Combinatorial Testing

From S. Somé, A. Williams

1

Equivalence Class Testing of Methods

```
public void aMethod( int x, int y, int z )
{
   ...
}
```

- Suppose that the equivalence classes for each individual parameter are:
  - **x** : (-∞, -4] (-4, 3] (3, 8] (8, +∞)
  - **y** : (-∞, 12) [12, 34] (34, +∞)
  - **z** : (-∞, 0) [0] (0, +∞)
- What about combinations of equivalence classes for **x**, **y**, and **z**?

2

Combinatorial Testing

- Context:
  - We have a set of "parameters".
    - Previous example: method parameters x, y, z
  - For each parameter, there is a finite set of discrete "values"
    - Previous example: the set of equivalence classes for each of x, y, z.
  - The value for each parameter can be chosen independently of any other value.

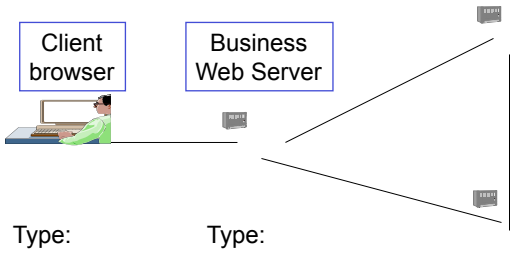
3

Example: User preferences

- Choosing from a set of options for user preferences for a web page display
  - Display mode:
    - Full-graphics
    - Limited bandwidth
    - Text only
  - Language
    - English
    - French
    - Spanish
  - Fonts:
    - Minimal
    - Standard
    - Loaded from document
  - Screen size
    - Cell phone
    - Regular monitor

4

Example: E-commerce components



- How many configurations should be used to test applications in this platform?

5

The issue

- Combinatorial explosion: The number of combinations is typically too large for any realistic test budget.
  - E-commerce example: 4 parameters, and each parameter can have 3 possible values.
    - Result is $3^4 = 81$ possible combinations
- How can we test a reduced number of combinations, while still achieving a known level of coverage of potential interactions?

6

Objectives

1. Develop a measure that shows how well potential interactions among parameters are covered by a set of test configurations.

2. Determine how to achieve the highest interaction coverage with the fewest number of configurations.

7

Test Configurations

- Created by selecting one value for each parameter from the set of permitted values.
  - Interaction degree: size of subset of interest
- For the e-commerce platform example:

Browser:	Firefox
Web server:	Apache
Payment method:	Visa
Database:	DB/2

8

Generic Example

- Suppose that we have three parameters: P1, P2, P3
- For each parameter, there are two possible values.
  - Values are :
    - A, B for parameter P1.
    - C, D for parameter P2.
    - E, F for parameter P3.
- Selected degree of interaction coverage is 2 ("pair-wise coverage").
  - We want to cover all potential 2-way interactions among parameter values.

9

Set of potential test configurations

	P1	P2	P3
C1	A	C	E
C2	A	C	F
C3	A	D	E
C4	A	D	F
C5	B	C	E
C6	B	C	F
C7	B	D	E
C8	B	D	F

Three parameters, P1, P2, P3 each of which has two values.

There are $2^3 = 8$ potential test configurations, C1,…, C8.

10

Set of potential 2-way interactions

P1	P2	[P3]
A	C	
A	D	
B	C	
B	D	

P1	[P2]	P3
A		E
A		F
B		E
B		F

[P1]	P2	P3
	C	E
	C	F
	D	E
	D	F

- There are $\binom{3}{2} \times 2^2 = 12$ potential 2-way interactions
  - 3 parameters but pairwise then for these 2 parameters there are 2 choices each…
- Coverage measure: percentage of these interactions included.

11

Interactions included in a configuration

One test configuration…

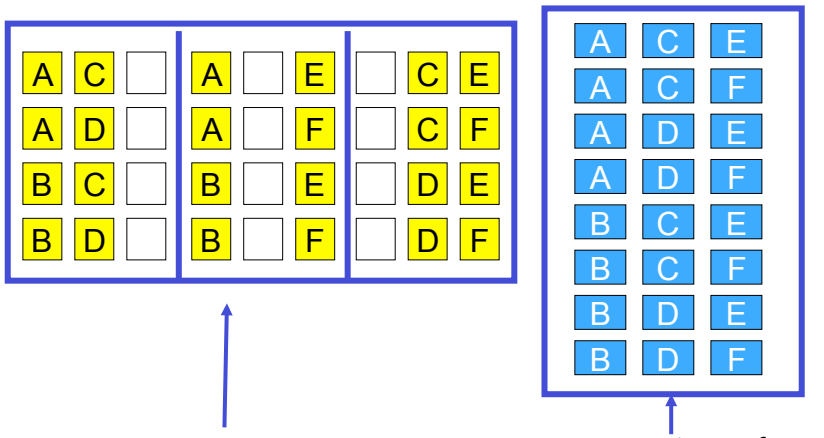
A	C	E

… covers 3 interactions.

