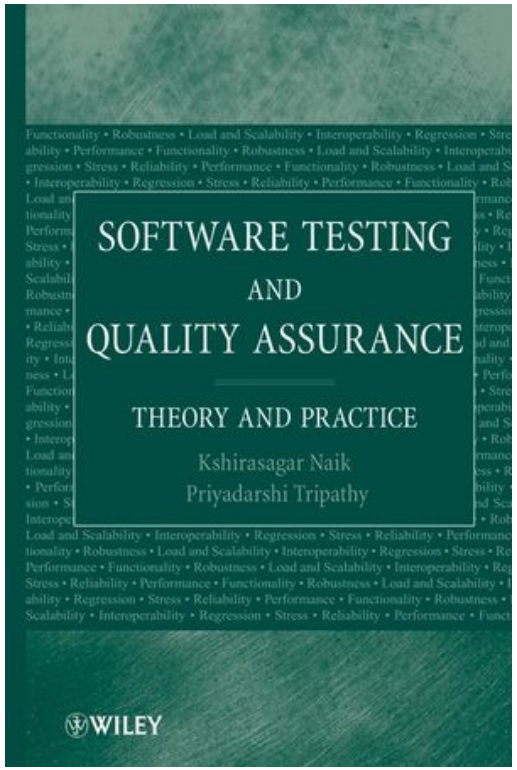
Software Testing

ETS 200

http://cs.lth.se/ets200



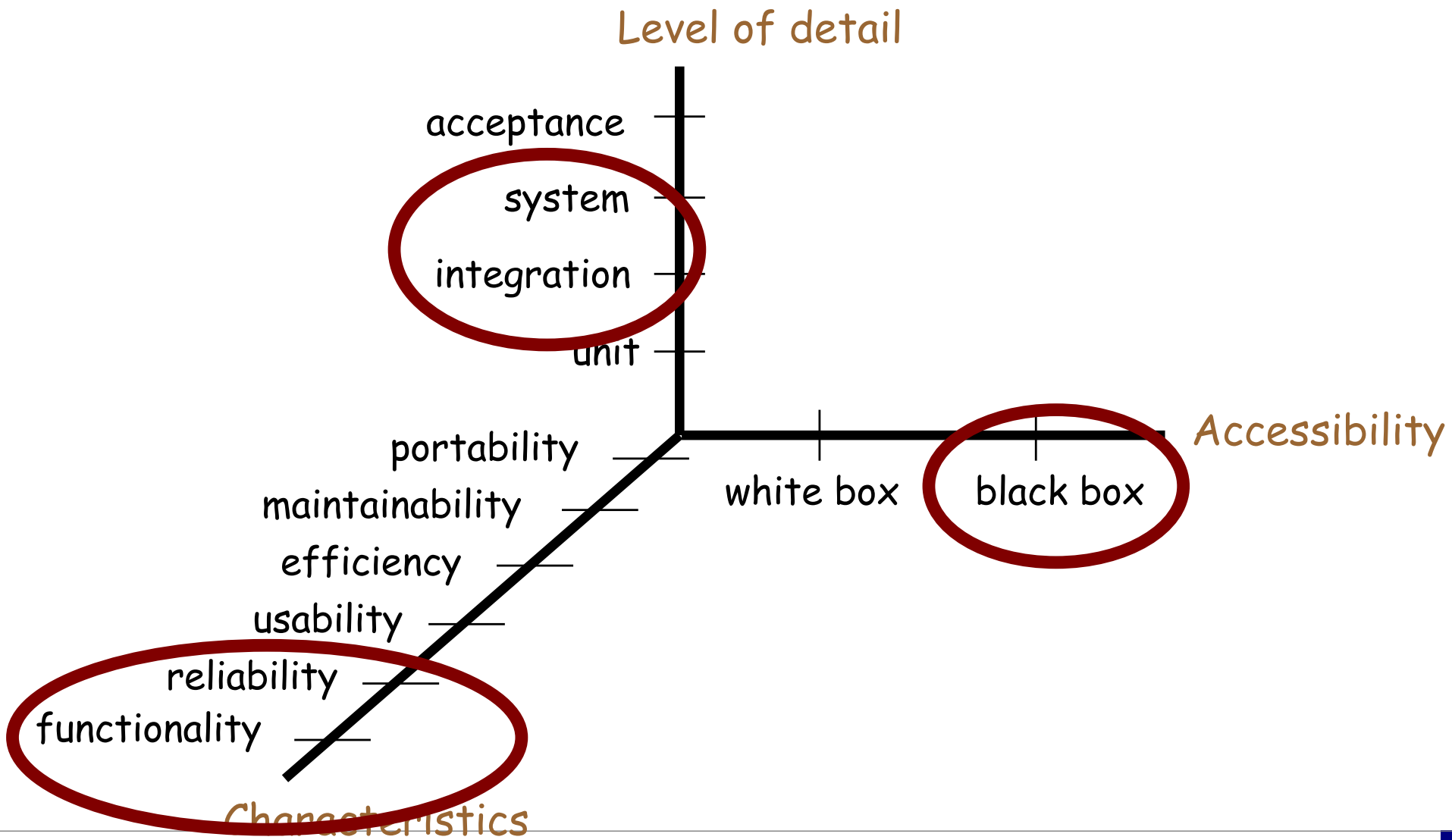
Chapter 6, 9.2-9.6

Prof. Per Runeson

Lecture

- Chapter 6: **Domain Testing**
- Chapter 9.2-9.6: **Functional Testing**
  - Black-box testing techniques (Lab 2)

