restrict		Restrict	No
	Restrict		Restrict	No
		No	Block	Yes
	Restrict		Block	Yes
		No	Block	Yes

Is the following 3-way interaction covered? :

'Load content'='Restrict', 'Cookies=Allow', 'Warn before (..)'='Yes' NO

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No

Is the following 3-way interaction covered? :

'Notify pop-up (..)'='No', 'Cookies=Allow', 'Warn before (..)'='Yes' NO

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No

Is the following 3-way interaction covered? :

'Load content'='Restrict', 'Cookies=Restrict', 'Warn before (..)'='No' NO

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No

Is the following 3-way interaction covered? :

'Notify pop-up (..)'='No', 'Cookies=Restrict', 'Warn before (..)'='No' YES

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No

Is the following 3-way interaction covered? :

'Load content'='Restrict', 'Cookies=Block', 'Warn before (..)'='Yes' NO

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No

Is the following 3-way interaction covered? :

'Notify pop-up (..)='No', 'Cookies=Block', 'Warn before (..)'='Yes' YES

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No

Web Browser Example

3-way interactions to cover:

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
	Restrict		Allow	Yes
		No	Allow	Yes
	Restrict		Restrict	No
	Restrict		Restrict	No
		No	Block	Yes
	Restrict		Block	Yes
		No	Block	Yes

Web Browser Example

3-way interactions to cover:

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
	Restrict		Allow	Yes
		No	Allow	Yes
	Restrict		Restrict	No
	Restrict		Block	Yes

What can we do to cover these interactions?

Add new test cases.

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
2	Block	No	Block	Yes
3	Block	Yes	Block	No
4	Allow	No	Block	No
5	Allow	Yes	Block	Yes
6	Restrict	Yes	Allow	No
8	Allow	No	Restrict	Yes
9	Restrict	Yes	Restrict	Yes
10	Allow	Yes	Restrict	No
11	Restrict	Yes	Block	No
12	Block	Yes	Restrict	Yes
13	Block	Yes	Allow	Yes
14	Block	No	Restrict	No
15	Allow	Yes	Allow	Yes
16	Allow	No	Allow	No
18	Block	No	Allow	No
	Restrict		Allow	Yes
		No	Allow	Yes
	Restrict		Restrict	No
	Restrict		Block	Yes

	Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
1	Block	No	Block	Yes
2	Block	Yes	Block	No
3	Allow	No	Block	No
4	Allow	Yes	Block	Yes
5	Restrict	Yes	Allow	No
6	Allow	No	Restrict	Yes
7	Restrict	Yes	Restrict	Yes
8	Allow	Yes	Restrict	No
9	Restrict	Yes	Block	No
10	Block	Yes	Restrict	Yes
11	Block	Yes	Allow	Yes
12	Block	No	Restrict	No
13	Allow	Yes	Allow	Yes
14	Allow	No	Allow	No
15	Block	No	Allow	No
16	Restrict	Yes	Allow	Yes
17	Block	No	Allow	Yes
18	Restrict	Yes	Restrict	No
19	Restrict	Yes	Block	Yes

Web Browser Example

But constraints were supposed to reduce test suite size !

Real-world instances contain a lot of constraints which help reduce test suite size.

Web Browser Example

Dealing with constraints:

- generate an unconstrained test suite; remove unsatisfiable tests; add new test cases to cover uncovered t-way interactions
- generate a CIT test suite directly

Web Browser Example

Generate a 3-way interaction test suite.

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow Restrict Block	Yes No	Allow Restrict Block	Yes No

Constraints:

1. Allow content to load if and only if cookies are allowed.
2. Always notify if pop-up is blocked.
3. If content is blocked, then block cookies and don't warn before add-ons install.
4. Content loading is restricted if and only if cookies are restricted.

Web Browser Example

Generate a 3-way interaction test suite.

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow Restrict Block	Yes No	Allow Restrict Block	Yes No

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Which value can we exclude?

Web Browser Example

Which 3-way interactions do we need to cover?

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow Restrict Block	Yes No	Allow Restrict Block	Yes No

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Which 3-way interactions do we need to cover?

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Allow	
Restrict	Yes	Restrict	
Block	Yes	Block	

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Which 3-way interactions do we need to cover?

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow		Allow	Yes
Restrict		Restrict	Yes
Block		Block	Yes
Allow		Allow	No
Restrict		Restrict	No
Block		Block	No

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Is there any 3-way interaction that violates some constraint?

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow		Allow	Yes
Restrict		Restrict	Yes
Block		Block	Yes
Allow		Allow	No
Restrict		Restrict	No
Block		Block	No

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Is there any 3-way interaction that violates some constraint?

