PROMPT Software Testing

Combinatorial Testing

SWEDISH ICT SICS

>PROMPT

Combinatorial Issues?

Book hotel room		
Room type	Single	Double
Breakfast	Breakfast	No Breakfast
Check-out	Regular	Late

Combinatorial Issues?

Book hotel room		
Room type	Single	Double
Breakfast	Breakfast	No Breakfast
Check-out	Regular	Late

Test case inputs (exhaustive)		
Room type	**Breakfast**	**Check-out**
Single	Breakfast	Regular
Single	Breakfast	Late
Single	No Breakfast	Regular
Single	No Breakfast	Late
Double	Breakfast	Regular
Double	Breakfast	Late
Double	No Breakfast	Regular
Double	No Breakfast	Late

Number of test cases = $2^3 = 8$

Combinatorial Issues?

Book hotel room			
Room type	Single	Double	Suite
Breakfast	Breakfast	No Breakfast	Giant Breakfast
Check-out	Regular	Late	Very Late

With three possibilities for each input…

Combinatorial Issues?

Book hotel room			
Room type	Single	Double	Suite
Breakfast	Breakfast	No Breakfast	Giant Breakfast
Check-out	Regular	Late	Very Late

Test case inputs (exhaustive)		
Room type	**Breakfast**	**Check-out**
Single	Breakfast	Regular
Single	Breakfast	Late
Single	Breakfast	Very Late
Single	No Breakfast	Regular
Single	No Breakfast	Late
Single	No Breakfast	Very Late
Single	Giant Breakfast	Regular
Single	Giant Breakfast	Late
Single	Giant Breakfast	Very Late
Double	Breakfast	Regular
Double	Breakfast	Late
Double	Breakfast	Very Late
Double	No Breakfast	Regular
Double	No Breakfast	Late
Double	No Breakfast	Very Late
Double	Giant Breakfast	Regular
Double	Giant Breakfast	Late
Double	Giant Breakfast	Very Late
Suite	Breakfast	Regular
Suite	Breakfast	Late
Suite	Breakfast	Very Late
Suite	No Breakfast	Regular
Suite	No Breakfast	Late
Suite	No Breakfast	Very Late
Suite	Giant Breakfast	Regular
Suite	Giant Breakfast	Late
Suite	Giant Breakfast	Very Late

Number of test cases = $3^3 = 27$

Sources of Combinatorial Issues in Software

- Equivalence partitioning input domain model
- Configuration space
- Product/system variants
- Platform portability requirements
- …

Pairwise Testing

Main idea: Instead of trying to cover *all* combinations exhaustively, ensure that *all pairwise* combinations of inputs are covered

Rests on the hypothesis that most interaction problems does not involve more than two variables.

Pairwise Testing

Book hotel room		
Room type	**Single**	Double
Breakfast	Breakfast	No Breakfast
Check-out	Regular	Late

Pairwise test requirement:

Each **possible value** of each **variable** is combined with **each possible value** for **each other variable** at least once.

For the **Single** value of the **Room type** variable, this requirement would look as follows…

Pairwise Testing

Book hotel room		
Room type	**Single**	Double
Breakfast	Breakfast	No Breakfast
Check-out	Regular	Late

Book hotel room		
Room type	**Single**	
Breakfast	**Breakfast**	
Check-out		

Book hotel room		
Room type	**Single**	
Breakfast		
Check-out	**Regular**	

Book hotel room		
Room type	**Single**	
Breakfast		**No Breakfast**
Check-out		

Book hotel room		
Room type	**Single**	
Breakfast		
Check-out		**Late**

Pairwise test case input generation

Book hotel room		
Room type	Single	Double
Breakfast	Breakfast	No Breakfast
Check-out	Regular	Late

pairwise

Test case inputs (pairwise)		
Room type	**Breakfast**	**Check-out**
Single	Breakfast	Regular
Single	No breakfast	Late
Double	Breakfast	Late
Double	No Breakfast	Regular

Number of test cases = 4

Other Combinatiorial Testing Strategies

- T-wise (general form of pairwise)
- Each choice
- Base choice