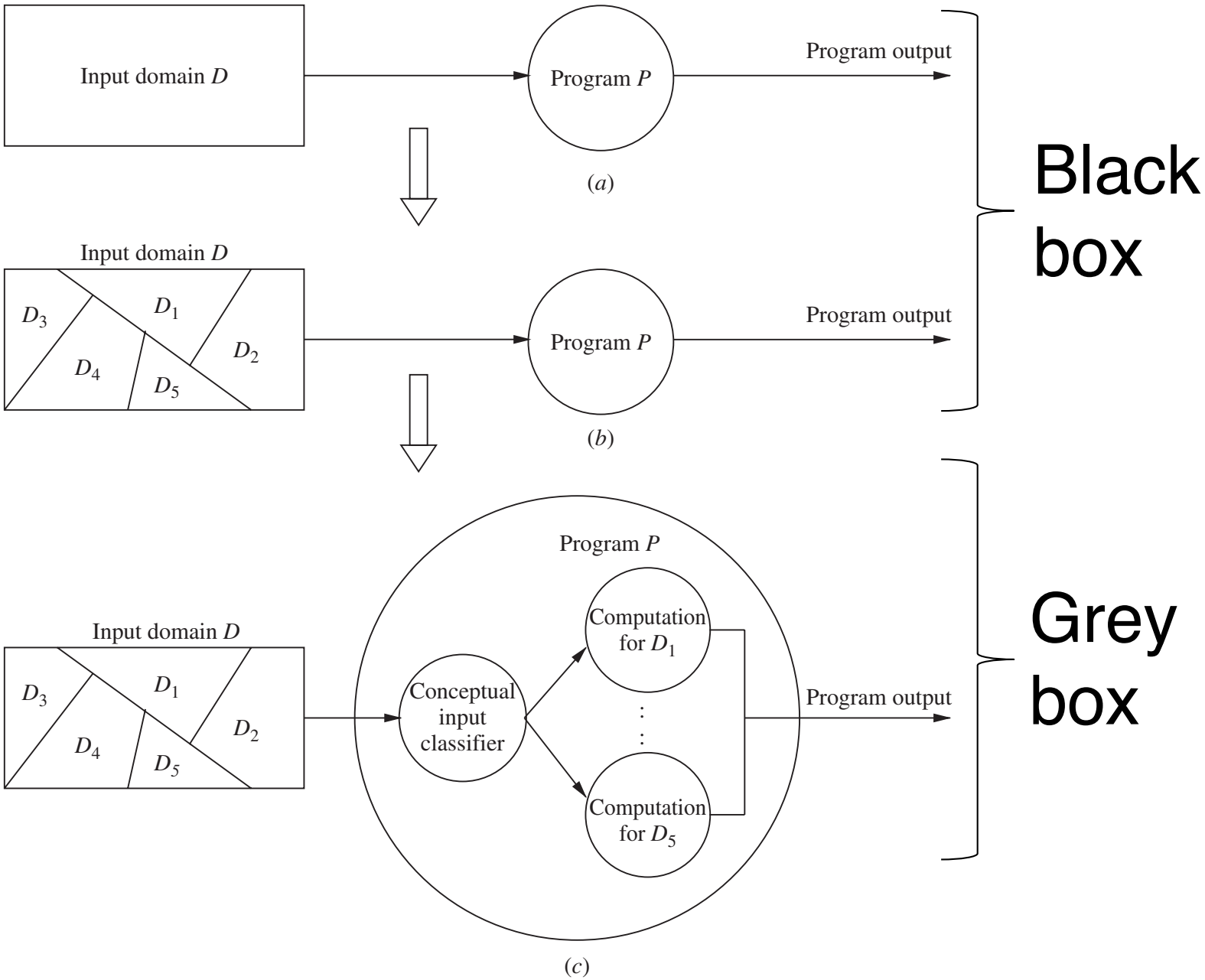
Types of Testing



Definitions

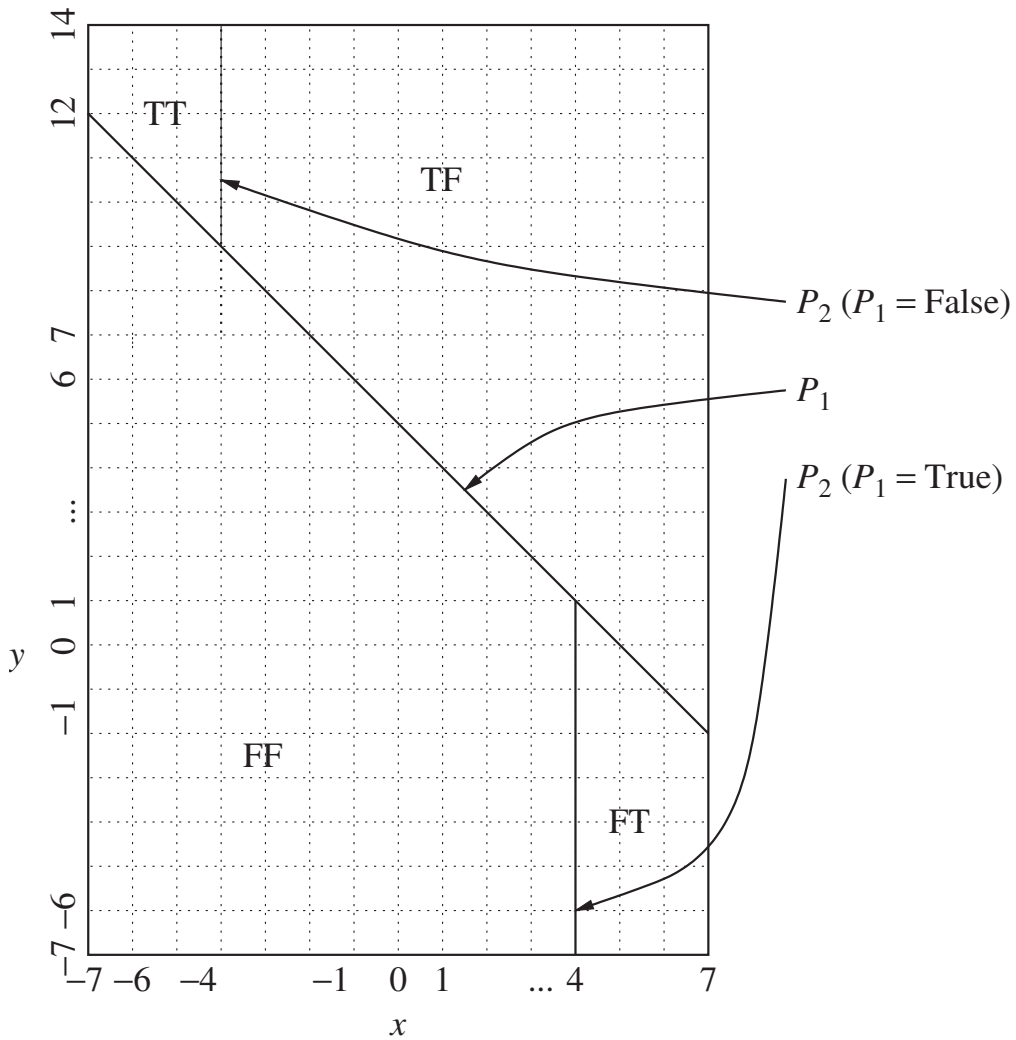
- *Computation Error* occurs when a specific input data causes the program to execute the correct, i.e., desired path, but the output value is wrong.
- *Domain Error* occurs when a specific input data causes the program to execute a *wrong*, that is, undesired, path in the program.
- A *domain* is a set of input values for which the program performs the same computation for every member of the set.

