SE-3002
SOFTWARE QUALITY ENGINEERING

RUBAB JAFFAR
RUBAB.JAFFAR@NU.EDU.PK

Part II-Software Testing

Functional Testing

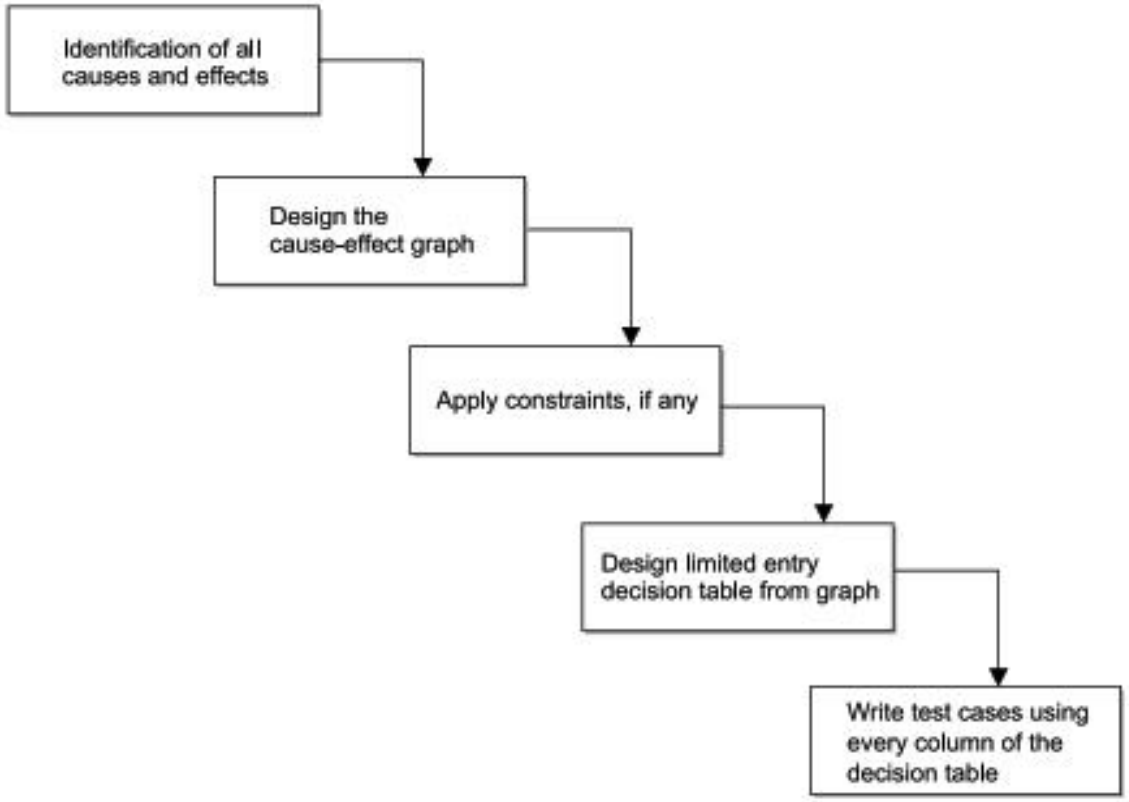
Lecture # 16, 17, 18
18, 21 Oct

TODAY'S OUTLINE

- Cause Effect Testing
- Pairwise Testing

CAUSE-EFFECT GRAPHING TECHNIQUE

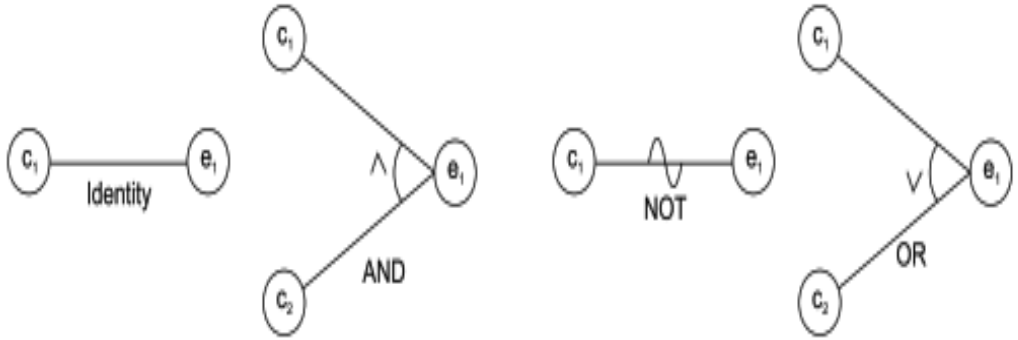
- Popular technique for small programs and considers the combinations of various inputs.
- Two terms: **Causes and Effects**, which are nothing but inputs and outputs respectively.
- SRS document is used for the identification of causes and effects.
- A list is prepared for all causes and effects.



Steps for the generation of test cases

DESIGN OF CAUSE-EFFECT GRAPH

- Each node represents either true or false state and may be assigned 1 and 0 value respectively.



Basic notations used in cause-effect graph

(a) Identity: This function states that if c_1 is 1, then e_1 is 1; else e_1 is 0.
(b) NOT: This function states that if c_1 is 1, then e_1 is 0; else e_1 is 1.