A	C	
A		E
	C	E

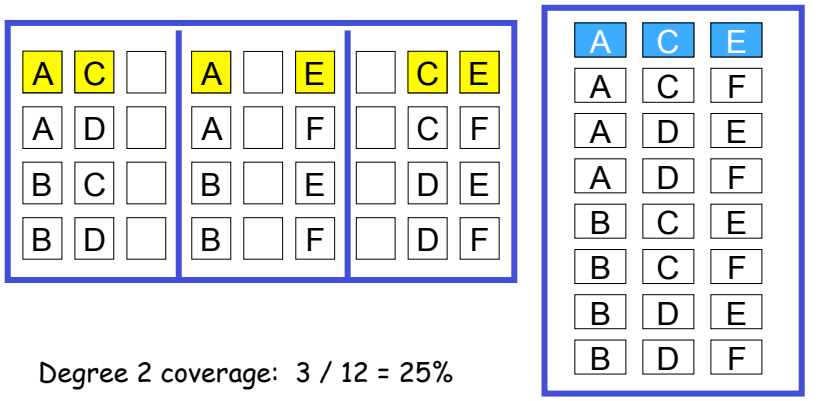
12

Interaction Coverage Goal



13

Selection of Configurations

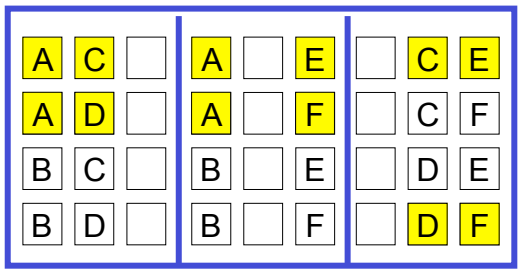


Degree 2 coverage: 3 / 12 = 25%

Degree 3 coverage: 1 / 8 = 12.5%

14

Selection of Configurations



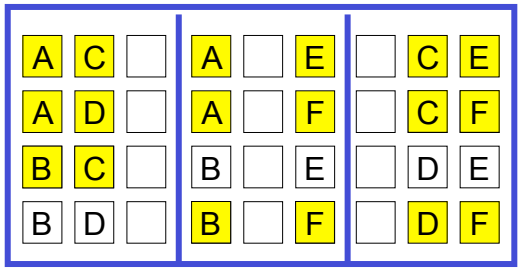
A	C	E
A	C	F
A	D	E
A	D	F
B	C	E
B	C	F
B	D	E
B	D	F

Degree 2 coverage: 6 / 12 = 50%

Degree 3 coverage: 2 / 8 = 25%

15

Selection of Configurations



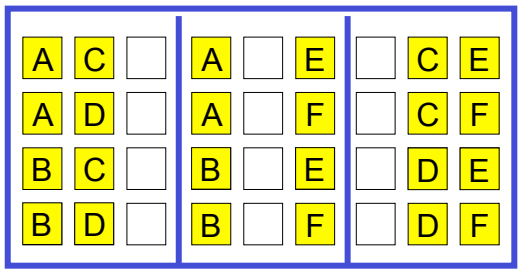
A	C	E
A	C	F
A	D	E
A	D	F
B	C	E
B	C	F
B	D	E
B	D	F

Degree 2 coverage: 9 / 12 = 75%

Degree 3 coverage: 3 / 8 = 37.5%

16

Selection of Configurations



A	C	E
A	C	F
A	D	E
A	D	F
B	C	E
B	C	F
B	D	E
B	D	F

Degree 2 coverage: 12 / 12 = 100%

Degree 3 coverage: 4 / 8 = 50%

17

Choosing the degree of coverage

- Trade-off: fewer test configurations versus leaving some combinations uncovered.
- What is the likelihood that an unwanted interaction is caused by a specific combination of 3 (or more) parameters?
- In one experiment, covering 2 way interactions of equivalence classes for a method's parameters resulted in the following average code coverage:
  - 93% block coverage (i.e., from control flow graphs)
  - 83% decision coverage (covering of Boolean conditions)
  - 73% all-uses (from data flow analysis) coverage.

18

Section summary

- We have defined how to measure coverage of potential system interactions.
- Strategy for choosing test configurations:
  - Maximize coverage of interaction elements for a given degree.
- Choose interaction degree based on:
  - Degree of interaction risk that can be tolerated.
  - Test budget constraints.

19

Objectives

1. Develop a measure that shows how well potential interactions among parameters are covered by a set of test configurations.

2. Determine how to achieve the highest interaction coverage with the fewest number of configurations.

20

But, how did we know what to select?

- Four ways to find the configurations:
  1. Look in a reference book ☺
  2. Constraint-based approach.
  3. Heuristics
  4. Combinatorial designs (nice URLs!)
  5. Use a pair-wise combination generator tool

21

Comparison of Methods

1. Look in a reference book
   - *CRC Handbook of Combinatorial Designs*
   - The specific number of parameters and values in your situation has to be in the book!
   - Check out URLs
2. Constraint-based approach
   - Requires solution to {0,1} integer program
   - Gives optimal solution
   - NP-complete problem – not feasible for realistic situations

22

Solution using freeware integer/linear program solver.

# parms	#value/ parm	#constraints	#variables	#configs	Run time (s)
3	2	12	8	4	< 0.01
4	2	24	16	5	0.01
5	2	40	32	6	0.70
6	2	60	64	6	16.57
7	2	84	128	6	441.21
4	3	54	81	9	0.08
5	3	90	243	13**	*

* process killed after 6.5 hours

** result at time the process was terminated.

23

Method 3: Heuristics

- The one shown here, In-Parameter Order, is due to Lei and Tai.
- Start with the first two parameters, and generate all possible combinations
- Then, add a third parameter. For the test configurations already generated, choose values for the new parameter so that the largest number of interactions are covered.
  - If there are interactions left uncovered at the end of this process, add additional configurations
- Repeat until all parameters have been added.