Domain testing

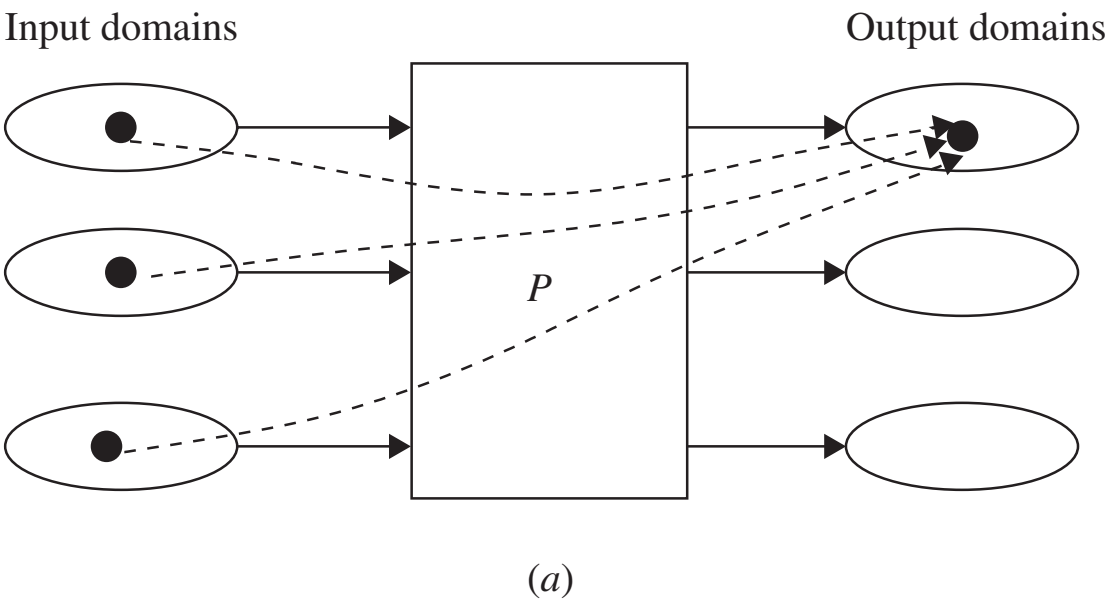


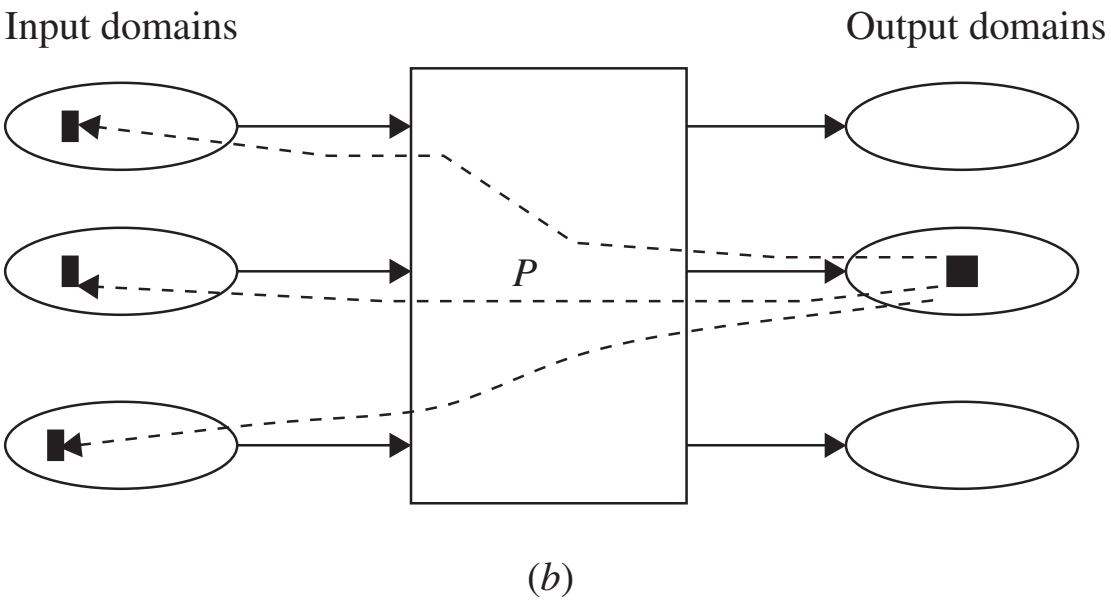
White/Grey box domain testing

- Draw a control flow graph from the given source code
- Find all possible interpretations of the predicates
- Analyze the interpreted predicates to identify domains