(c) AND: This function states that if both c_1 and c_2 are 1, then e_1 is 1; else e_1 is 0.
(d) OR: This function states that if either c_1 or c_2 is 1, then e_1 is 1; else e_1 is 0.

The AND and OR functions are allowed to have any number of inputs.

USE OF CONSTRAINTS IN CAUSE-EFFECT GRAPH

(a) Exclusive

The Exclusive (E) constraint states that at most one of c_1 or c_2 can be 1 (c_1 or c_2 cannot be 1 simultaneously). However, both c_1 and c_2 can be 0 simultaneously.

(b) Inclusive

The Inclusive (I) constraints states that at least one of c_1 or c_2 must always be 1. Hence, both cannot be 0 simultaneously. However, both can be 1.

(c) One and Only One

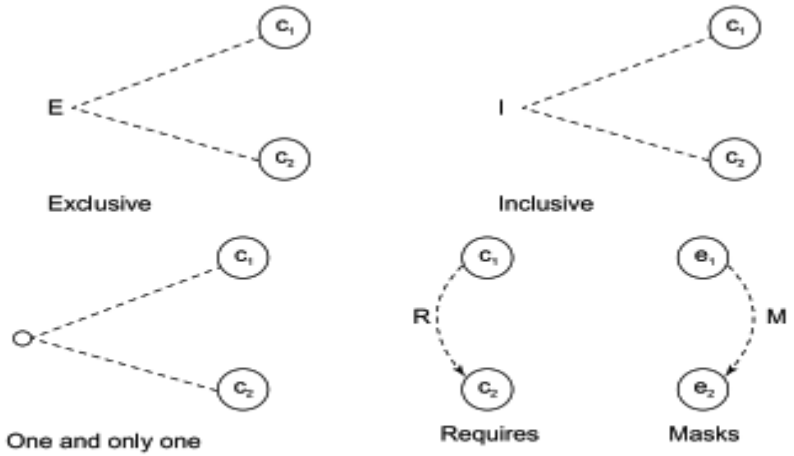
The one and only one (O) constraint states that one and only one of c_1 and c_2 must be 1.

(d) Requires

The requires (R) constraint states that for c_1 to be 1, c_2 must be 1; it is impossible for c_1 to be 1 if c_2 is 0.

(e) Mask

This constraint is applicable at the effect side of the cause-effect graph. This states that if effect e_1 is 1, effect e_2 is forced to be 0.



