Software Testing Methodologies

BITS Pilani

Prashant Joshi

Module 3: Agenda

Module 3: Specification Based Testing – (1/2)

Topic 3.1	Specification Based Testing – Overview
Topic 3.2	Equivalence Class
Topic 3.3	Boundary Value Analysis
Topic 3.4	Examples & Case Study

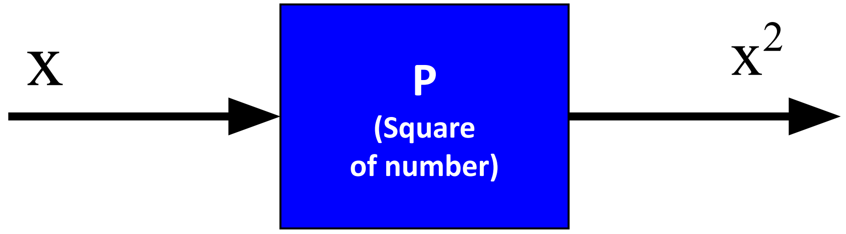
innovate achieve lead

BITS Pilani
Pilani Campus

Topic 3.1: Specification Based Testing – Overview

The Concept

- A black box test technique
- Based on specifications
- Independent of implementation
- Focus
    - Functional testing
    - Behaviour
    - Input & corresponding output



Implementation may be,

A. Multiplication *(x*x)*

B. successive addition *(x+x… x times)*

Approaches & “View”

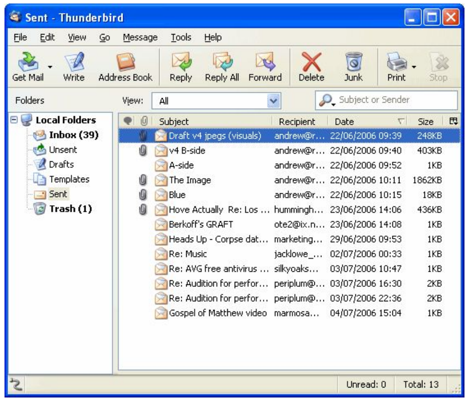
- Purpose is to uncover defects
- Demonstrate the system works (Treat this as a by-product!)
- Validate that it functions per specifications
- Works as specified – always!

Perspectives

- Customer/Client
- Alpha/Beta User
- End User/Consumer
- Development Engineer
- Architect
- Product Manager
- Maintenance Engineer
- …

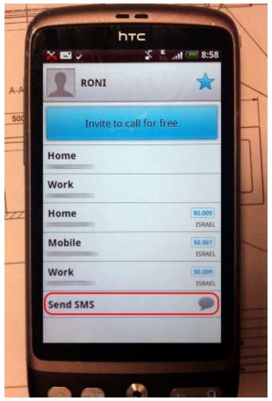
Examples

- Automated Teller Machine
- Tea/Coffee Vending Machine
- Washing Machine
- Contacts – Mobile Phone Application
- Messaging – Mobile Phone Application
- Email – Webmail/App/Client
- …


Sent - Thunderbird
File Edit View Go Message Tools Help
Get Mail Write Address Book Reply Reply All Forward Delete Junk Print Stop
Folders
View: All
Subject or Sender
Local Folders
Inbox (39)
Unsent
Drafts
Templates
Sent
Trash (1)

Subject	Recipient	Date	Size
Draft v4 jpegs (visuals)	andrew@r…	22/06/2006 09:39	248KB
v4 B-side	andrew@r…	22/06/2006 09:40	403KB
A-side	andrew@r…	22/06/2006 09:52	1KB
The Image	andrew@r…	22/06/2006 10:11	1862KB
Blue	andrew@r…	22/06/2006 10:15	18KB
Hove Actually Re: Los …	hummingh…	23/06/2006 14:06	436KB
Berkoff's GRAFT	ote2@ix.n…	23/06/2006 14:08	1KB
Heads Up - Corpse dat…	marketing…	29/06/2006 09:53	1KB
Re: Music	jacklowe_…	02/07/2006 00:33	1KB
Re: AVG free antivirus …	silkyoaks…	03/07/2006 10:47	1KB
Re: Audition for perfor…	periplum@…	03/07/2006 16:30	2KB
Re: Audition for perfor…	periplum@…	03/07/2006 22:36	2KB
Gospel of Matthew video	marmosa…	04/07/2006 15:04	1KB

Unread: 0 Total: 13



Software Testing Methodologies

BITS Pilani

Prashant Joshi

innovate achieve lead

BITS Pilani
Pilani Campus

Topic 3.2: Equivalence Class Partitioning

Examples

Problem 1

- Design Test Cases for a Software Program that takes in an input of up to 1000 numbers, finds the maximum and output is the max number

Problem 2

- Design and Discuss test cases for a function returns the max of 3 numbers. The numbers must be integer, else it returns an error

Equivalence Class

- What is an equivalence class?
- How is it useful to us as test designers?

Equivalence Class

- EC forms a partition of a set (input domain), where partition refers to a collection of mutually disjoint subsets (subdomains) when the union is an entire set
- Two important implications
  - The fact that the entire set is represented provides a form of <u>completeness</u>
  - The disjointedness ensures a form of <u>non-redundancy</u>

Equivalence Class

Reduces the potential redundancy

- The subsets are determined by an Equivalence relation, the elements have something in common
- Idea of EC is to identify (at least) one test case from each EC

Choice of EC is a challenge!

