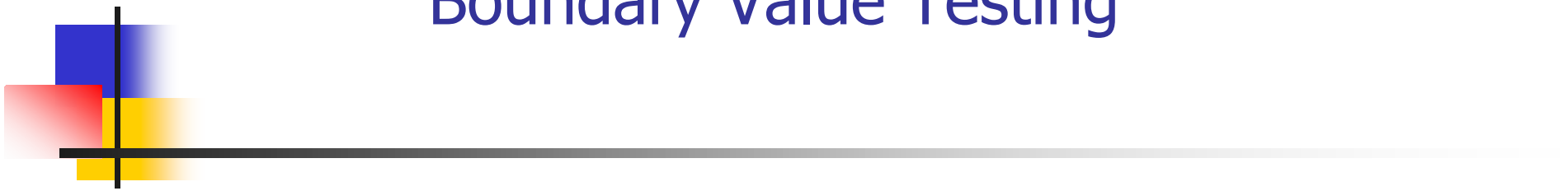
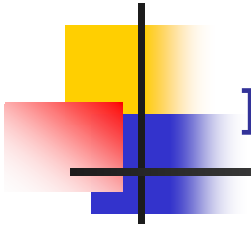


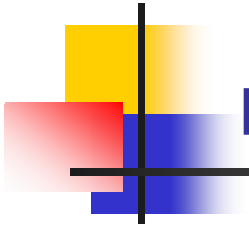
Boundary Value Testing

Chapter 5



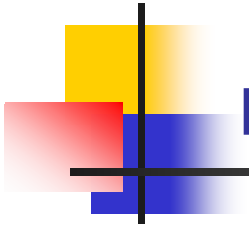
Introduction

- Input domain testing is the most commonly taught (and perhaps the most commonly used) software testing technique
- There are a number of approaches to boundary value analysis
- We will study some of the limitations of domain testing



Program view for boundary analysis

- **What is the view we take of a program for boundary analysis?**



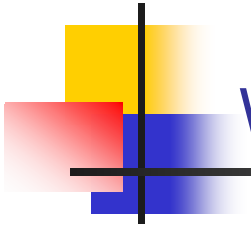
Program view for boundary analysis – 2

- Many programs can be viewed as a function *F* that maps values from a set *A* (its domain) to values in another set *B* (its range)

- The input variables of F will have some (possibly unstated) boundaries:

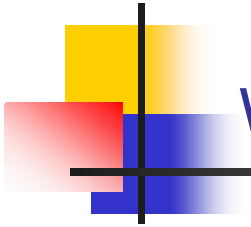
$$F(x_1, x_2) : A \rightarrow B$$

$$a \leq x_1 \leq b \qquad c \leq x_2 \leq d$$



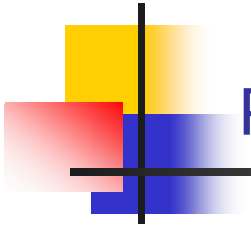
What is BVA?

- **What is boundary analysis?**



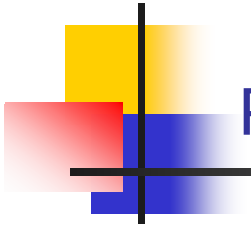
What is BVA – 2

- For each variable, select five values
  - Min The minimum
  - Min+ Slightly above the minimum
  - Nom Nominal
  - Max– Slightly below the maximum
  - Max Maximum



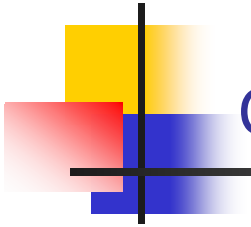
Rational for BVA

- **What is the rationale for BVA?**



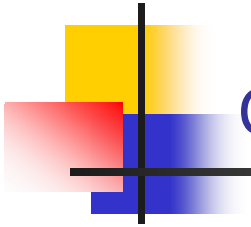
Rational for BVA – 2

- **What is the rationale for BVA?**
  - Errors occur at boundaries because people
    - Mistake logical relations such as mixing **<** with **≤**
    - Are off by one in counting
      - Fence posts and rails