Constraint symbols for any cause-effect graph

EXAMPLE

- Consider the example of keeping the record of marital status and number of children of a citizen. The value of marital status must be 'U' or 'M'. The value of the number of children must be digit or null in case a citizen is unmarried. If the information entered by the user is correct then an update is made. If the value of marital status of the citizen is incorrect, then the error message 1 is issued. Similarly, if the value of number of children is incorrect, then the error message 2 is issued.

causes are:

c_1: marital status is 'U'

c_2: marital status is 'M'

c_3: number of children is a digit

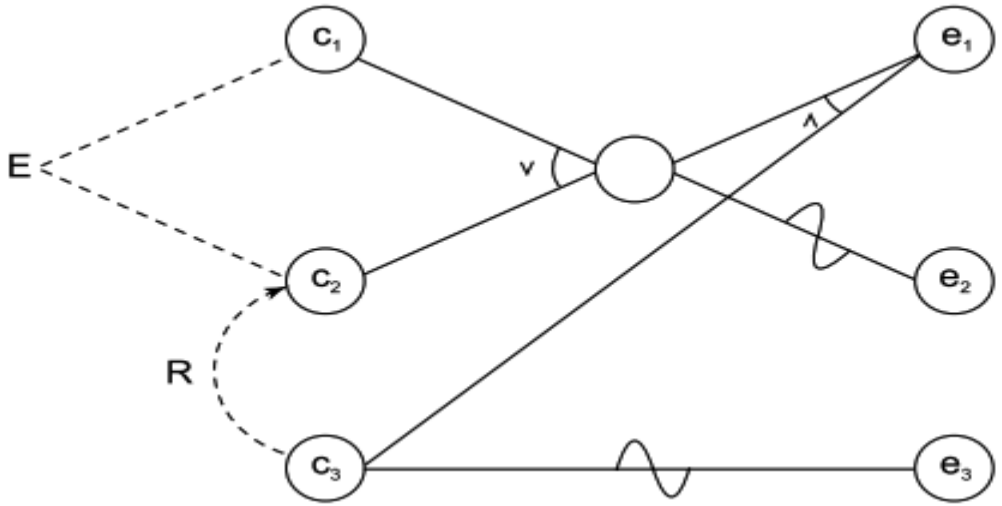
effects are:

e_1: updation made

e_2: error message 1 is issued

e_3: error message 2 is issued

SOLUTION



APPLICABILITY

- Cause-effect graphing considers dependency of inputs using some constraints.
- Effective only for small programs because, as the size of the program increases, the number of causes and effects also increases and thus complexity of the cause-effect graph increases.
- For large-sized programs, a tool may help us to design the cause-effect graph with the minimum possible complexity.
- Limited applications in unit testing and hardly any application in integration testing and system testing.

EXAMPLE

- A tourist of age greater than 21 years and having a clean driving record is supplied a rental car. A premium amount is also charged if the tourist is on business, otherwise it is not charged. If the tourist is less than 21 year old, or does not have a clean driving record, the system will display the following message: "Car cannot be supplied"
- Draw the cause-effect graph and generate test cases.

SOLUTION

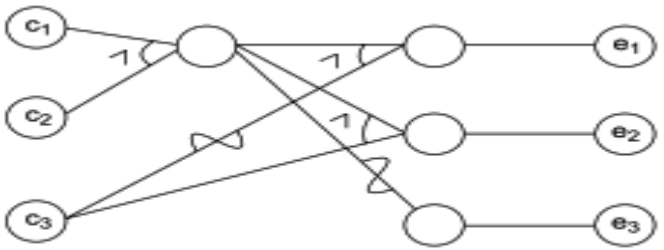


Figure 2.15. Cause-effect graph of rental car problem

Table 2.45. Decision table of rental car problem

Conditions	1	2	3	4
c_1 : Over 21 ?	F	T	T	T
c_2 : Driving record clean ?	-	F	T	T
c_3 : On Business ?	-	-	F	T
e_1 : Supply a rental car without premium charge			X	
e_2 : Supply a rental car with premium charge				X
e_3 : Car cannot be supplied	X	X		

Table 2.46. Test cases of the given decision table

Test Case	Age	Driving_record_clean	On_business	Expected Output
1.	20	Yes	Yes	Car cannot be supplied
2.	26	No	Yes	Car cannot be supplied
3.	62	Yes	No	Supply a rental car without premium charge
4.	62	Yes	Yes	Supply a rental car with premium charge.

