Pairwise Testing

What is Pairwise Testing?

- In computer science all-pairs testing or pairwise testing is a combinatorial method of software testing that, for each pair of input parameters to a system (typically a software algorithm) test all possible discrete combinations of those parameters.
- Using carefully chosen test vectors, this can be done much faster than an exhaustive search of all combinations of all parameters, by "parallelizing" the tests of parameter pairs.
- The most common bugs in a program are generally triggered by either a single input parameter or an interaction between pairs of parameters.
- Bugs involving interactions between three or more parameters are both progressively less common and also progressively more expensive to find---such testing has as its limit the testing of all possible inputs.

What is Pairwise Testing?

- Thus, a combinatorial technique for picking test cases like all-pairs testing is a useful cost-benefit compromise that enables a significant reduction in the number of test cases without drastically compromising functional coverage.
- In computer systems, defects usually involve a single condition, independent of any other condition in the system. If there is no problem with a device or a variable or a setting, the problem is usually occurs in that device, variable and settling alone.
- In testing, we want to be sure that we don’t miss problems based on the conflicts between two or more conditions, variables, or configurations, so we often test in combinations, in order to find the defects most efficiently.

All Pairs Technique

- Suppose that there are five variables and each represented with a letter of alphabet. Also, suppose each variable contains values from 1 to 5.
- Let’s set variable A,B,C, and D all to equal value 1. If those values are fixed, the variable E can have values from 1 to 5. We will keep track of the total number of combinations in the left most column; and our first 5 combinations are numbered as 1 to 5.

Table 1: Varying Column E only

Combination No.	A	B	C	D	E
1	1	1	1	1	1
2	1	1	1	1	2
3	1	1	1	1	3
4	1	1	1	1	4
5	1	1	1	1	5

All Pairs Technique

- Let’s set variable D all to value 2. And keep A,B, and C to 1.
- Then we obtained 5 more combinations (6-10).
- Then we set D to 3, 4 and 5 with setting E value 1 to 5 each time.
- We will get 25 combinations by changing values of D and E keeping other variable with constant value 1.

Table 2: Varying Column E only

Combination No.	A	B	C	D	E
6	1	1	1	2	1
7	1	1	1	2	2
8	1	1	1	2	3
9	1	1	1	2	4
10	1	1	1	2	5

All Pairs Technique

- In fact if there are 5 variables in a program with 5 possible states, the program would be absurdly simple, yet we have to test 3125 combinations.
- If we could test one combination per minute, then complete testing would take 7 and a half days of continuous, seven-hour-a-day tests. There’s got a way to reduce the number of tests into something that we can handle.

All Pairs Table

- Imagine that our defect depend upon checkbox A being cleared (that is. unchecked) and checkbox B being set (Checked). If we try all of the possible settings in combination with one another, we will find the defect.
- Combinallons 1, 3, and 4 work all right, but Combination 2 exhibits the problem. We'd required four tests to make sure that we had covered all possible which A and B could be found. Two variables, and two settings for each; four combinations.

Table 3: Possible states for two checkboxes

Combination No.	Box A	Box B
1	set	set
2(defect)	clear	set
3	set	clear
4	clear	clear

Orthogonal Array

- Orthogonal Array Testing is a systematic way of testing all-pair combinations of variables using orthogonal arrays.
- It significantly reduces the number of combinations of variable to test all pair combinations.
- By definition, it is a two-dimensional array constructed with special mathematical properties such that choosing only two columns in the array provides every combination of each number in the array.

Pairwise Testing Example-1

An application with simple list box with 10 elements (Let's say 0,1,2,3,4,5,6,7,8,9) along with a checkbox, radio button, Text Box and OK Button. The Constraint for the Text box is it can accept values only between 1 and 100. Below are the values that each one of the GUI objects can take :

List Box - 0,1,2,3,4,5,6,7,8,9

Check Box - Checked or Unchecked

Radio Button - ON or OFF

Text Box - Any Value between 1 and 100

Pairwise Testing Example-1

Exhaustive combination of the product is calculated.

List Box = 10

Check Box = 2

Radio Button = 2

Text Box = 100

Total Number of Test Cases using Cartesian Method : 10*2*2*100 = 4000

Total Number of Test Cases including Negative Cases will be > 4000

Pairwise Testing Example-1

Now, the idea is to bring down the number of test cases. We will first try to find out the number of cases using the conventional software testing technique. We can consider the list box values as 0 and others as 0 is neither positive nor negative. Radio button and check box values cannot be reduced, so each one of them will have 2 combinations (ON or OFF). The Text box values can be reduced into three inputs (Valid Integer, Invalid Integer, Alpha-Special Character).

Now, we will calculate the number of cases using software testing technique, 2*2*2*3 = 24 (including negative cases).

Pairwise Testing Example-1

Now, we can still reduce the combination further into All-pairs technique.

Step 1: Order the values such that one with most number of values is the first and the least is placed as the last variable.

Step 2: Now start filling the table column by column. List box can take 2 values.

Step 3: The Next column under discussion would be check box. Again Check box can take 2 values.

Step 4: Now we need to ensure that we cover all combinations between list box and Check box.

Step 5: Now we will use the same strategy for checking the Radio Button. It can take 2 values.