24

Example

- 3 parameters
  - First parameter can take values A or B
  - Second parameter can take values J or K
  - Third parameter can take values X, Y, or Z
- 12 possible configurations

25

Start with 2 parameters

Interaction elements

A	J	
A	K	
B	J	
B	K	

A		X
A		Y
A		Z
B		X
B		Y
B		Z

	J	X
	J	Y
	J	Z
	K	X
	K	Y
	K	Z

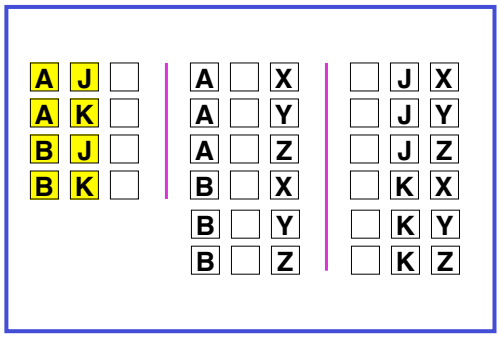
Test configurations

A	J
A	K
B	J
B	K

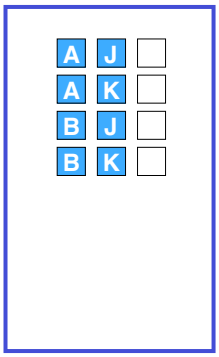
26

Add spaces for next parameter

Interaction elements



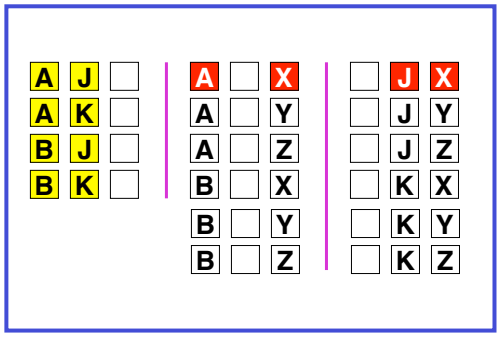
Test configurations



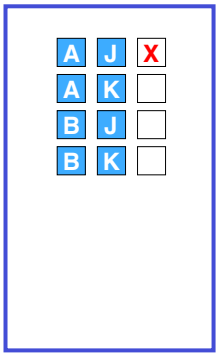
27

Try values to see which covers the most interactions