EC Types

- **Equivalence Classes (EC) - Types**
- Weak Normal (WN)
- Strong Normal (SN)
- Weak Robust (WR)
- Strong Robust (SR)

Types which ensure that we choose the "correct" set of test cases from the ECs we come up with

EC - Example

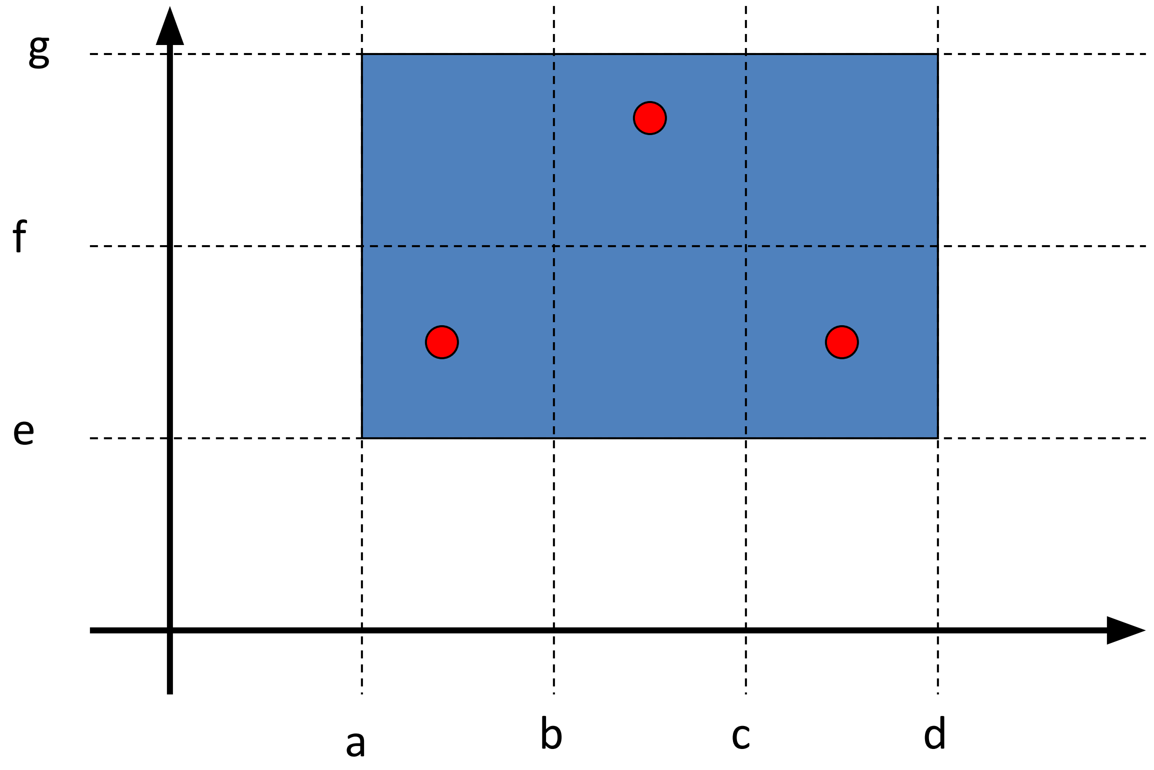
- A program takes 2 inputs x1 and x2
  - a <= x1 <= d
  - e <= x2 <= g
- We have the intervals
  - [a, b), [b, c), [c, d] □ x1
  - [e, f), [f, g] □ x2

  - [□ closed interval endpoint
  - (□ open interval endpoint
  - < > □ Ordered pair
  - ( ) □ Unordered pair

EC – Weak Normal

- One variable from each EC
- A systematic way of deriving the EC
- Same number of weak EC test cases as classes in the partition with the largest number of subsets
- Based on a single fault assumption
- Testing valid subdomains
- Assumption
  - Input variables are independent
  - One dimensional valid subdomains
- Selects tests from one dimensional (one variable) subdomains