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow		Allow	Yes
Restrict		Restrict	Yes
Block		Block	Yes
Allow		Allow	No
Restrict		Restrict	No
Block		Block	No

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Which 3-way interactions do we need to cover?

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow		Allow	Yes
Restrict		Restrict	Yes
Allow		Allow	No
Restrict		Restrict	No
Block		Block	No

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Which 3-way interactions do we need to cover?

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
	Yes	Allow	Yes
	Yes	Restrict	Yes
	Yes	Block	Yes
	Yes	Allow	No
	Yes	Restrict	No
	Yes	Block	No

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Is there any 3-way interaction that violates some constraint? NO

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
	Yes	Allow	Yes
	Yes	Restrict	Yes
	Yes	Block	Yes
	Yes	Allow	No
	Yes	Restrict	No
	Yes	Block	No

Constraints:

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

14 3-way interactions need to be covered (previously 60 !).

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Allow	
Restrict	Yes	Restrict	
Block	Yes	Block	
Allow		Allow	Yes
Restrict		Restrict	Yes
Allow		Allow	No
Restrict		Restrict	No
Block		Block	No
	Yes	Allow	Yes
	Yes	Restrict	Yes
	Yes	Block	Yes
	Yes	Allow	No
	Yes	Restrict	No
	Yes	Block	No

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

How many test cases do we need?

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Allow	
Restrict	Yes	Restrict	
Block	Yes	Block	
Allow		Allow	Yes
Restrict		Restrict	Yes
Allow		Allow	No
Restrict		Restrict	No
Block		Block	No
	Yes	Allow	Yes
	Yes	Restrict	Yes
	Yes	Block	Yes
	Yes	Allow	No
	Yes	Restrict	No
	Yes	Block	No

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

‘Notify pop-up’ takes just one value, so we only need to cover:

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Allow	
Restrict	Yes	Restrict	
Block	Yes	Block	
Allow		Allow	Yes
Restrict		Restrict	Yes
Allow		Allow	No
Restrict		Restrict	No
Block		Block	No
	Yes	Allow	Yes
	Yes	Restrict	Yes
	Yes	Block	Yes
	Yes	Allow	No
	Yes	Restrict	No
	Yes	Block	No

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

But one interaction is left uncovered !

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Allow	
Restrict	Yes	Restrict	
Block	Yes	Block	
Allow	Yes	Allow	Yes
Restrict	Yes	Restrict	Yes
Allow	Yes	Allow	No
Restrict	Yes	Restrict	No
Block	Yes	Block	No
	Yes	Allow	Yes
	Yes	Restrict	Yes
	Yes	Block	Yes
	Yes	Allow	No
	Yes	Restrict	No
	Yes	Block	No

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

But one interaction is left uncovered !

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Block	Yes
Restrict	Yes	Block	Yes
Block	Yes	Block	Yes

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

First test case violates the first constraint.

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Block	Yes
Restrict	Yes	Block	Yes
Block	Yes	Block	Yes

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Second test case violates the fourth constraint.

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Block	Yes
Restrict	Yes	Block	Yes
Block	Yes	Block	Yes

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Third test case violates the third constraint.

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Block	Yes
Restrict	Yes	Block	Yes
Block	Yes	Block	Yes

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Combinations of constraints forbid certain interactions.

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Allow	
Restrict	Yes	Restrict	
Block	Yes	Block	
Allow	Yes	Allow	Yes
Restrict	Yes	Restrict	Yes
Allow	Yes	Allow	No
Restrict	Yes	Restrict	No
Block	Yes	Block	No
	Yes	Allow	Yes
	Yes	Restrict	Yes
	Yes	Block	Yes
	Yes	Allow	No
	Yes	Restrict	No
	Yes	Block	No

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Web Browser Example

Minimal constrained test suite of size 5 (previously 18).

Load content	Notify pop-up blocked	Cookies	Warn before add-ons install
Allow	Yes	Allow	Yes
Restrict	Yes	Restrict	Yes
Allow	Yes	Allow	No
Restrict	Yes	Restrict	No
Block	Yes	Block	No

1. ‘Load content’=‘Allow’ $\leftrightarrow$ ‘Cookies’=‘Allow’
2. ‘Notify pop-up (..)’=‘Yes’
3. (‘Load content’=‘Block’) $\rightarrow$ (‘Cookies=‘Block’ $\&$ ‘Warn (..)’=‘No’)
4. ‘Load content’=‘Restrict $\leftrightarrow$ ‘Cookies’=‘Restrict’

Combinatorial Interaction

40 tools available from pairwise.org

and others…

Summary of Part 2

A CIT test suite covers all t-way interactions between any t-parameters.

Finding a minimal test suite is a challenging task.

Most popular CIT generation approaches: greedy and meta-heuristic search.