Interaction elements



Test configurations

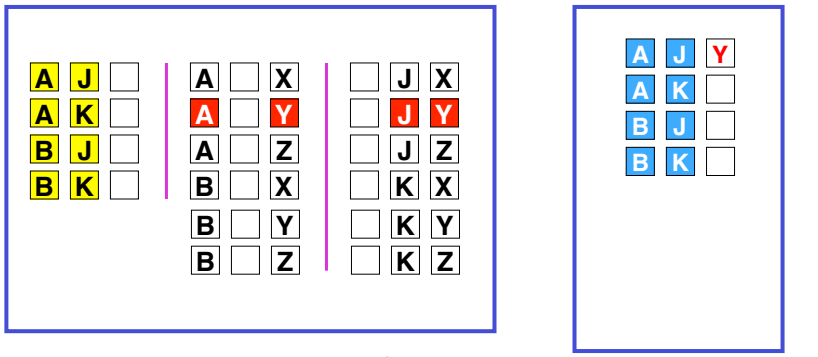


X: 2 interactions covered

28

Interaction elements

Test configurations



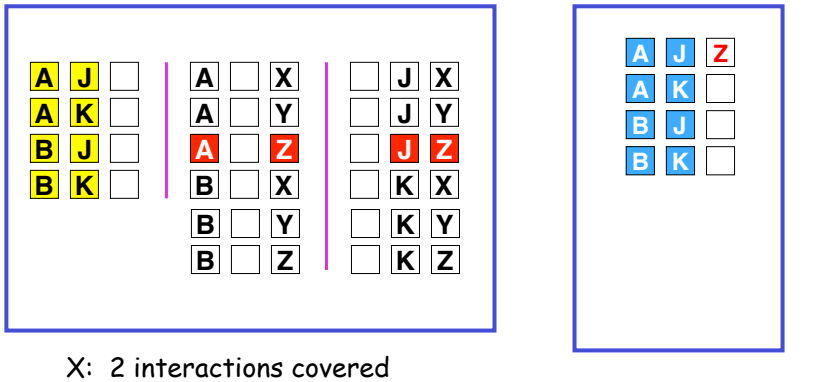
X: 2 interactions covered

Y: 2 interactions covered

29

Interaction elements

Test configurations



X: 2 interactions covered

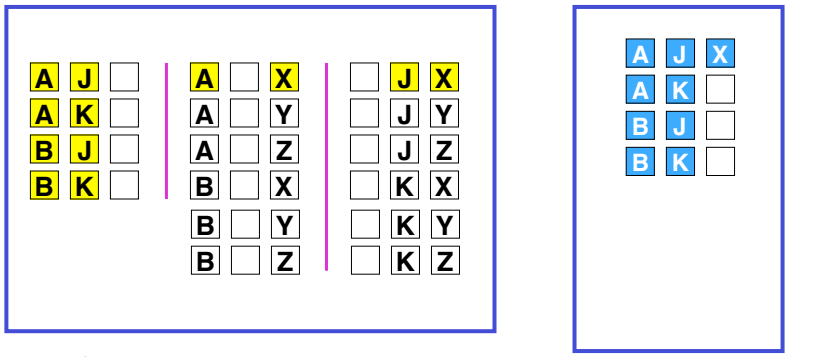
Y: 2 interactions covered

Z: 2 interactions covered

30

Interaction elements

Test configurations

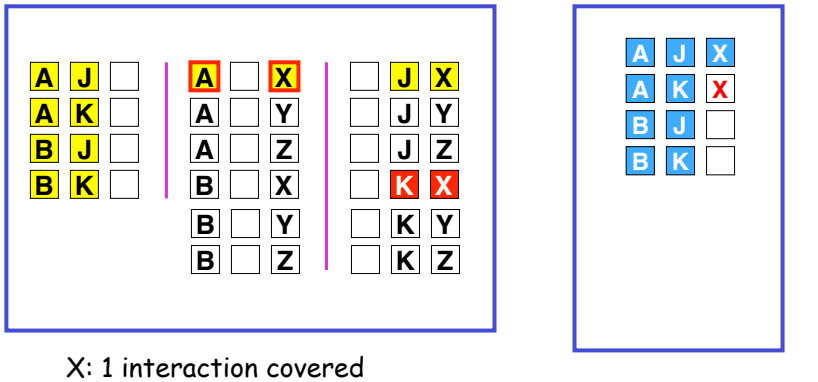


Choose X

31

Interaction elements

Test configurations

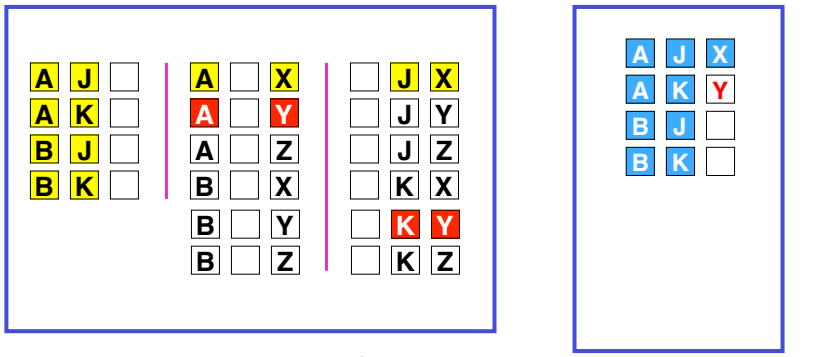


X: 1 interaction covered

32

Interaction elements