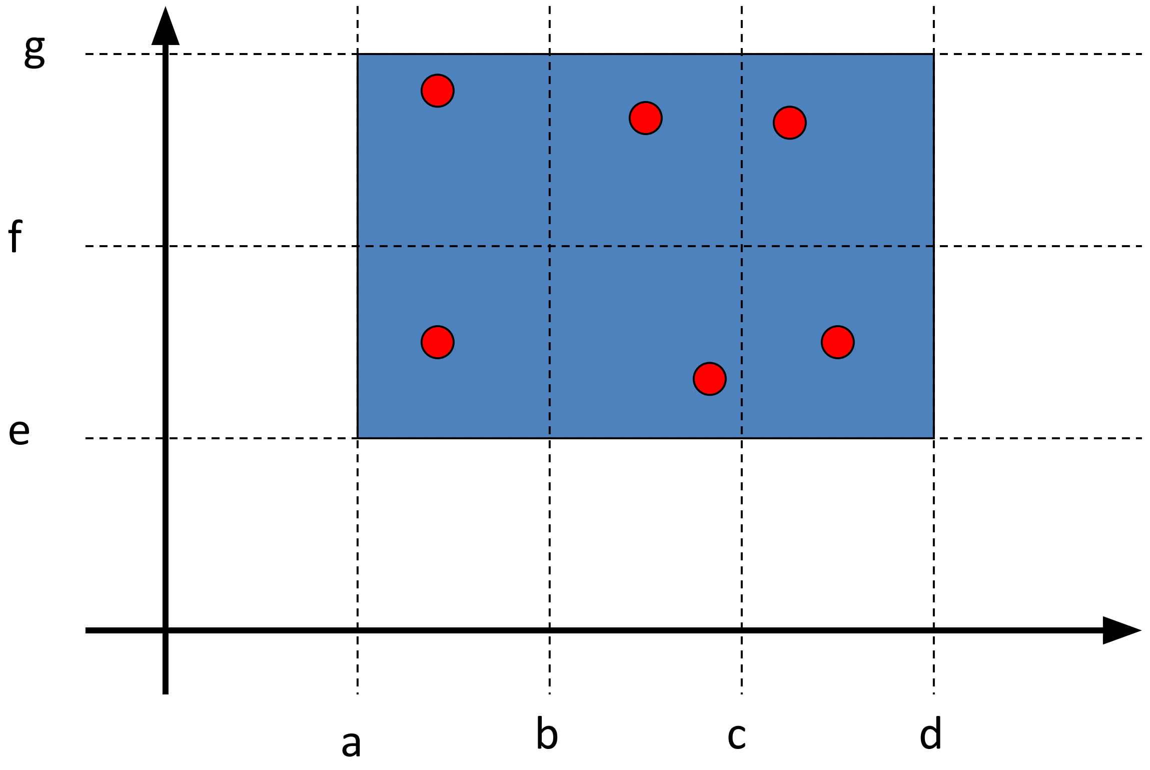
EC – Weak Normal



EC – Strong Normal

- Based on a multiple fault assumption
- We derive test cases out of the Cartesian product of equivalence classes
- Notion of “completeness”
- Testing valid subdomains
- Assumption
  - Input variables are related
  - Multidimensional subdomains. (Example)
- Test selection: Select at least one test from each of the multidimensional sub domain

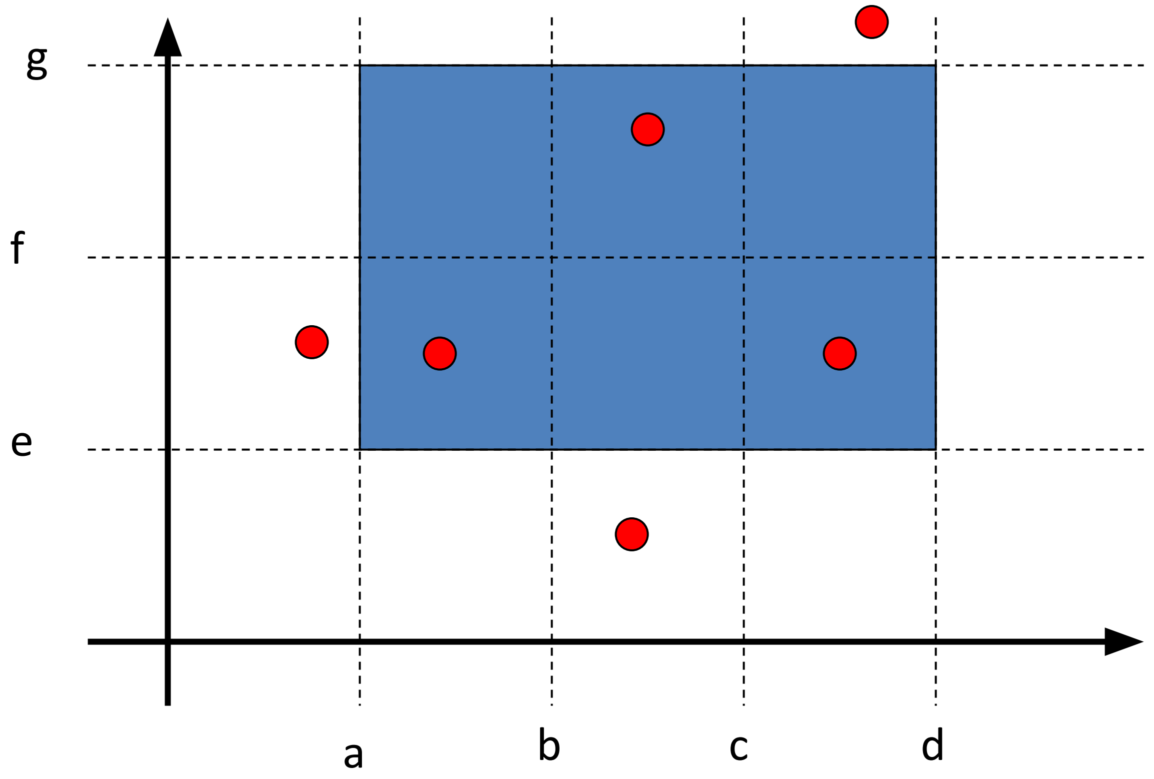
EC – Strong Normal



EC – Weak Robust

- Weak Robust is counter-intuitive.
- Robust comes from the consideration of invalid values
- Weak refers to the single fault assumption
- A test case should have one invalid value and the remaining values should be valid
- One dimensional invalid subdomains