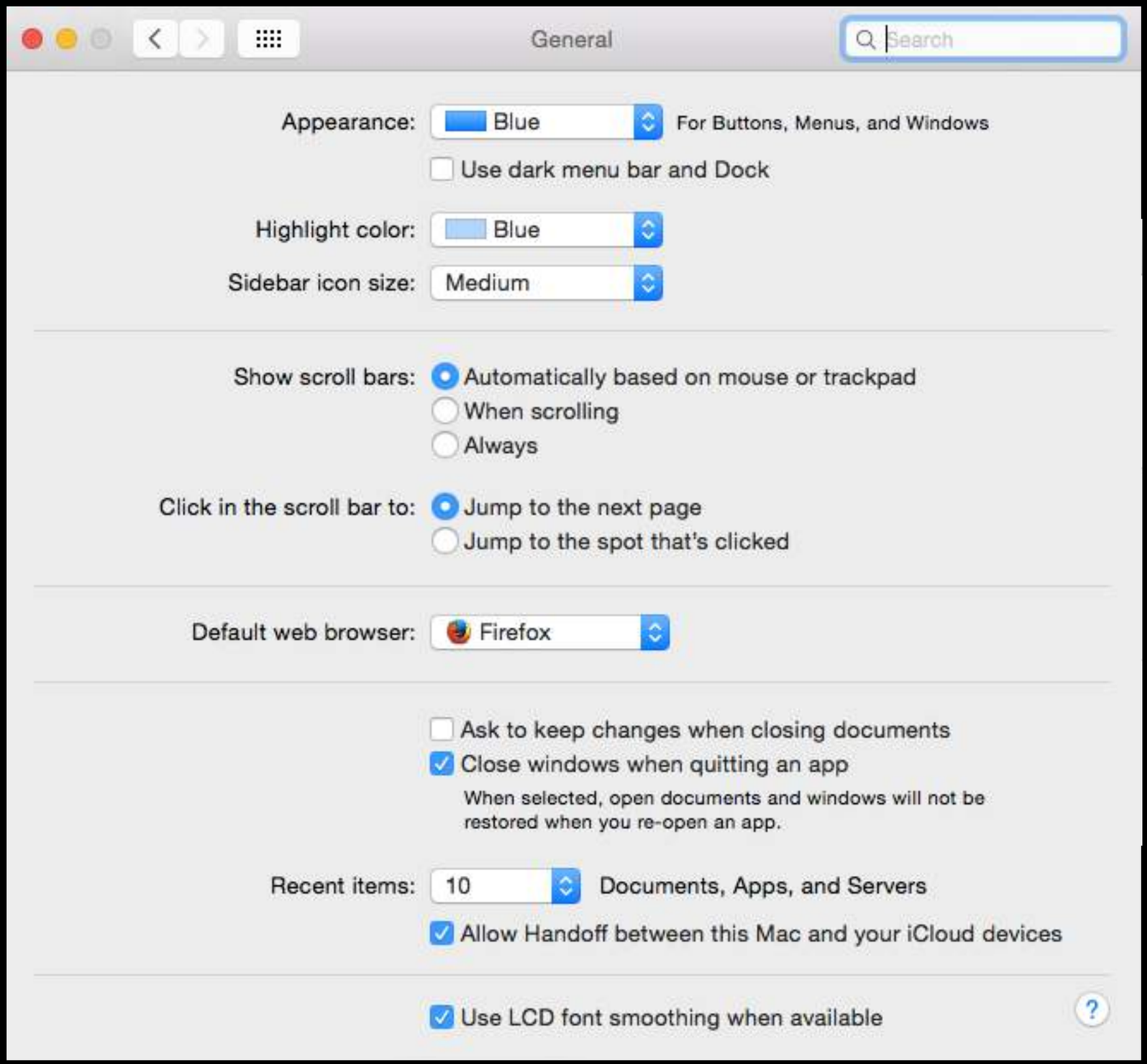
Greedy approach generates larger test suites than meta-heuristic search, but in shorter amount of time.

Constraints help reduce the test suite size even further.

t-way testing for $t > 2$ is feasible thanks to constraints !

Exercises

What are the parameters and values ?