Test configurations



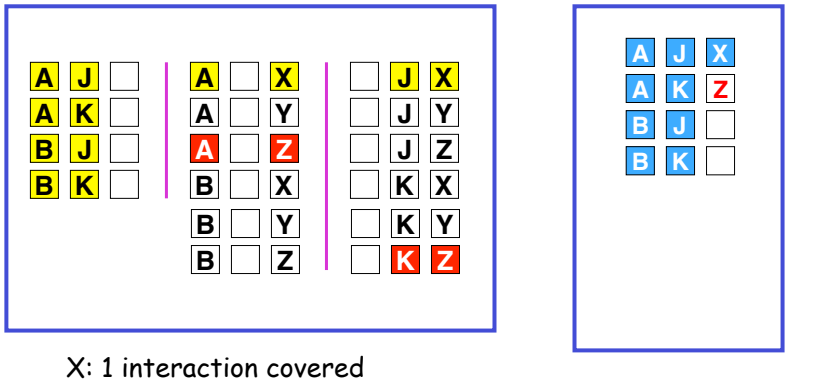
X: 1 interaction covered

Y: 2 interactions covered

33

Interaction elements

Test configurations

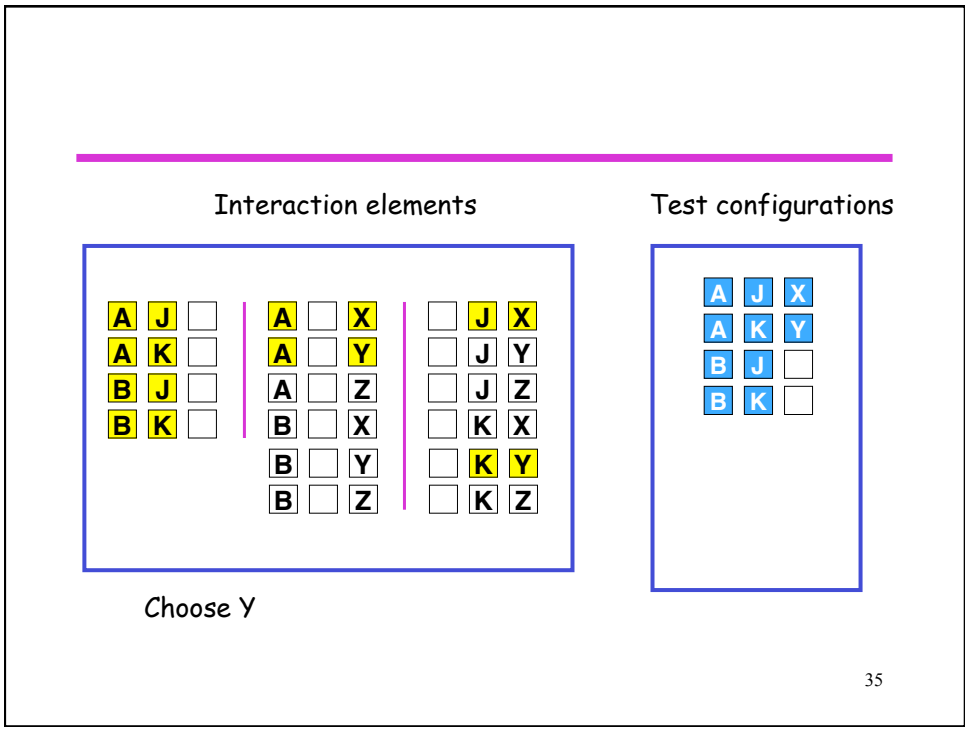


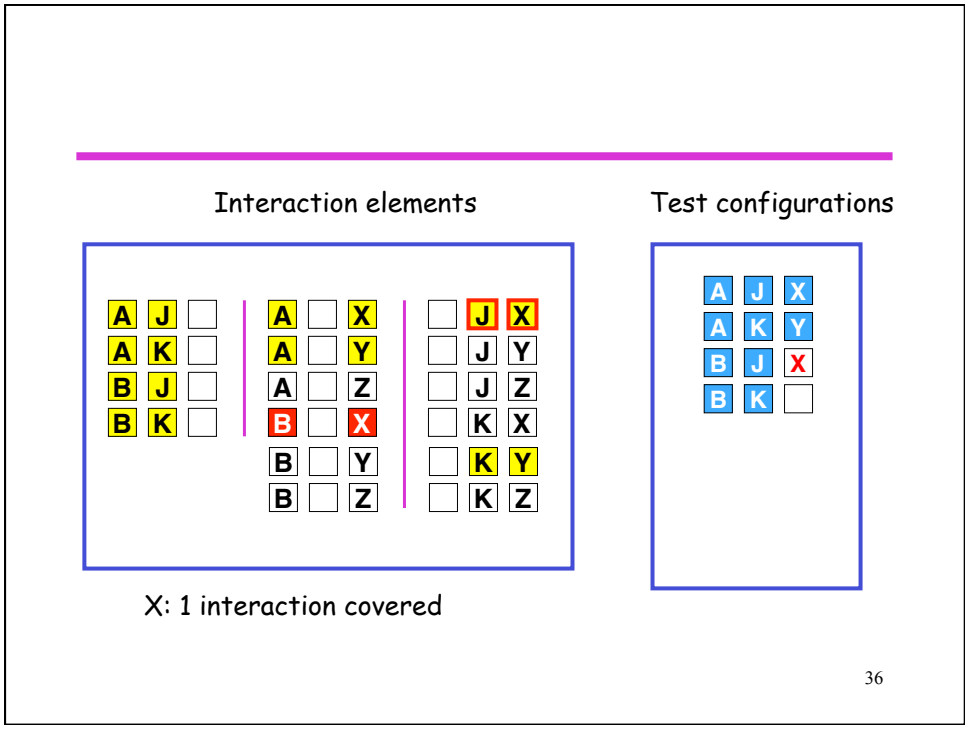
X: 1 interaction covered

Y: 2 interactions covered

Z: 2 interactions covered

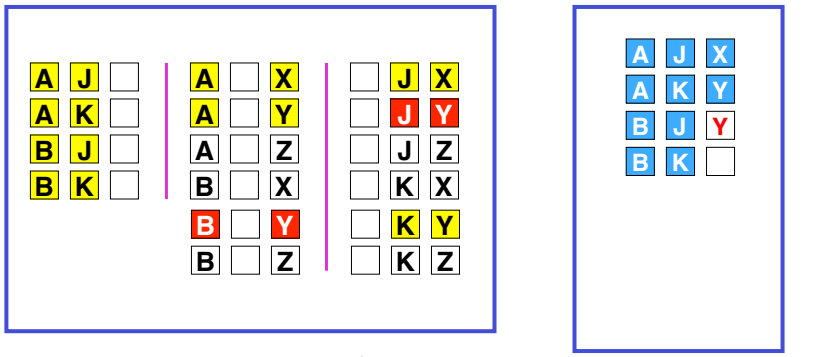
34


Interaction elements
Test configurations
A J
A K
B J
B K
A X
A Y
A Z
B X
B Y
B Z
J X
J Y
J Z
K X
K Y
K Z
A J X
A K Y
B J
B K
Choose Y
35


Interaction elements
Test configurations
A J
A K
B J
B K
A X
A Y
A Z
B X
B Y
B Z
J X
J Y
J Z
K X
K Y
K Z
A J X
A K Y
B J X
B K
X: 1 interaction covered
36