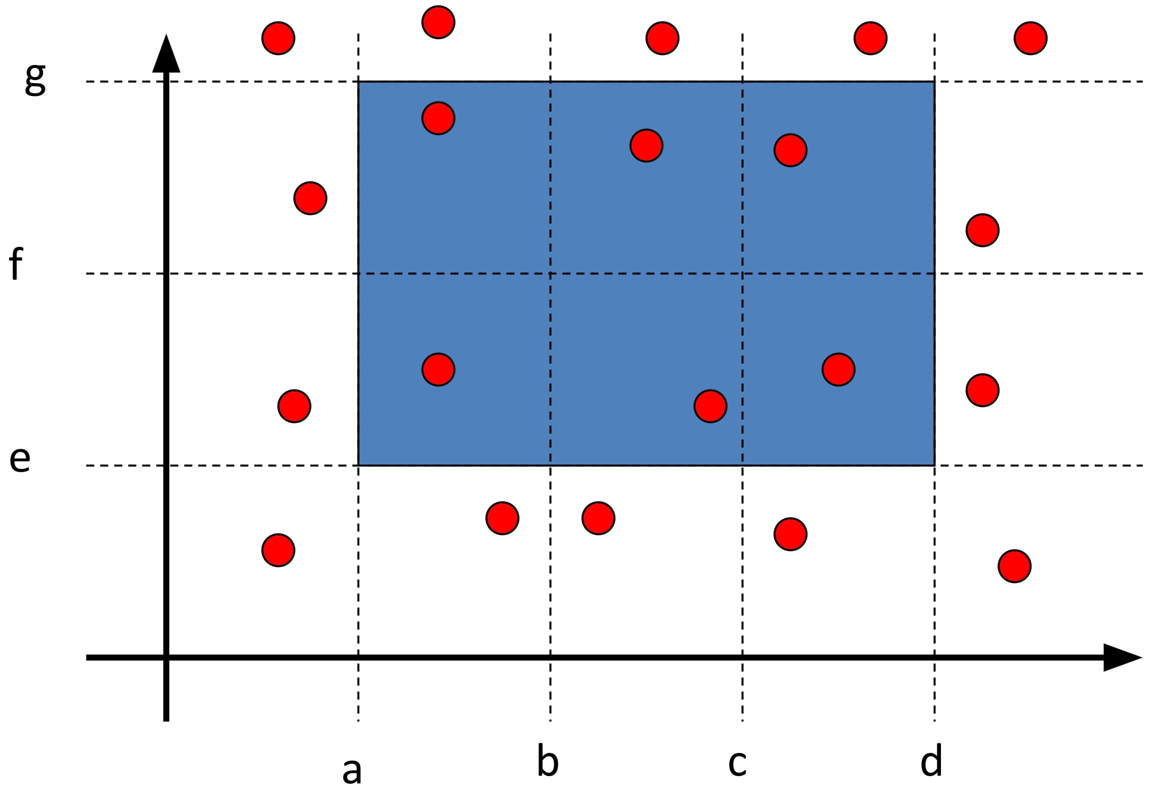
EC – Weak Robust



EC – Strong Robust

- Robust comes from consideration of invalid values for inputs
- Strong refers to the multiple fault assumption

EC – Strong Robust



EC - Characteristics

- A group forms a EC if
  - They all test the same thing
  - If one test case catches a defect, the others probably will too
  - If one test case doesn't catch a defect, the others probably won't either
- What makes us consider them as equivalent
  - They involve the same input variable
  - They result in similar operations in the program
  - They affect the same output variable
  - None force the program to do error handling or all of them do

Recommendations for Identifications of EC

- Equivalence class for invalid inputs
- Looks for Range in numbers
- Look for membership in a group
- Analyse responses to lists and menus
- Looks for variables that must be equal
- Create time-determined equivalence classes
- Look for equivalent output events
- Look for variable groups that must calculate to a certain value or range
- Look for equivalent operating environments

Ref: Testing Computer Software, Kaner, Falk and Nguyen, Chapter 7

Software Testing Methodologies

BITS Pilani

Prashant Joshi

innovate achieve lead

BITS Pilani
Pilani Campus

Topic 3.3: Boundary Value Analysis

Boundary Value Analysis

- Boundary Value Analysis focuses on the boundary of the input space to identify test cases
  - Rationale is, errors tend to occur near the extreme value of the input variable
- Examples
  - Loop counters off by 1
  - Inputs at the boundary of ranges. 10 < x < 100

Boundary Value Analysis

- Idea of BVA is to use input variable values at
  - Their minimum
  - Just above the minimum
  - A nominal value
  - Just below their maximum
  - At their maximum