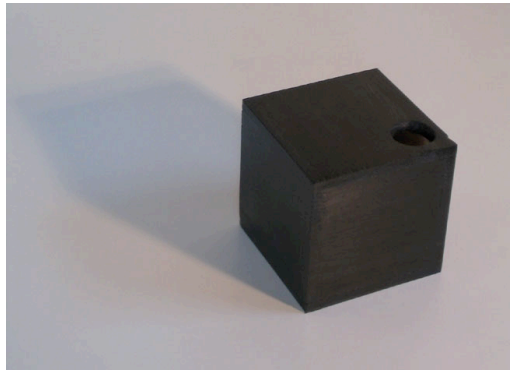


Black box domain testing





Black-Box Testing Methods



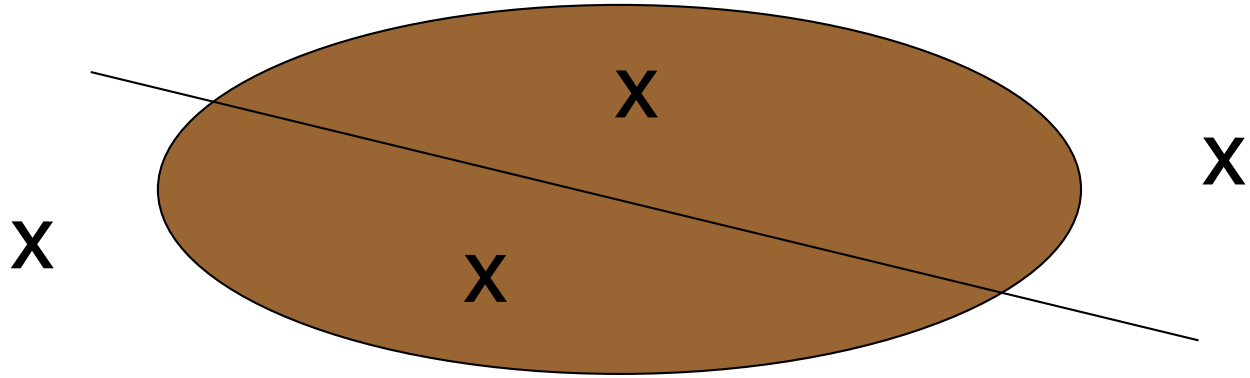
- Equivalence class partitioning (ECP)
- Boundary value analysis (BVA)
- Combinatorial testing
- State transition testing
- Error guessing

Example
– Insurance System

- Specification Statement
  - System shall reject over-age insurance applicants.
- Specification Item
  - Reject male insurance applicants over the age of 80 years on day of application.
  - Reject female insurance applicants over the age of 85 years on day of application.

Equivalence Class Partioning

- Split input/output into classes which the software handles equivalently.
- Select test cases to represent each class.



Example (cont.)

Input: Gender & Age

Output: accept/reject

Classes

C1: Input: Males over 80

C2: Input: Males 80 or under

C3: Input: Females over 85

C4: Input: Females 85 or under

C5: Output: “accept”

C6: Output: “reject”

Test Cases – Valid:

TC1: male, 83, reject

TC2: male, 56, accept

TC3: female, 101, reject

TC4: female, 83, accept

Invalid:

TC5: female, -3, ??

TC6: fmale, 14, ??

Equivalence Partitioning Guidelines

If input condition [Myers79/04]:

- is a range, e.g., x = [0, 9]
- is an ordered list of values, e.g., owner = <1, 2, 3, 4>

→ one valid and two invalid classes are defined

- is a set, e.g., vehicle = [car, motorcycle, truck]
- is a “must be” condition (boolean)