Interaction elements

Test configurations



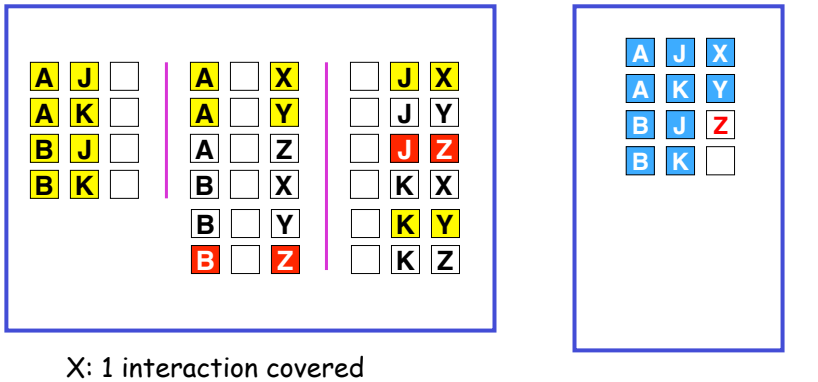
X: 1 interaction covered

Y: 2 interactions covered

37

Interaction elements

Test configurations



X: 1 interaction covered

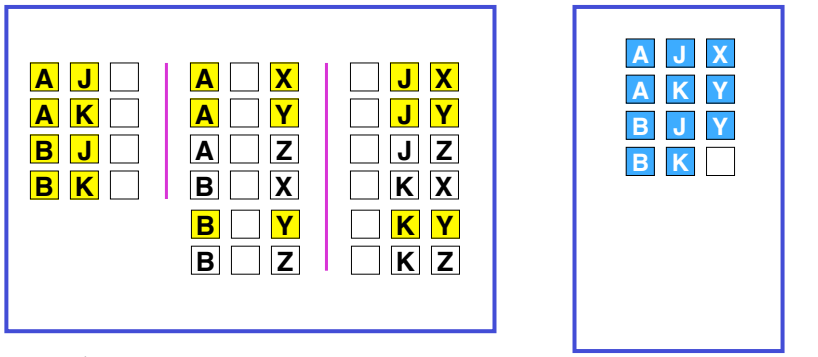
Y: 2 interactions covered

Z: 2 interactions covered

38

Interaction elements

Test configurations

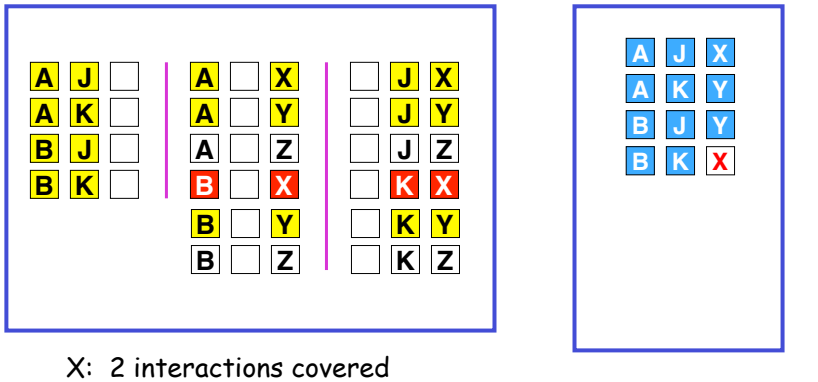


Choose Y

39

Interaction elements

Test configurations

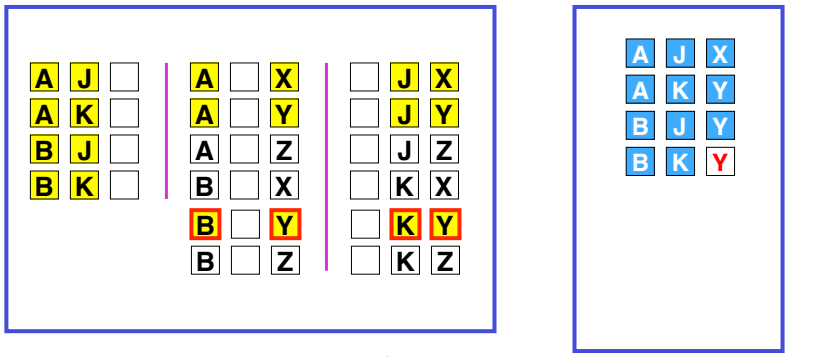


X: 2 interactions covered

40

Interaction elements

Test configurations

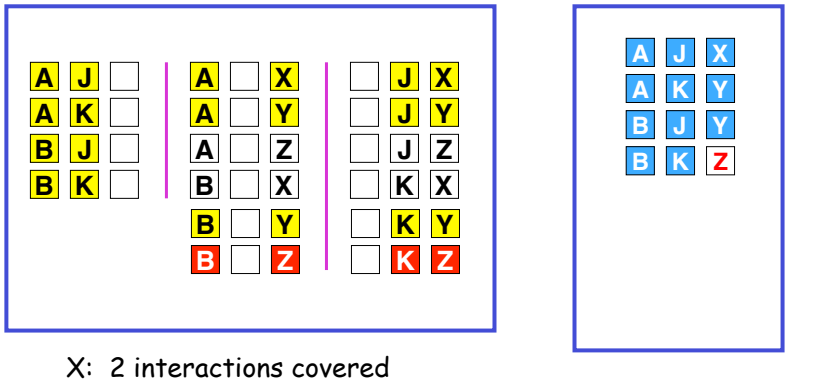


X: 2 interactions covered
Y: 0 interactions covered

41

Interaction elements

Test configurations

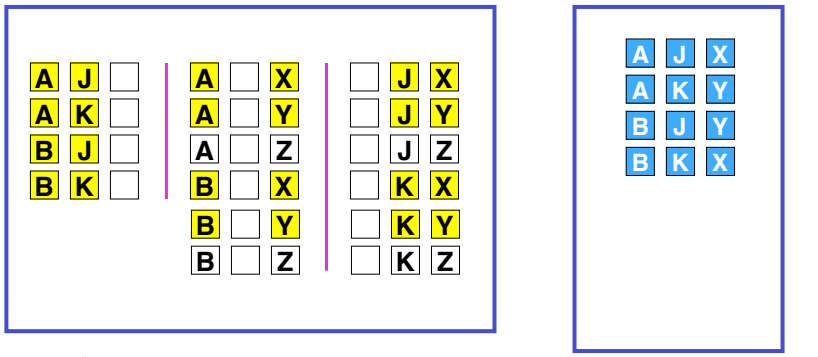


X: 2 interactions covered
Y: 0 interactions covered
Z: 2 interactions covered

42

Interaction elements

Test configurations



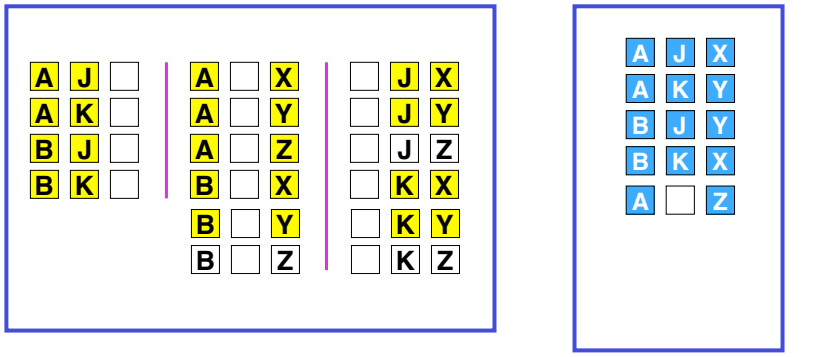
Choose X

43

Additional configurations needed to complete coverage

Interaction elements

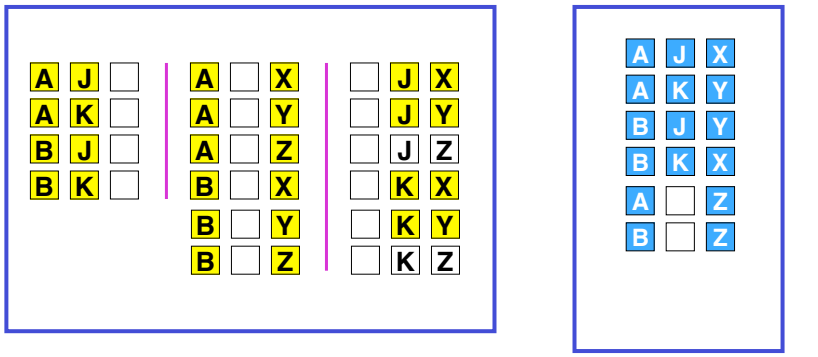
Test configurations



44

Interaction elements