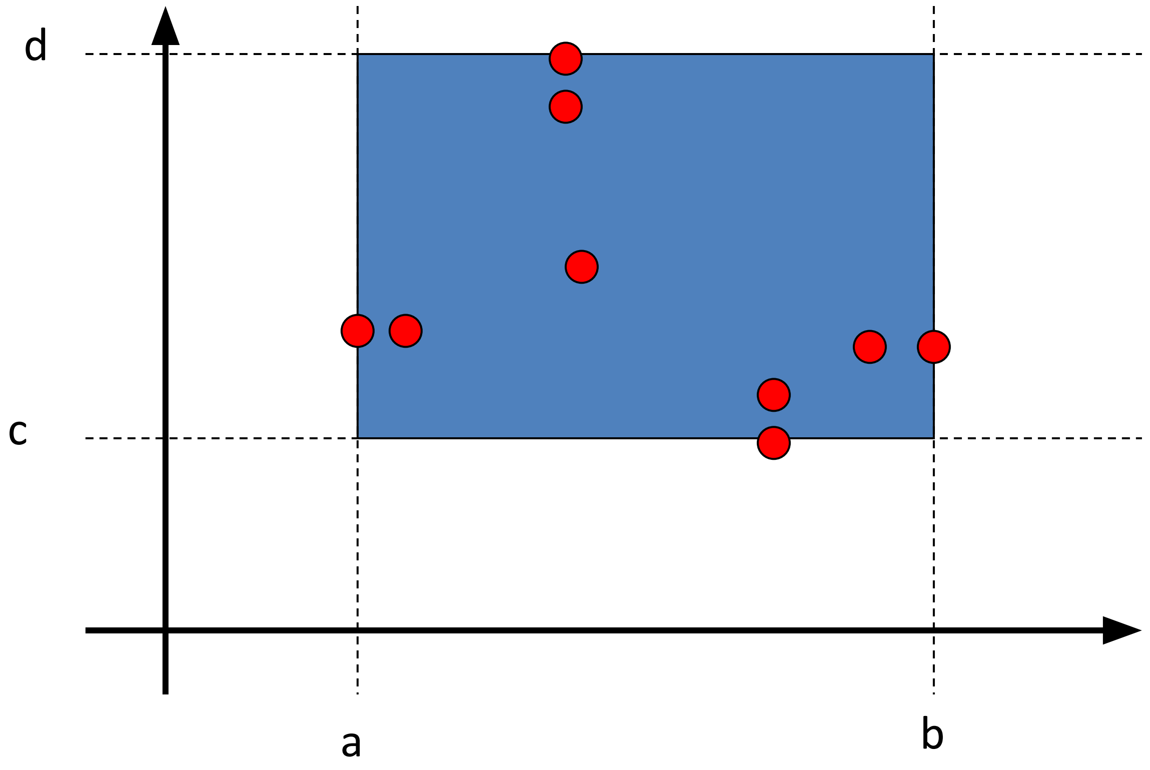
BVA – Explore the types

Example

- A program takes 2 inputs x1 and x2
  - a <= x1 <= b
  - c <= x2 <= d
- We have the intervals
  - [a, b] □ x1
  - [c, d] □ x2

BVA

BVA test cases for a function of two variables – single fault assumption



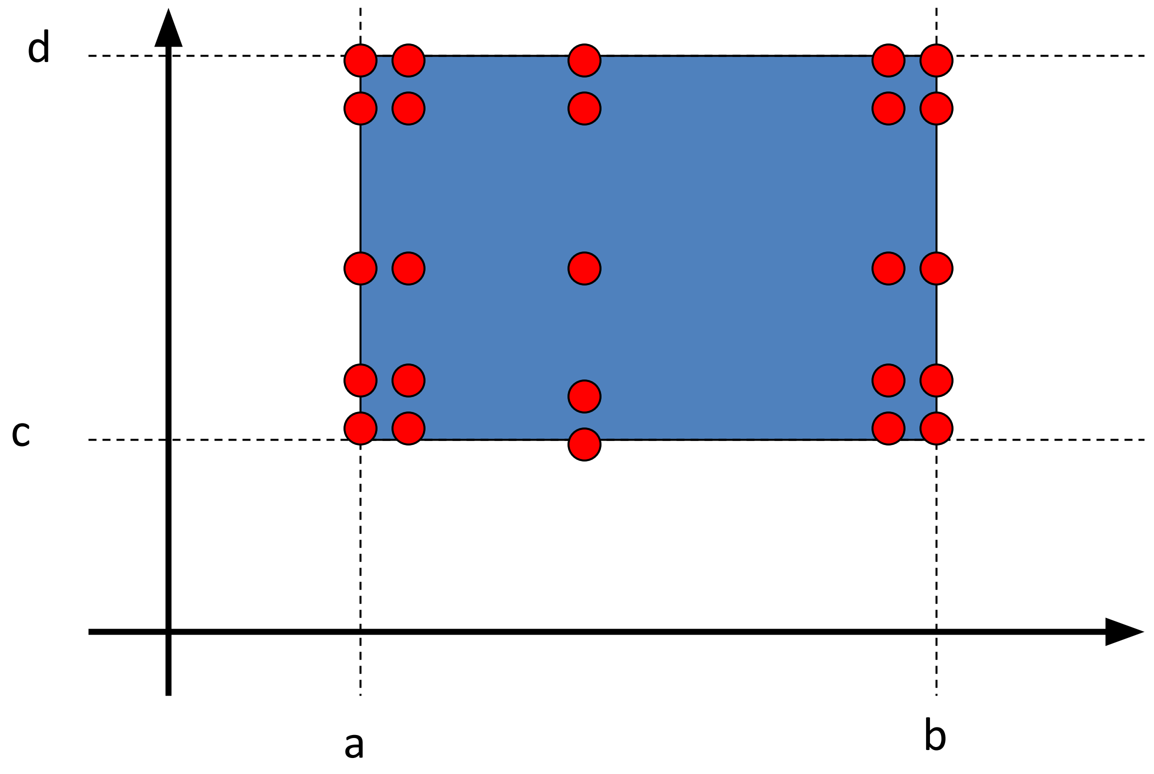
Generalising BVA

- **Two ways**
  - Number of input variables
  - Ranges
- **Variable generalization**
  - Hold one at the nominal value and let the other variable assume min, min+, nom, max- and max. i.e. 4n+1 test cases