PAIR-WISE TESTING

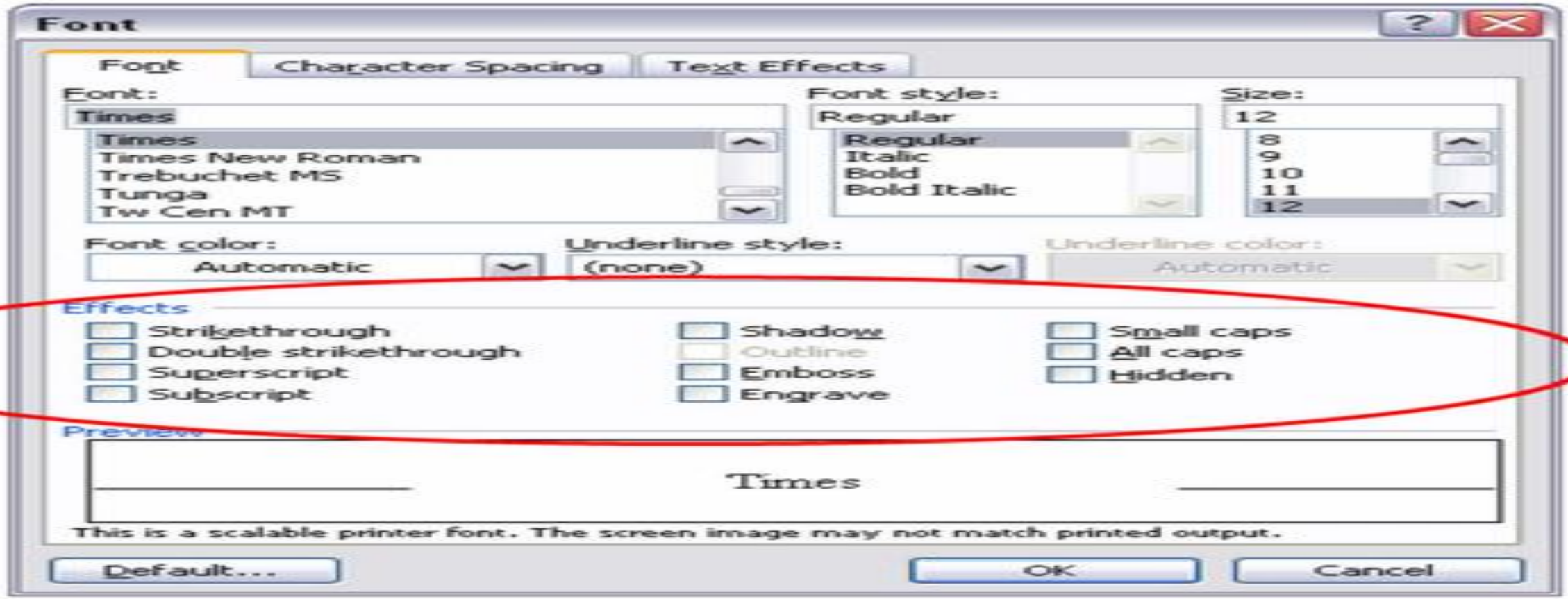
- **Pairwise Testing** is a test design technique that delivers hundred percent test coverage.
- *A black-box test design technique in which test cases are designed to execute all possible discrete combinations of each pair of input parameters.*
- Techniques like boundary value analysis and equivalence partitioning can be useful to identify the possible values for individual factors. But it is impractical to test all possible combinations of values for all those factors. So instead **a subset of combinations is generated** to satisfy all factors.
- All-Pairs technique is very helpful for designing tests for applications involving multiple parameters. Tests are designed such that for each pair of input parameters to a system, there are all possible discrete combinations of those parameters. The test suite covers all combinations; therefore it is not exhaustive yet very effective in finding bugs

EXAMPLE

- 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
- Exhaustive combination of the product B 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