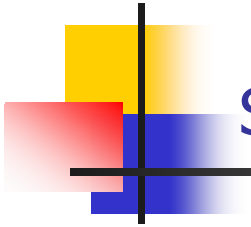
Critical assumption

- **What is the critical assumption made with boundary value testing?**



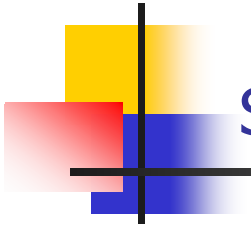
Critical assumption

- **What is the critical assumption made with boundary value testing?**
- **Single Fault Assumption**
  - Failures are rarely the result of the simultaneous occurrence of two (or more) faults



Single fault assumption

- Based on this assumption
  - **How are test cases selected when multiple variables are involved?**



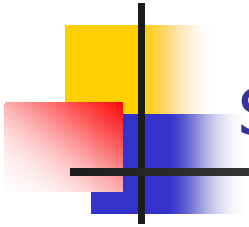
Single fault assumption – 2

- Generate test cases as such for all i

  See Figure 5.2

  - Values of all but one variable x_i at nominal
  - x_i assumes the four non-nominal values from the slide *Boundary Analysis – 2*
  - One test case with all nominal values

- **What are the number of test cases?**



Single fault assumption – 3

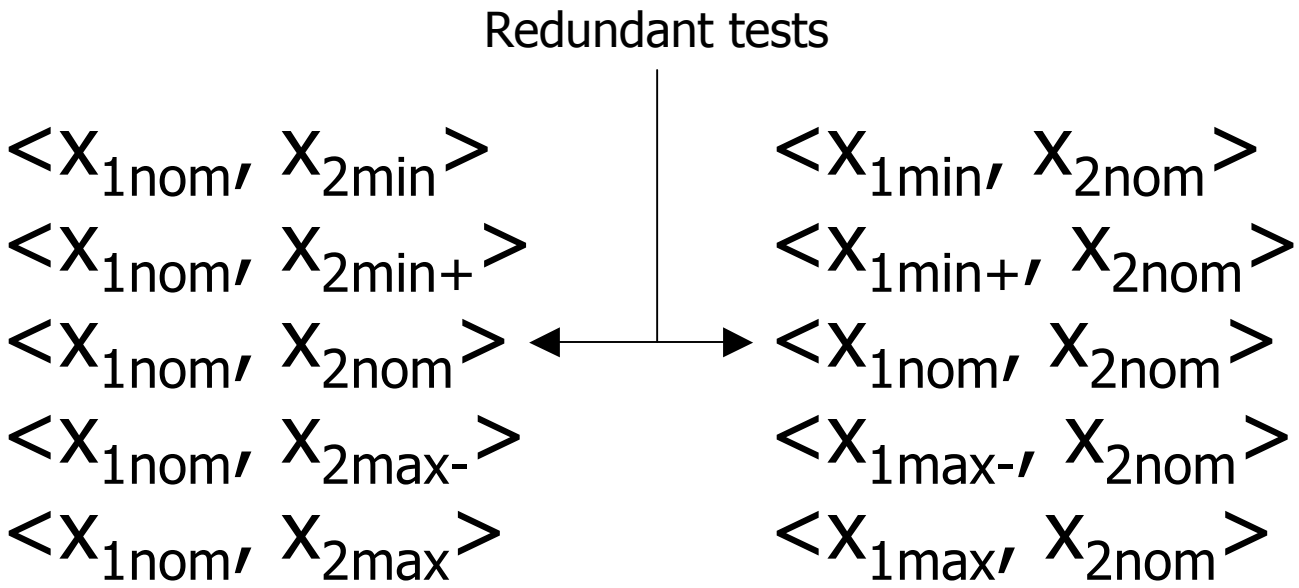
- Generate test cases as such for all i

  See Figure 5.2

  - Values of all but one variable x_i at nominal
  - x_i assumes the four non-nominal values from the slide *Boundary Analysis – 2*
  - One test case with all nominal values

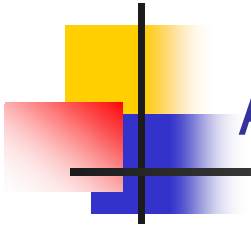
- **What are the number of test cases?**
  - **#Variables * 4 + 1**