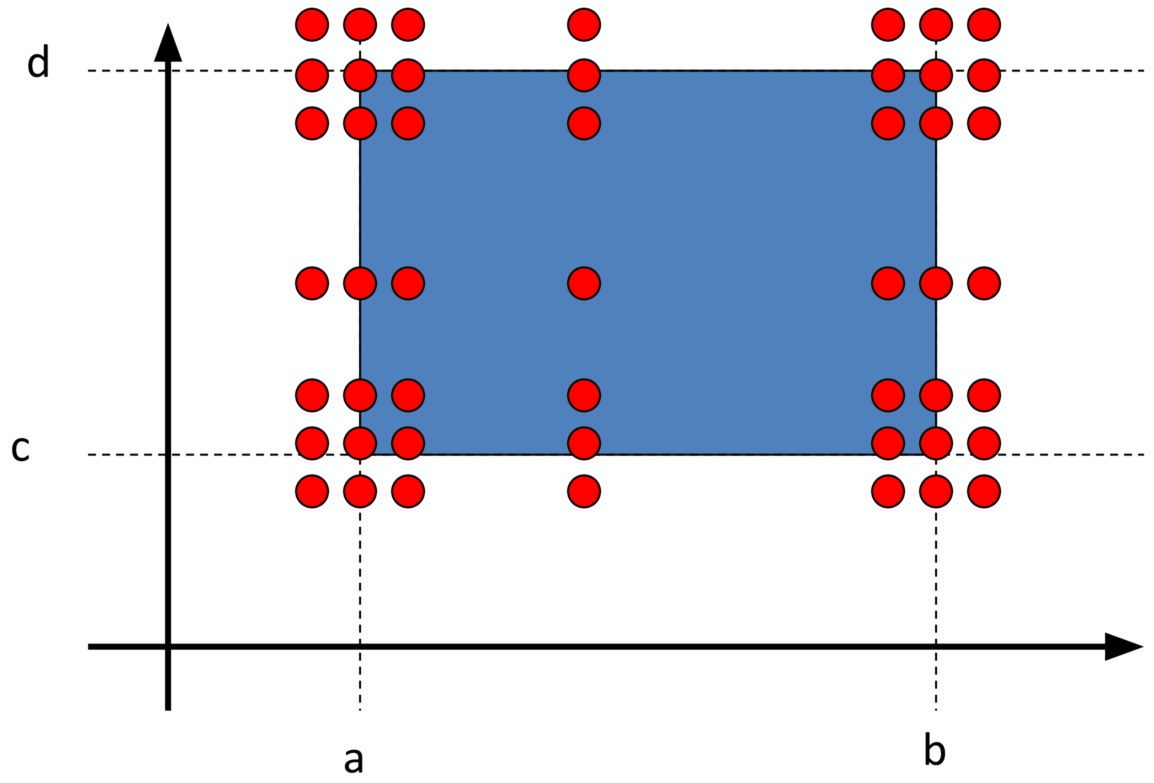
BVA Limitations

- BVA works well when the program to be tested is a function of several independent variables that represent bounded physical quantities
- No consideration to the functionality or semantic meaning of variables

BVA – Worst Case Analysis



BVA – Robust Worst Case

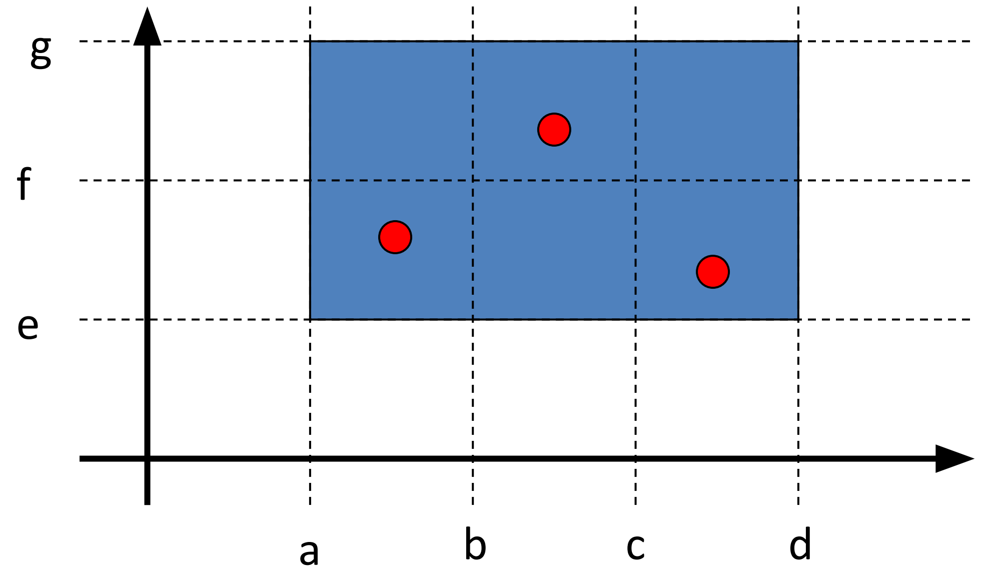


BVA – Special Value Testing

- Practiced form of functional testing
- Most intuitive and least uniform
- Use of Test Engineer’s domain knowledge
  - *Gut feel*
  - Ad hoc testing