System Preferences Model

Combinatorial Interaction Testing Model:

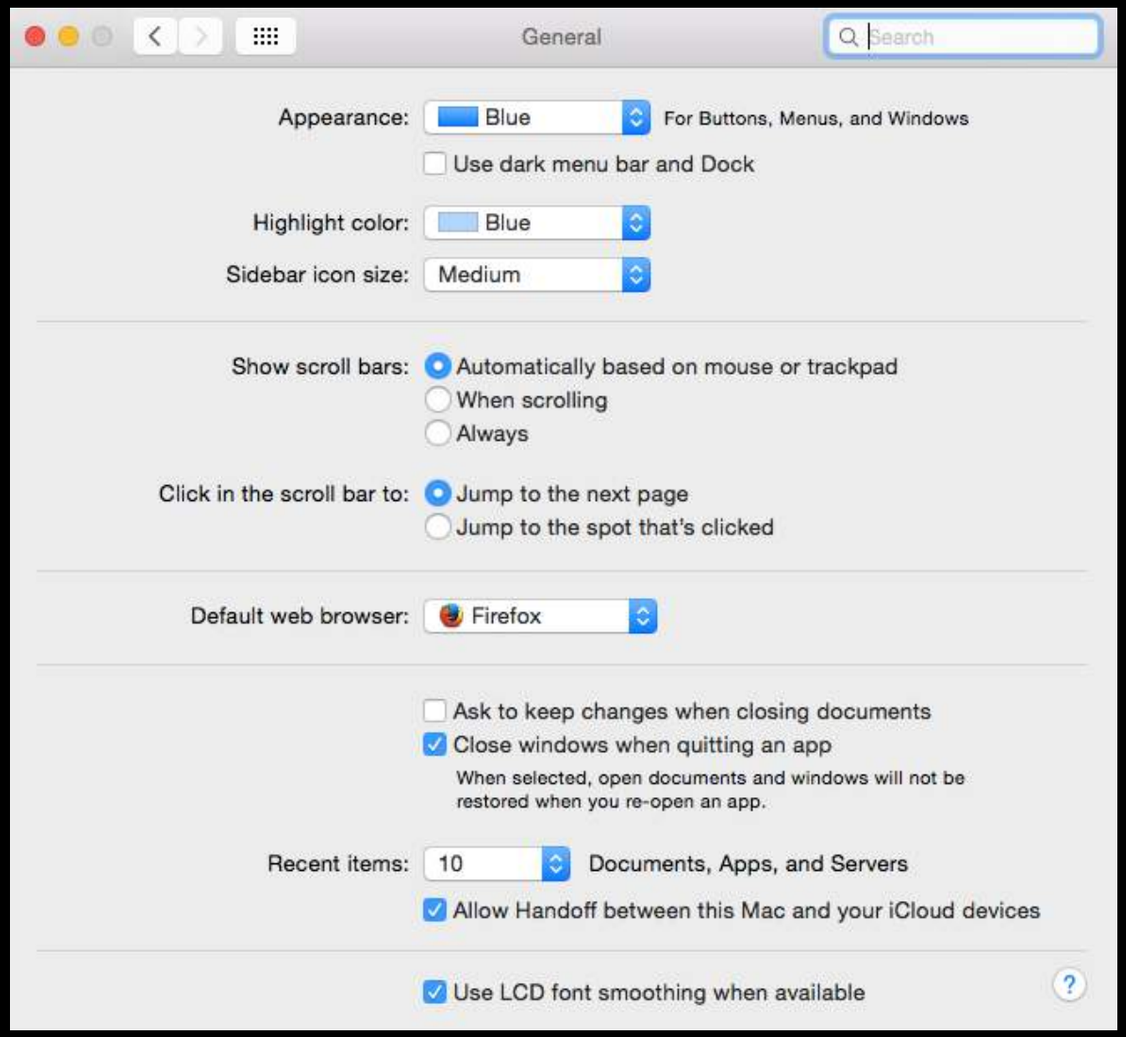
7 Boolean parameters

2 parameters with 3 values

1 parameter with 5 values

1 parameter with 7 values

1 parameter with 10 values

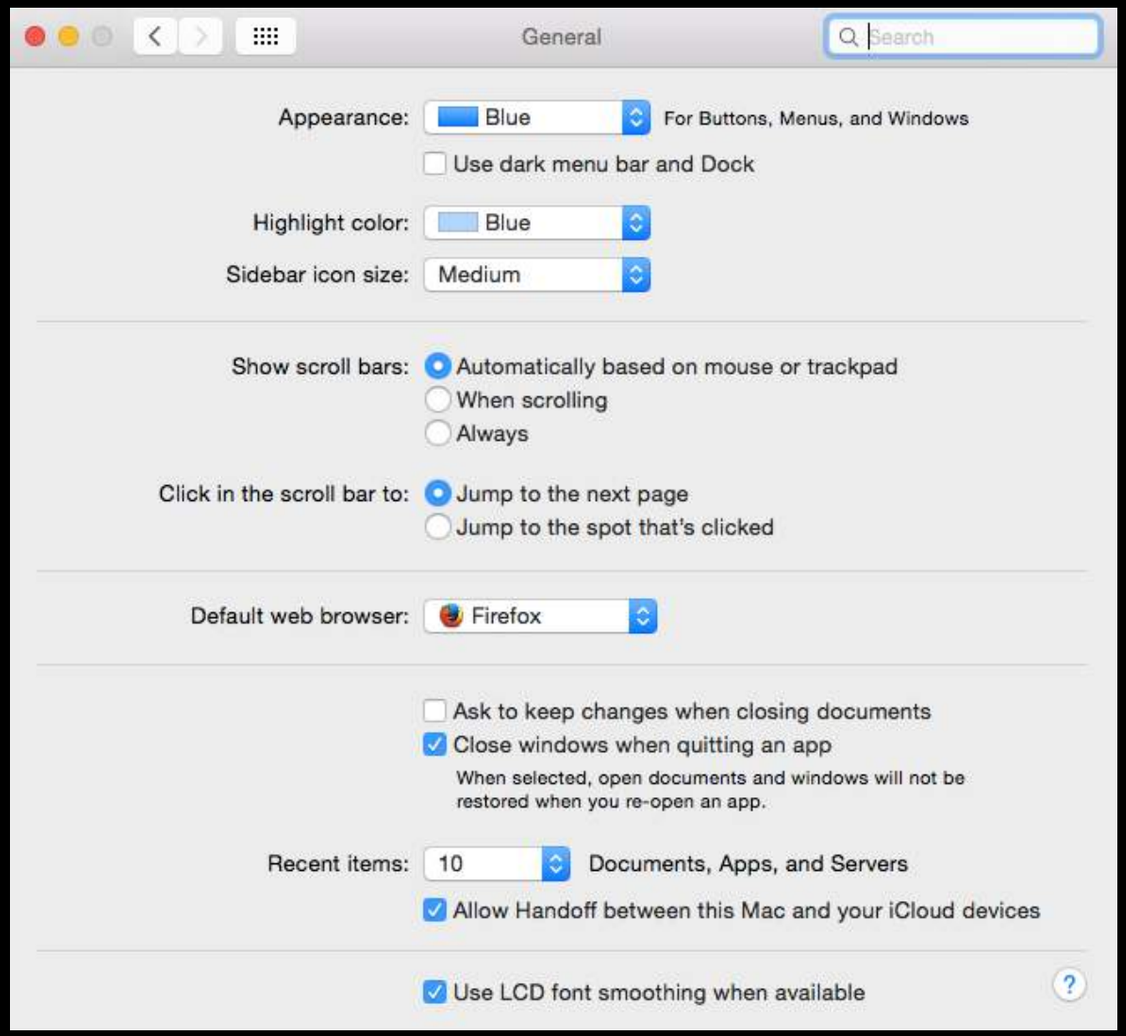


System Preferences Model

Combinatorial Interaction Testing Model:

12 parameters

42 values



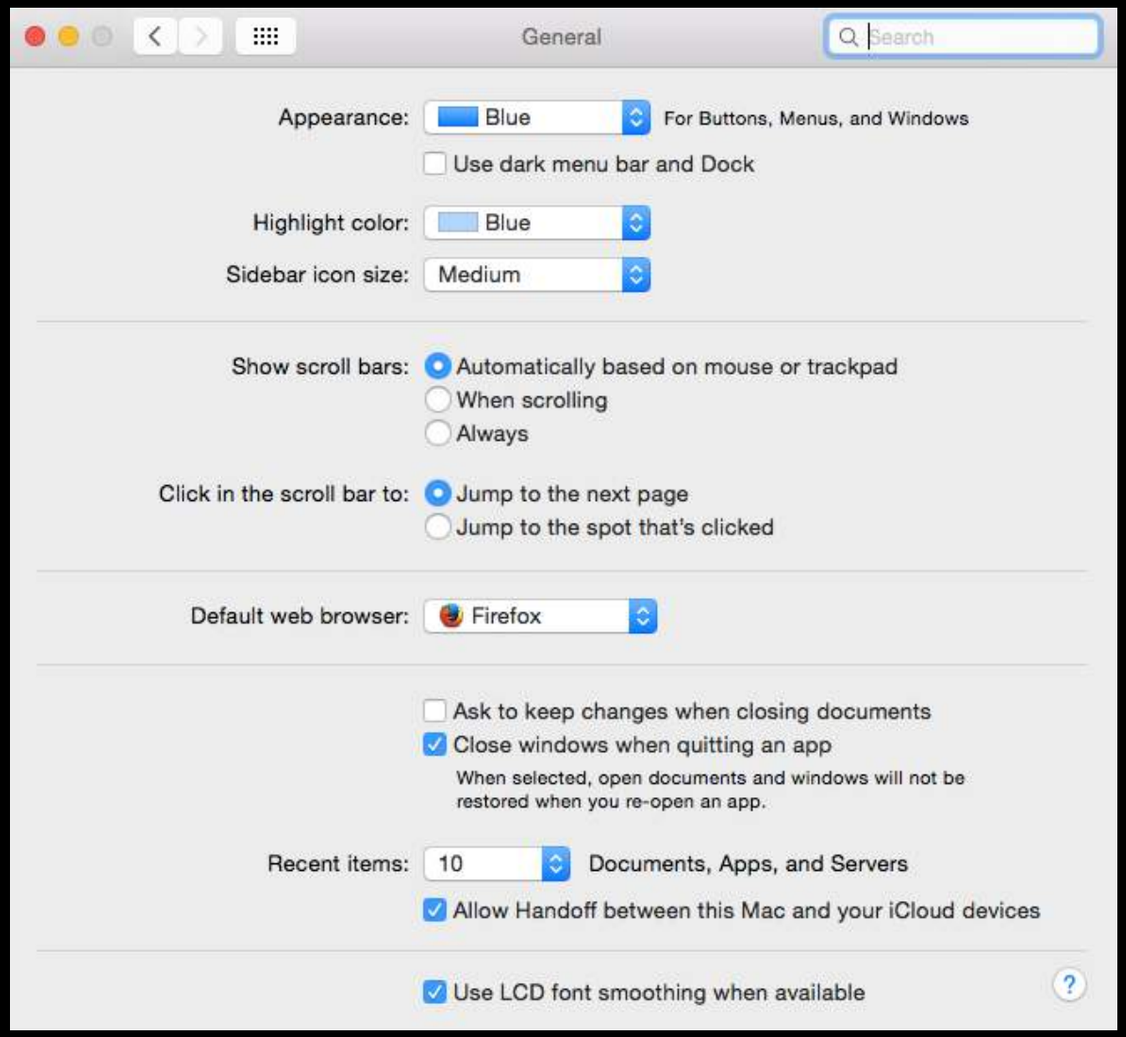
System Preferences Model

Combinatorial Interaction Testing Model (CASA):