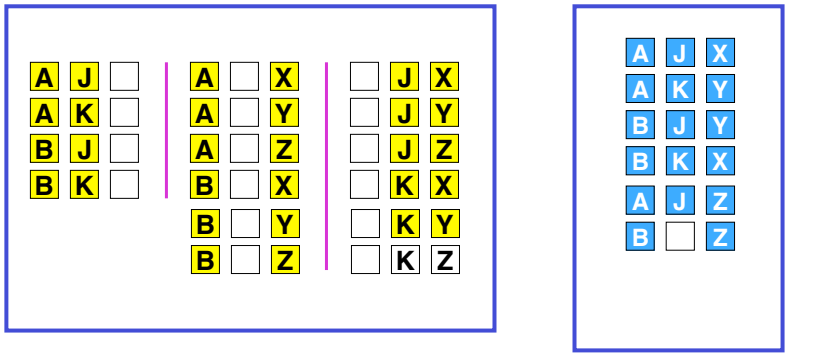
Test configurations



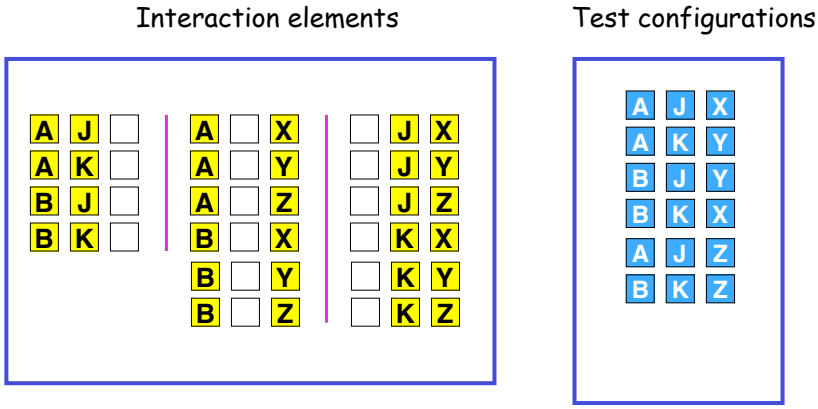
45

Interaction elements

Test configurations



46



- 6 out of 12 configurations are selected for testing

47

Method 4

- Use principles of combinatorial designs used in the design of statistical experiments.
- Check out:
  - http://csrc.nist.gov/groups/SNS/acts/documents/comparison-report.html
  - http://www.public.asu.edu/~ccolbou/src/tabby/catable.html
  - http://www.ccrwest.org/cover/sage.htmlhttp://www.ccrwest.org/cover/show_improvements.php
  - http://www.ccrwest.org/cover/low.htmlhttp://www.ccrwest.org/cover.html
  - http://www.public.asu.edu/~ccolbou/http://www.nist.gov/itl/csd/set/acts.cfm
  - http://csrc.nist.gov/groups/SNS/acts/index.html

48

Orthogonal Arrays

- Orthogonal arrays are a standard construction used for statistical experiments.
- Strength 2: select any 2 columns and all ordered pairs occur the same number of times.
  - Covers all 2-way interactions.
- Orthogonal arrays can be found in statistical tables, or can be calculated from algebraic finite fields.
  - Many existence restrictions.