SOME EXAMPLES

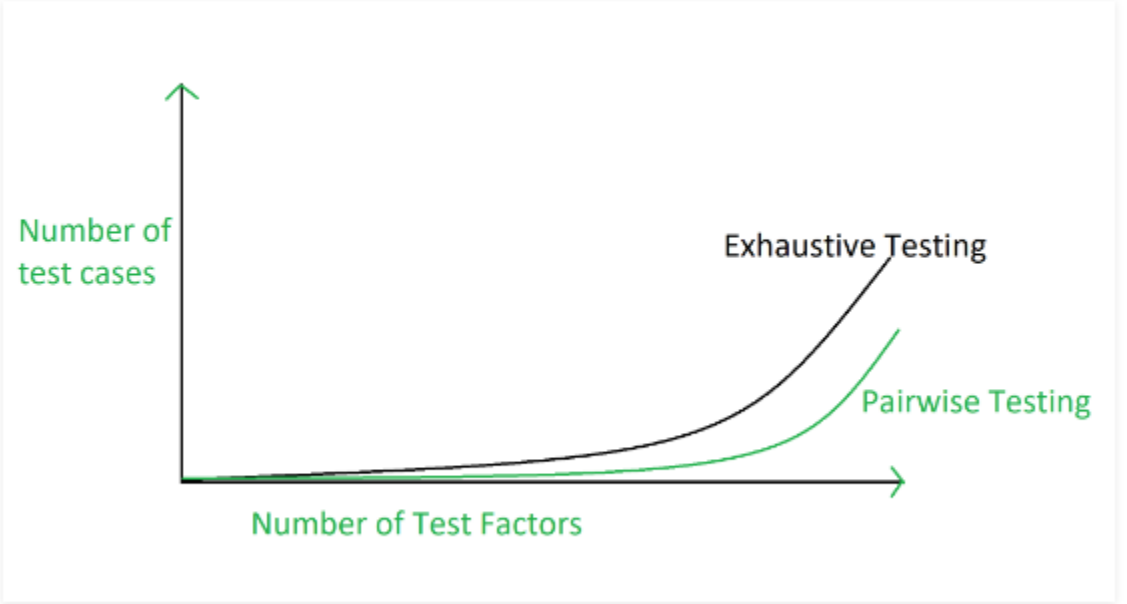
- Suppose we have a system with on-off switches:


Font
Font
Character Spacing
Text Effects
Font:
Times
Times
Times New Roman
Trebuchet MS
Tunga
Tw Cen MT
Font style:
Regular
Regular
Italic
Bold
Bold Italic
Size:
12
8
9
10
11
12
Font color:
Automatic
Underline style:
(none)
Underline color:
Automatic
Effects
Strikethrough
Double strikethrough
Superscript
Subscript
Shadow
Outline
Emboss
Engrave
Small caps
All caps
Hidden
Preview
Times
This is a scalable printer font. The screen image may not match printed output.
Default...
OK
Cancel

SOME MORE EXAMPLES

- **Car Ordering Application:**
- The car ordering application allows for Buying and Selling cars. It should support trading in Delhi and Mumbai.
- The application should have registration numbers, may be valid or invalid. It should allow the trade of following cars: BMW, Audi, and Mercedes.
- Two types of booking can be done: E-booking and In Store.
- Orders can be placed only during trading hours.

GRAPHICAL REPRESENTATION OF PAIRWISE TESTING



ADVANTAGES & DISADVANTAGES OF PAIRWISE TESTING

- Pairwise testing reduces the number of execution of test cases.
- Pairwise testing increases the test coverage almost up to hundred percentage.
- Pairwise testing increases the defect detection ratio.
- Pairwise testing takes less time to complete the execution of the test suite.
- Pairwise testing reduces the overall testing budget for a project.

- Pairwise testing is not beneficial if the values of the variables are inappropriate.
- In pairwise testing it is possible to miss the highly probable combination while selecting the test data.
- In pairwise testing, defect yield ratio may be reduced if a combination is missed.
- Pairwise testing is not useful if combinations of variables are not understood correctly.

That is all