2 ⟵ pairwise

12

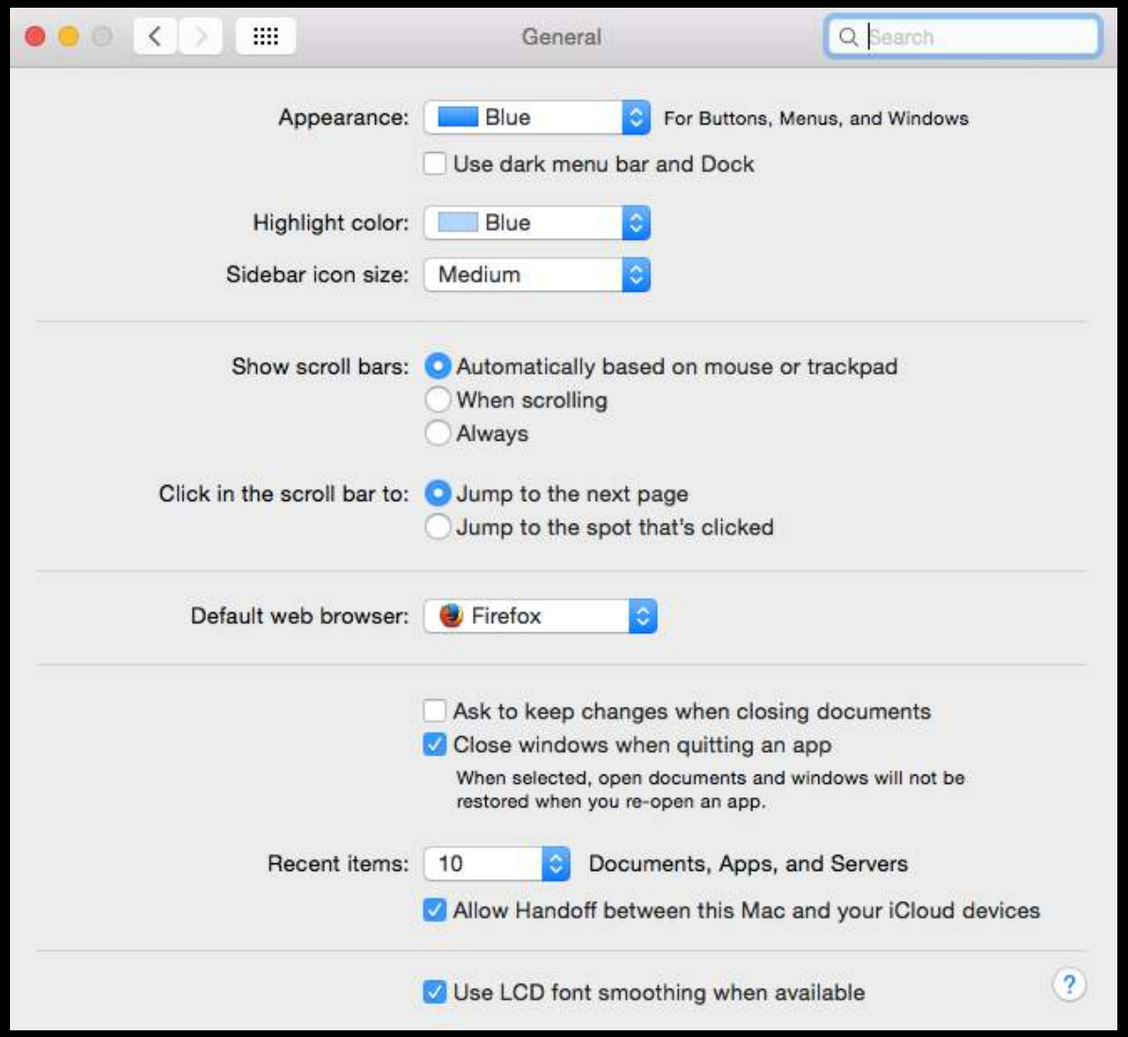
2 2 10 3 3 2 5 2 2 7 2 2



System Preferences Model

Combinatorial Interaction Testing Model (exhaustive):