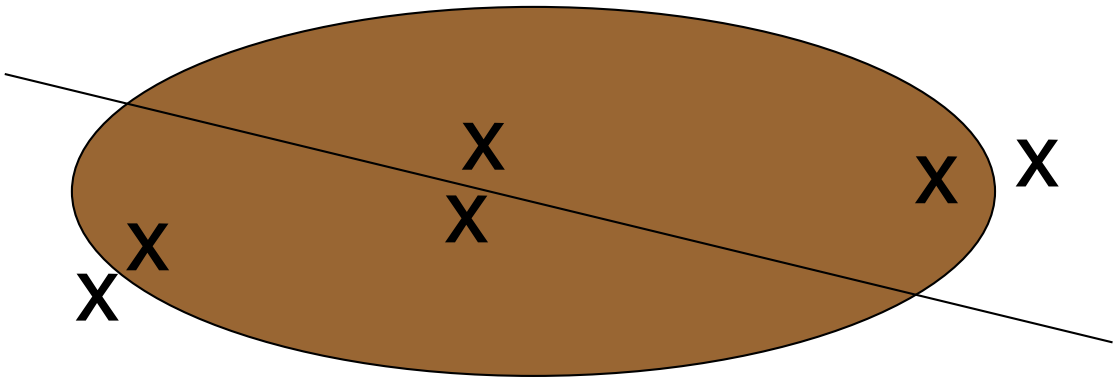
→ one valid and one invalid class are defined

- is anything else

→ partition further

Boundary Value Analysis

- Adds to the equivalence partitioning method.
- Select test cases to represent each side of the class boundaries.



Example cont.

Input: Gender & Age

Output: accept/reject

Classes

C1: Males over 80

C2: Males 80 or under

C3: Females over 85

C4: Females 85 or under

C5: Output “accept”.

C6: Output “reject”.

Test Cases – Valid

TC1: male, 81, reject

TC2: male, 80, accept

TC3: female, 86, reject

TC4: female, 85, accept

Invalid:

TC5: female, -1, ??

TC6: female, 0, accept

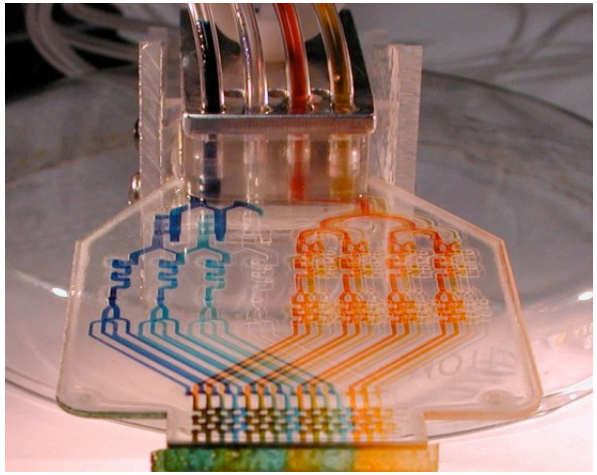
TC7: fmale, -1, ??

TC8: fmale, 0, ??

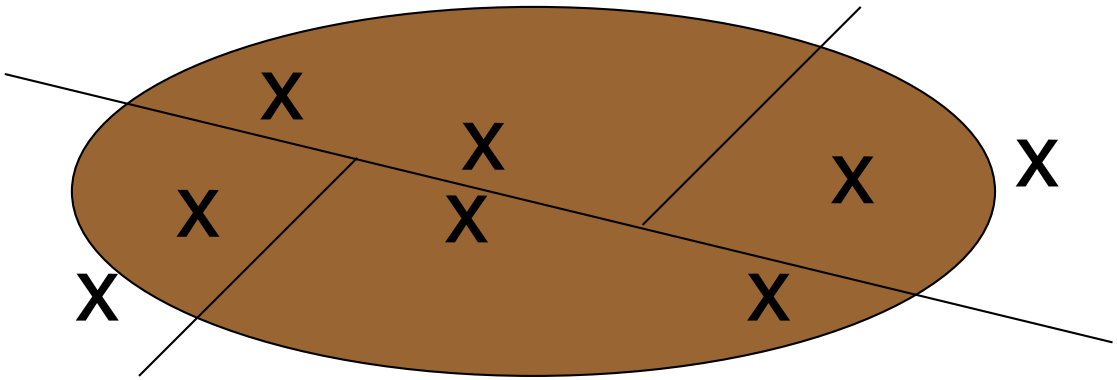
Boundary Value Analysis Guidelines

- Range a..b ⇒ a, b, just above a, just below b
- List of values: max, min, just below min, just above max
- Boundaries of externally visible data structures shall be checked (e.g. ordered sets, arrays)
- Output bounds should be checked

Combinatorial Designs



- ECP and BVA define test cases per class
- Combinations of equivalence classes need to be handled
- Combinatorial explosion needs to be handled



Combinatorial Testing

Chap 9.3 or [Kuhn et al 2008]

http://csrc.nist.gov/groups/SNS/acts/

- What is it?
  - Methods for systematically testing t-way interaction effects of input (or configuration parameter) values.
- Why do it?
  - The interaction of specific combinations of input values may trigger failures that won't be triggered if testing input (or configuration parameter) values only in isolation.