Two-variable function test cases



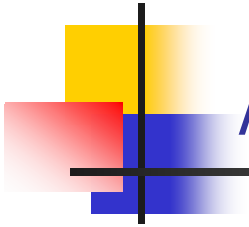
Apply BVA to the Triangle problem

$1 \le a \le 200$
$1 \le b \le 200$
$1 \le c \le 200$



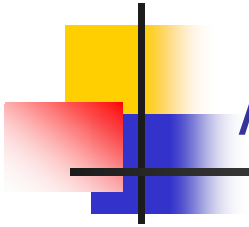
Advantages

- **When does boundary value analysis work well?**



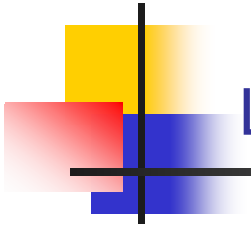
Advantages – 2

- Independent variables
  - Single fault assumption
- Physical quantities
- Languages that are not strongly typed
  - **Why were strongly typed languages developed?**



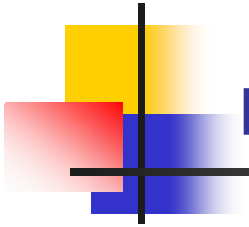
Advantages – 3

- Independent variables
  - Single fault assumption
- Physical quantities
- Languages that are not strongly typed
  - **Why were strongly typed languages developed?**
    - To prevent errors easily found with BVA



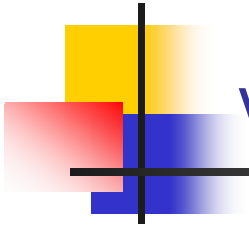
Limitations

- **What are the limitations of boundary value analysis?**



Limitations – 2

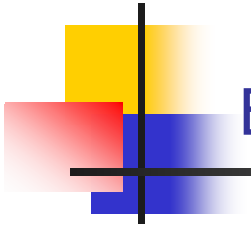
- Does not work well for Boolean variables
  - **Why are these not suitable?**
- Does not work well for logical variables
  - PIN, transaction type
  - **Why are these not suitable?**
- When variables are not independent – i.e. are dependent
  - **What example does the textbook give?**
- Not that useful for strongly-typed languages



Variations of boundary value analysis

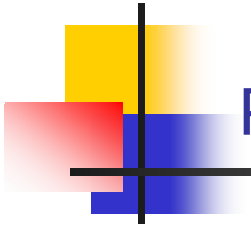
- **What extensions or variations are made for boundary value analysis?**

- **What is the justification for each?**



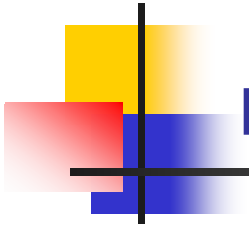
Extensions

- Robustness testing
- Worst case testing
- Robust worst case testing
- Special value testing
- Random testing



Robustness testing

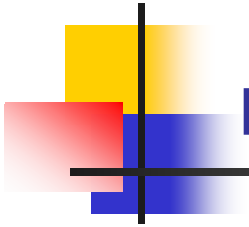
- **What is robustness testing?**



Robustness testing – 2

- Add two more values per variable
  - Max+ Slightly greater than the maximum
  - Min– Slightly less than the minimum
- What is the expected output?
  - Hopefully error message, system recovers
- Implementing these test cases may not be possible
  - **What is the difficulty?**

See Figure 5.3

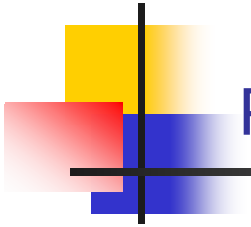


Robustness testing – 3

- Add two more values per variable
  - Max+ Slightly greater than the maximum
  - Min– Slightly less than the minimum

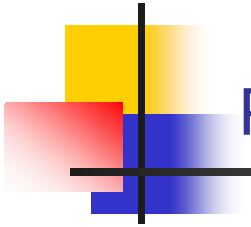
See Figure 5.3

- What is the expected output?
  - Hopefully error message, system recovers
- Implementing these test cases may not be possible
  - **What is the difficulty?**
    - Determining the expected output for out of range values