49

Covering Arrays

- Definition of covering array:
  - If we select d columns, all possible ordered d-tuples occur at least once.
- A covering array of strength d will ensure than any consistent interaction problem caused by a particular combination of two elements is detected. Problems caused by an interaction of $d+1$ (or more) elements may not be detected.
- Choosing the degree of coverage defines the trade-off in risk we are making:
  - Fewer test configurations versus potential uncovered interactions.

50

A covering array

Four parameters, two values for each.

A	C	E	G
B	D	F	G
B	D	E	H
B	C	F	H
A	D	F	H

- Note that in some cases, a specific 2-way interaction appears several times, but all 2-way interactions appear at least once.

51

How to construct covering arrays?

- Various algorithms have been developed to use small orthogonal arrays as building blocks to construct covering arrays for larger cases.
- The best way to use these algorithms: use a tool that has already implemented them…

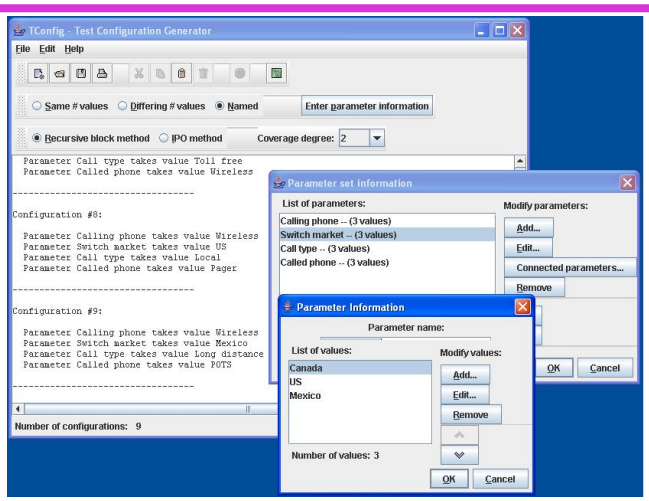
52

Method 5

- Use a combinatorial testing tool.
- Commercial:
  - AETG (Telcordia)
- Freeware:
  - TConfig (U. Ottawa)
  - Allpairs (Bach)

53

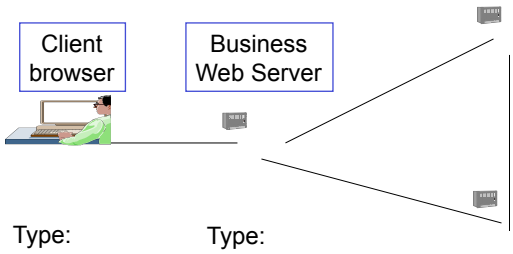
TConfig: Test configuration generator



Try it: www.site.uottawa.ca/~awilliam/TConfig.jar

54

E-commerce Example Again



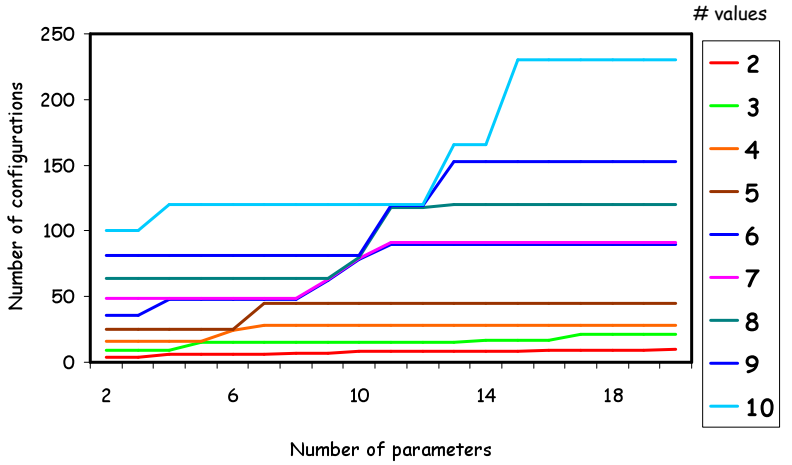
55

Strength 2 covering array

Configuration	Browser	Web Server	Payment	Data Base
1	Netscape	Glassfish	MasterCard	DB/2
2	Netscape	Apache	Visa	Oracle
3	Netscape	JBoss	AmEx	MySQL
4	Explorer	Glassfish	Visa	MySQL
5	Explorer	Apache	AmEx	DB/2
6	Explorer	JBoss	MasterCard	Oracle
7	Firefox	Glassfish	AmEx	Oracle
8	Firefox	Apache	MasterCard	MySQL
9	Firefox	JBoss	Visa	DB/2

56

Number of configurations needed for degree 2 coverage



57

Comments from testers for future work:

- Inclusion of specified or existing tests:
  - "I have some recommended configurations and I want to be sure they are included."
  - "I already have a collection of tests that are working fine, and have been developed at great expense. How do I determine which additional tests need to be added to bring the test suite to a certain level of interaction coverage?"
- Changes in set of allowed parameters and values:
  - "What additional configurations are required if…
    - … a new component is added to the system?"
    - … a new version of an existing component becomes available?"
- Dealing with forbidden combinations of values.

58