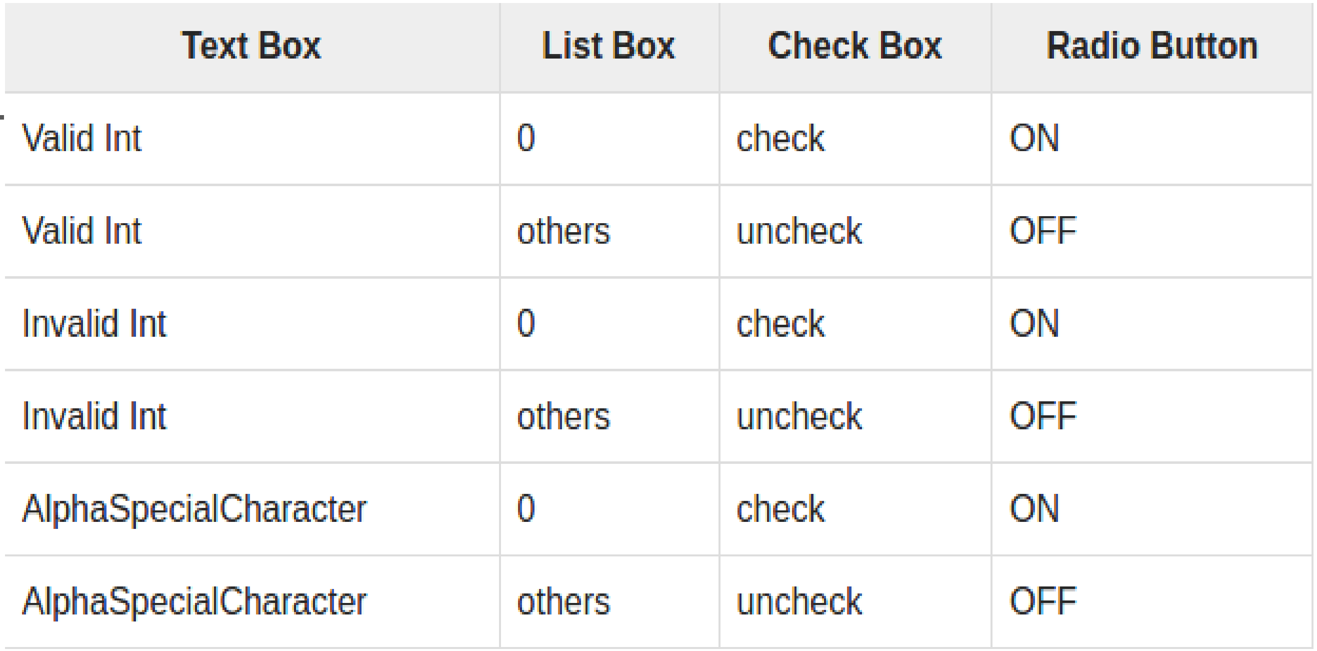
Step 6: Verify if all the pair values are covered as shown in the table below.

Pairwise Testing Example-1

Exhaustive Combination results in > 4000 Test Cases.

Conventional Software Testing technique results in 24 Test Cases.

Pair Wise Software Testing technique results in just 6 Test Cases.



Text Box	List Box	Check Box	Radio Button
Valid Int	0	check	ON
Valid Int	others	uncheck	OFF
Invalid Int	0	check	ON
Invalid Int	others	uncheck	OFF
AlphaSpecialCharacter	0	check	ON
AlphaSpecialCharacter	others	uncheck	OFF

Pairwise Testing Example-2

'Enabled', 'Choice Type' and 'Category' have a choice range of 2, 3 and 4, respectively. An exhaustive test would involve 24 tests (2 x 3 x 4). Multiplying the two largest values (3 and 4) indicates that a pair-wise tests would involve 12 tests.

Parameter Name	Value-1	Value-2	Value-3	Value-4
'Enabled'	True	False	-	-
'Choice Type'	1	2	3	-
'Category'	A	B	C	D

Pairwise Testing Example-2

The pairwise test cases, generated by Microsoft's "pict" tool, are shown below.

Enabled	Choice type	Category
True	3	a
True	1	d
False	1	c
False	2	d
True	2	c
True	1	b

Enabled	Choice type	Category
False	2	a
False	1	a
False	3	b
True	2	b
True	3	d
False	3	c

Pairwise Testing Limitations

The pairwise testing technique has some limitations as well.

- It fails when the values selected for testing are incorrect.
- It fails when highly probable combinations get too little attention.
- It fails when interactions between the variables are not understood well.

Pairwise Testing Tools:

Tools are available that applies the all-pairs testing technique that facilitates us to effectively automate the Test Case Design process by generating a compact set of parameter value choices as the desired Test Cases. Some well-known tools from the industry are:

- PICT – ‘Pairwise Independent Combinatorial Testing’, provided by Microsoft Corp.
- IBM FoCuS – ‘Functional Coverage Unified Solution’, provided by IBM.
- ACTS – ‘Advanced Combinatorial Testing System’, provided by NIST, an agency of the US Government.

- Hexawise
- Jenny
- Pairwise by Inductive AS
- VPTag free All-Pair Testing Tool

Conclusion:

- The pairwise testing technique can dramatically reduce the number of combinations to be covered but remains very effective in terms of fault detection. It is indeed a smart test design technique that guarantees a win-win situation for both test effort and test effectiveness.
- During the Test planning phase of software testing, the Pairwise testing technique should always be taken into consideration. Either we are doing it manually or using any tool to generate test cases, it becomes a necessary component of the test plan because it, in turn, affects Test estimation.