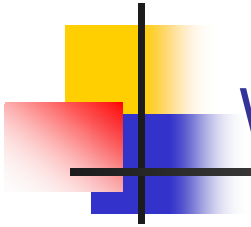
Robustness testing – 4

- **What are the number of test cases?**
- **When is robust testing mandated?**



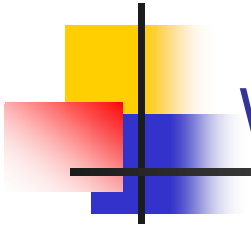
Robustness testing – 5

- **What are the number of test cases?**
  - **#variables * 6 + 1**
- **When is robust testing mandated?**
  - With exception handling



Worst case testing

- **What is worst case testing?**

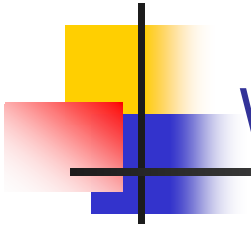


Worst-Case Testing – 2

- Rejects the simple fault assumption and tests all **combinations** of values
- Often leads to a large number of test cases with low bug-finding power
  - **Why?**

See Figure 5.4

- **What are the number of test cases?**

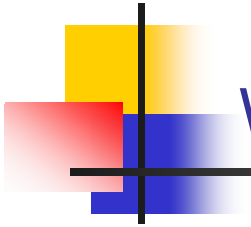


Worst-Case Testing – 3

- Rejects the simple fault assumption and tests all **combinations** of values

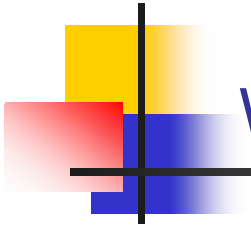
See Figure 5.4

- Often leads to a large number of test cases with low bug-finding power
  - **Why?**
    - Typically there are few bugs compared to the number of tests
- **What are the number of test cases?**
  - **5 ^ #variables**



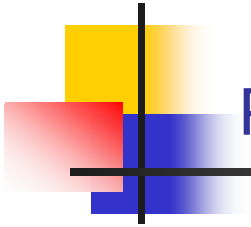
Worst-Case Testing – 4

- **What type of testing is better to do in place of worst case testing?**



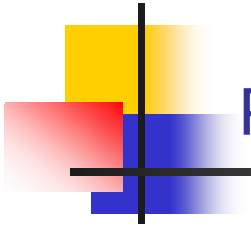
Worst-Case Testing – 5

- **What type of testing is better to do in place of worst case testing?**
  - Often better to use Special Value Testing



Robust worst case testing

- **What is robust worst case testing?**



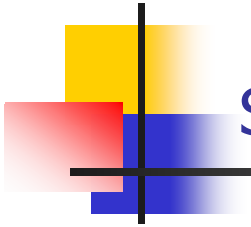
Robust worst case testing – 2

- Add the values min– and max+ to the possible variable values

See Figure 5.5

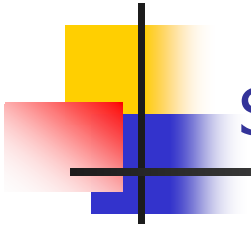
- Now take all combinations of variable values

- **What are the number of test cases?**



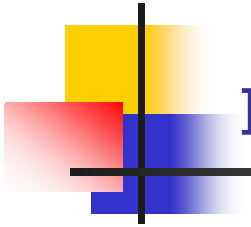
Special value testing

- **What is special value testing?**



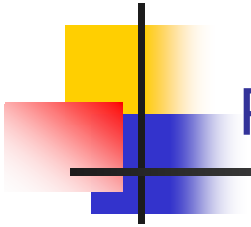
Special value testing – 2

- Use best engineering judgment
  - Intuition
  - Domain knowledge
  - Experience
  - Soft spots



In class activity

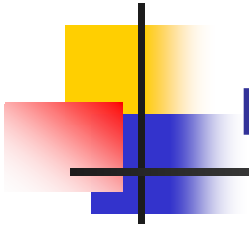
- Do exercises 1, 2 and 3



Random testing

- Select random values for each variable

- **How many tests do we make?**



Random testing – 2

- Select random values for each variable

- **How many tests do we make?**
  - Related to the probability of producing every outcome at least once
  - Related to the probability of executing every statement / path at least once