2 x 2 x10 x 3 x 3 x 2 x 5 x 2 x 2 x 7 x 2 x 2

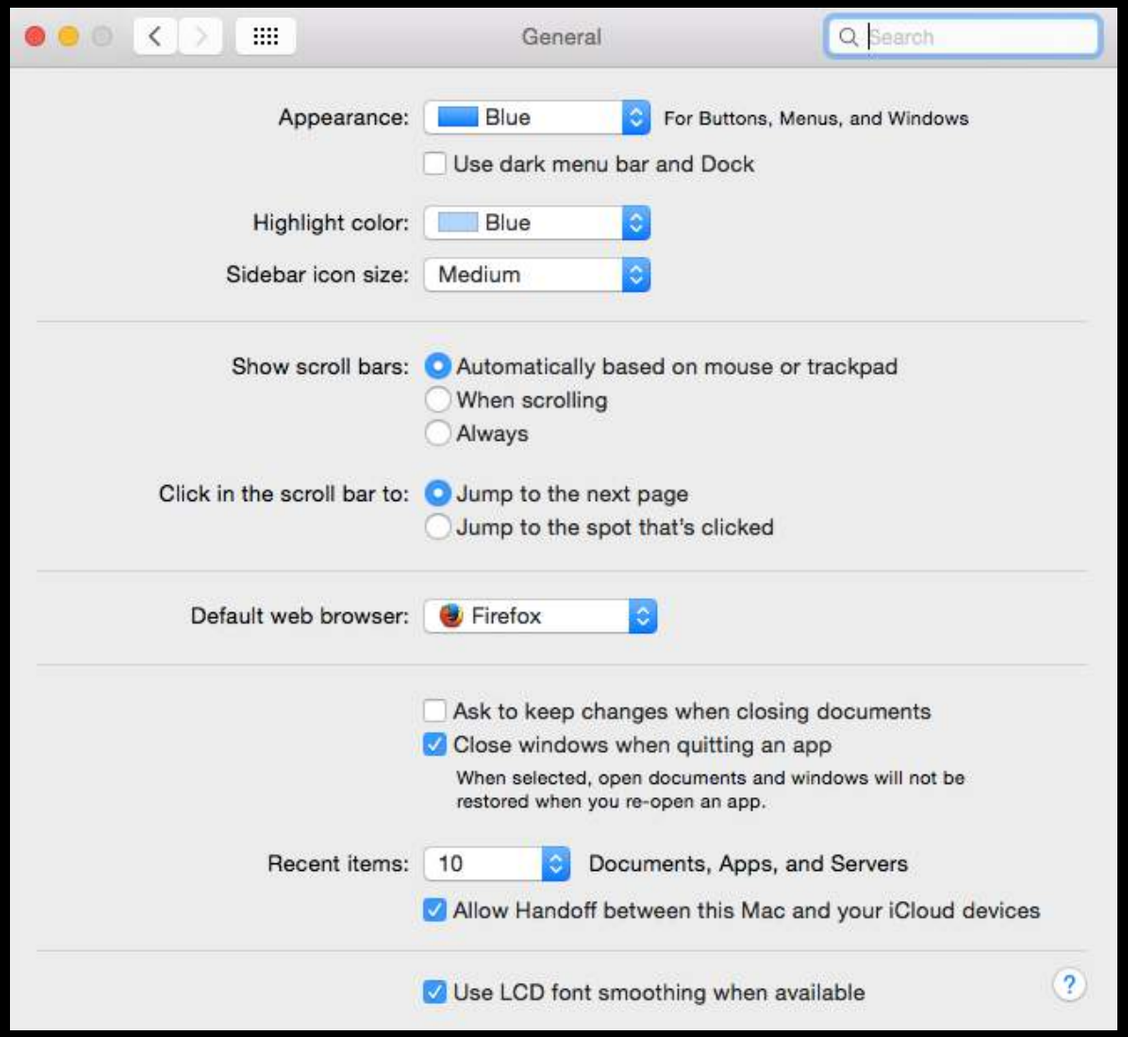


System Preferences Model

Combinatorial Interaction Testing Model (exhaustive):