Edge Testing

- ISTQB Advanced Level Syllabus (ISTQB, 2012) describes a hybrid of BVA and EC
- Edge Testing
- Faults near the boundaries of the classes
- Normal & Robust for Edge Testing



Software Testing Methodologies

BITS Pilani

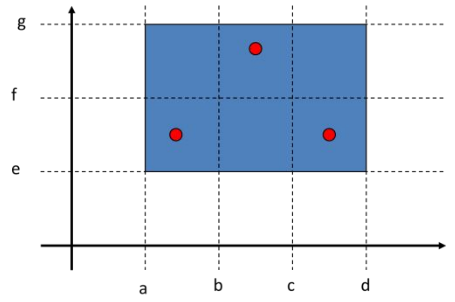
Prashant Joshi

innovate achieve lead

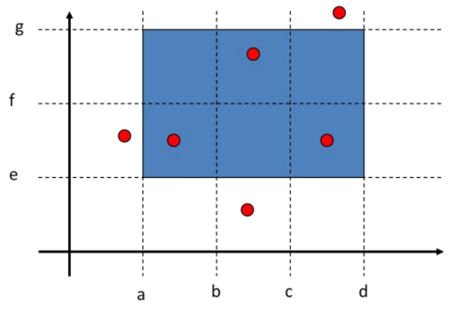
BITS Pilani
Pilani Campus

Topic 3.4: Examples & Case Study

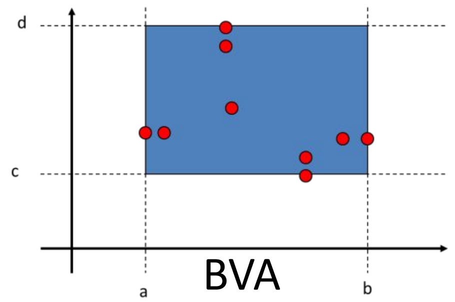
EC & BVA



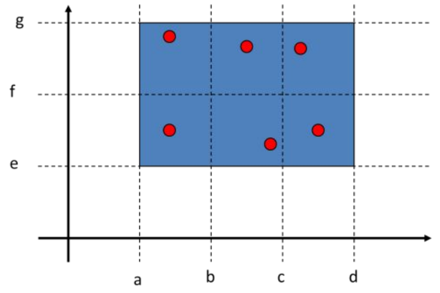
Weak Normal



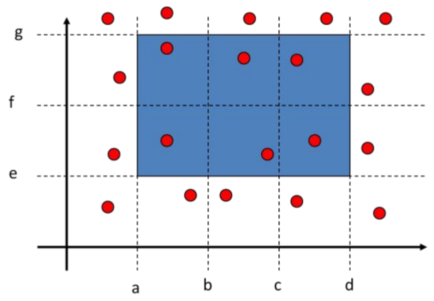
Weak Robust



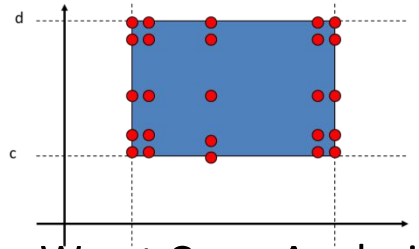
BVA



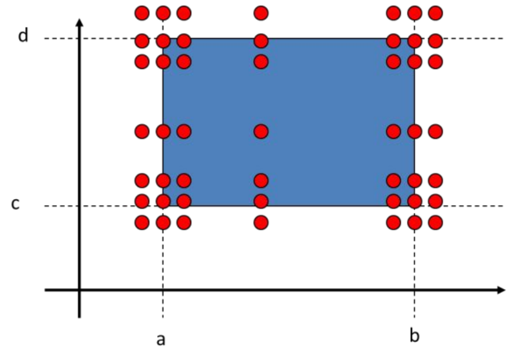
Strong Normal



Strong Robust



Worst Case Analysis



Examples

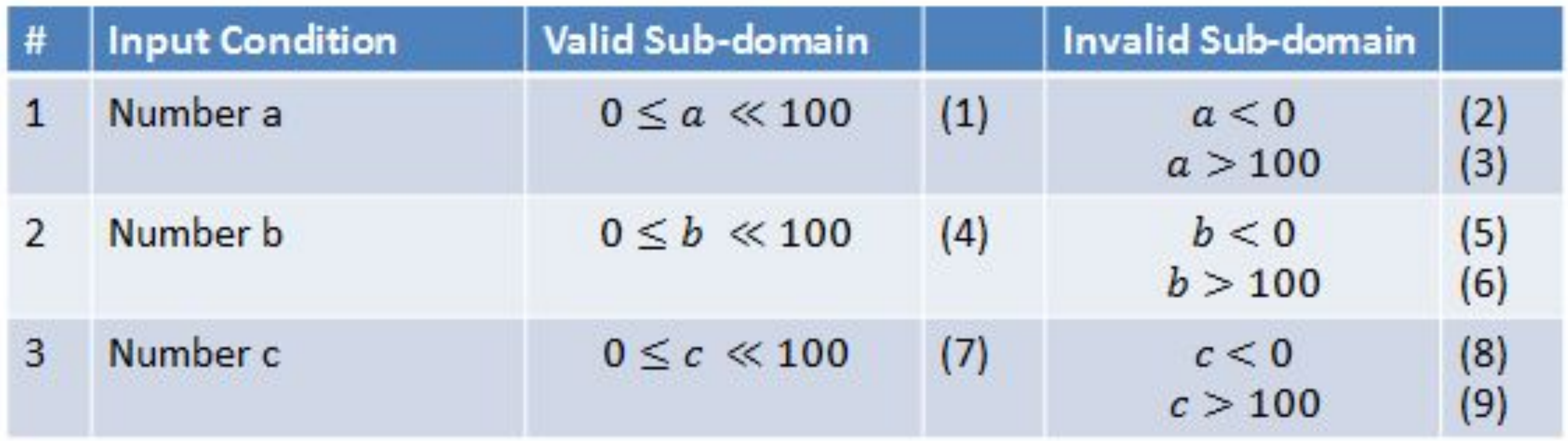
Problem 1

- Design Test Cases for a Software Program that takes in an input of up to 1000 numbers, finds the maximum and output is the max number

Problem 2

- Design and Discuss test cases for a function returns the max of 3 numbers. The numbers must be integer, else it returns an error

Example – Max of 3 numbers



#	Input Condition	Valid Sub-domain		Invalid Sub-domain	
1	Number a	$0 \le a \ll 100$	(1)	$a < 0$ $a > 100$	(2) (3)
2	Number b	$0 \le b \ll 100$	(4)	$b < 0$ $b > 100$	(5) (6)
3	Number c	$0 \le c \ll 100$	(7)	$c < 0$ $c > 100$	(8) (9)

- Choose the subdomains to satisfy for a specific type of EC or BVA
- Choose an input to form the test case

Process for Test Case Creation

- Create a table with valid and in-valid subdomains
- Number the rows
- Based on the focus (WN, SN, WR, SR of EC) pick the combination of the rows (valid and in-valid subdomains)
- Choose a value and outcome which will form a test case

Repeat any or all steps to arrive at coverage and completeness as required for the problem at hand

Example – Max of 3 numbers

#	Input Condition	Valid Sub-domain		Invalid Sub-domain	
1	Number a	$0 \le a \ll 100$	(1)	$a < 0$ $a > 100$	(2) (3)
2	Number b	$0 \le b \ll 100$	(4)	$b < 0$ $b > 100$	(5) (6)
3	Number c	$0 \le c \ll 100$	(7)	$c < 0$ $c > 100$	(8) (9)
4	Max	a b c	(10) (11) (12)		
5	Two equal	a & b b & c c & a	(13) (14) (15)		
6	All three equal	a, b &c	(16)		

Examples - discuss

- Discuss various test cases and design approaches
- What types of faults are anticipated?
- Are the requirements sufficient?
- Any assumptions made? How were the assumptions made?
- It is recommended that code should be written for both to understand the problem better.

The Triangle Example

Problem Statement

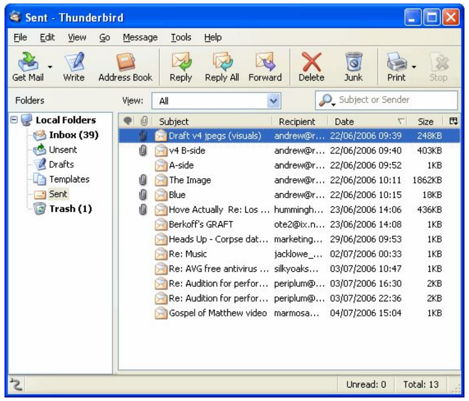
- A program takes an input of a, b and c, which are three sides of a triangle. Based on the length of the three sides the following output is generated,

1. Not a Triangle
2. Equilateral triangle
3. Isosceles Triangle
4. Scalene Triangle

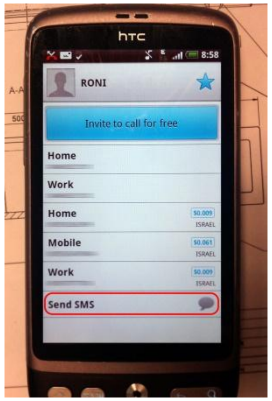
Variants (a) Type of triangle (b) Which side is the hypotenuse? (c) Area of the triangle

Examples

- Automated Teller Machine
- Tea/Coffee Vending Machine
- Washing Machine
- Contacts – Mobile Phone Application
- Messaging – Mobile Phone Application
- Email – Webmail/App/Client
- …



Subject	Recipient	Date	Size
Draft v4 jpegs (visuals)	andrew@r…	22/06/2006 09:39	248KB
v4 B-side	andrew@r…	22/06/2006 09:40	403KB
A-side	andrew@r…	22/06/2006 09:52	1KB
The Image	andrew@r…	22/06/2006 10:11	1862KB
Blue	andrew@r…	22/06/2006 10:15	18KB
Hove Actually Re: Los …	hummingh…	23/06/2006 14:06	436KB
Berkoff's GRAFT	ote2@ix.n…	23/06/2006 14:08	1KB
Heads Up - Corpse dat…	marketing…	29/06/2006 09:53	1KB
Re: Music	jacklowe_…	02/07/2006 00:33	1KB
Re: AVG free antivirus …	silkyoaks…	03/07/2006 10:47	1KB
Re: Audition for perfor…	periplum@…	03/07/2006 16:30	2KB
Re: Audition for perfor…	periplum@…	03/07/2006 22:36	2KB
Gospel of Matthew video	marmosa…	04/07/2006 15:04	1KB



Software Testing Methodologies

BITS Pilani

Prashant Joshi