2 x 2 x10 x 3 x 3 x 2 x 5 x 2 x 2 x 7 x 2 x 2

= 403,200

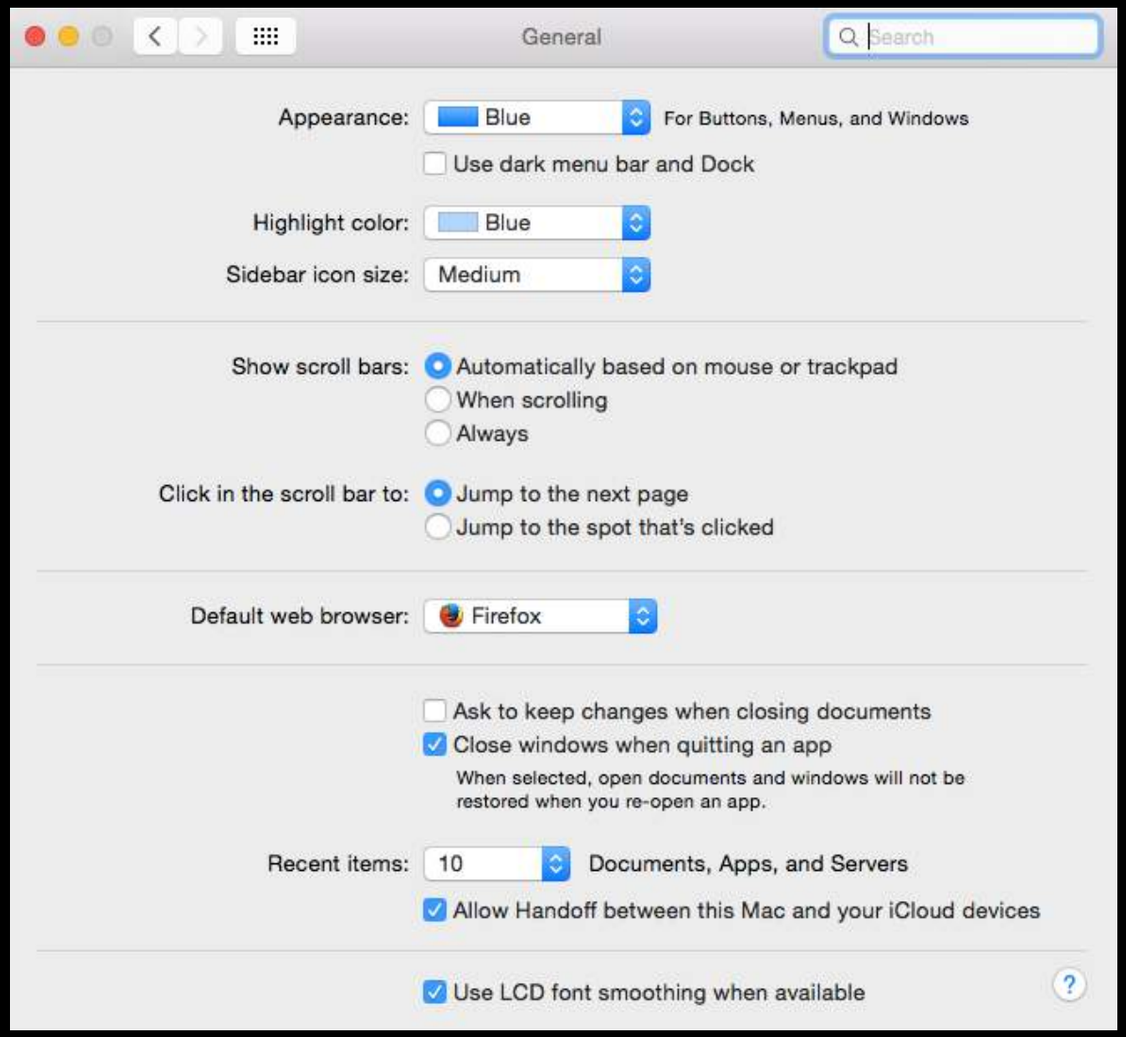


System Preferences Model

Combinatorial Interaction Testing Model (CASA):

pairwise test suite:

70 test cases within 0.6 sec



System Preferences Model

Combinatorial Interaction Testing Model (CASA):

3-way interaction test suite:

354 test cases within 24 sec