Two Scopes of Combinatorial Testing

Test Configurations

Test case	OS	CPU	Protocol
1	Windows	Intel	IPv4
2	Windows	AMD	IPv6
3	Linux	Intel	IPv6
4	Linux	AMD	IPv4

Test Inputs

Size	Topp	Addr	Phone
Sm	Custom	Val	Val
Sm	Preset	Inv	Inv
Med	Custom	Inv	Val
Med	Preset	Val	Inv
Lg	Custom	Val	Inv
Lg	Preset	Inv	Val

Pizza Delivery

System under test

Combinatorial Testing – Example 1

- Platform configuration parameters:
  - OS: Windows XP, Apple OS X, Red Hat Linux
  - Browser: Internet Explorer, Firefox
  - Protocol: IPv4, IPv6
  - CPU: Intel, AMD
  - DBMS: MySQL, Sybase, Oracle
- Total number of combinations: $3*2*2*2*3 = 72$
- Do we need 72 test cases?

- https://courses.cs.ut.ee/MTAT.03.159/2013_spring/uploads/Main/SWT_comb-paper2.pdf

Pair-Wise Testing (2-Way Interaction)

- Only 10 test needed, if we want to test all interactions of one parameter with one other parameter (pair-wise interaction)
- # of pairs:

$$\binom{5}{2} = \frac{5!}{2!(5-2)!} = 10$$

- # of 2-way interactions N=57:

$$40 = \binom{5}{2}2^2 \le N \le \binom{5}{2}3^2 = 90$$

Test	OS	Browser	Protocol	CPU	DBMS
1	XP	IE	IPv4	Intel	MySQL
2	XP	Firefox	IPv6	AMD	Sybase
3	XP	IE	IPv6	Intel	Oracle
4	OS X	Firefox	IPv4	AMD	MySQL
5	OS X	IE	IPv4	Intel	Sybase
6	OS X	Firefox	IPv4	Intel	Oracle
7	RHL	IE	IPv6	AMD	MySQL
8	RHL	Firefox	IPv4	Intel	Sybase
9	RHL	Firefox	IPv4	AMD	Oracle
10	OS X	Firefox	IPv6	AMD	Oracle

Pair-Wise Testing (2-Way Interaction)

- Only 10 tests needed, if we want to test all interactions of one parameter with one other parameter (pair-wise interaction)

- Pair 1:
  3 x 2 = 6 Combinations

Test	OS	Browser	Protocol	CPU	DBMS
1	XP	IE	IPv4	Intel	MySQL
2	XP	Firefox	IPv6	AMD	Sybase
3	XP	IE	IPv6	Intel	Oracle
4	OS X	Firefox	IPv4	AMD	MySQL
5	OS X	IE	IPv4	Intel	Sybase
6	OS X	Firefox	IPv4	Intel	Oracle
7	RHL	IE	IPv6	AMD	MySQL
8	RHL	Firefox	IPv4	Intel	Sybase
9	RHL	Firefox	IPv4	AMD	Oracle
10	OS X	Firefox	IPv6	AMD	Oracle

Pair-Wise Testing (2-Way Interaction)

- Only 10 tests needed, if we want to test all interactions of one parameter with one other parameter (pair-wise interaction)

- Pair 2:
  2 x 2 = 4 Combinations

Test	OS	Browser	Protocol	CPU	DBMS
1	XP	IE	IPv4	Intel	MySQL
2	XP	Firefox	IPv6	AMD	Sybase
3	XP	IE	IPv6	Intel	Oracle
4	OS X	Firefox	IPv4	AMD	MySQL
5	OS X	IE	IPv4	Intel	Sybase
6	OS X	Firefox	IPv4	Intel	Oracle
7	RHL	IE	IPv6	AMD	MySQL
8	RHL	Firefox	IPv4	Intel	Sybase
9	RHL	Firefox	IPv4	AMD	Oracle
10	OS X	Firefox	IPv6	AMD	Oracle

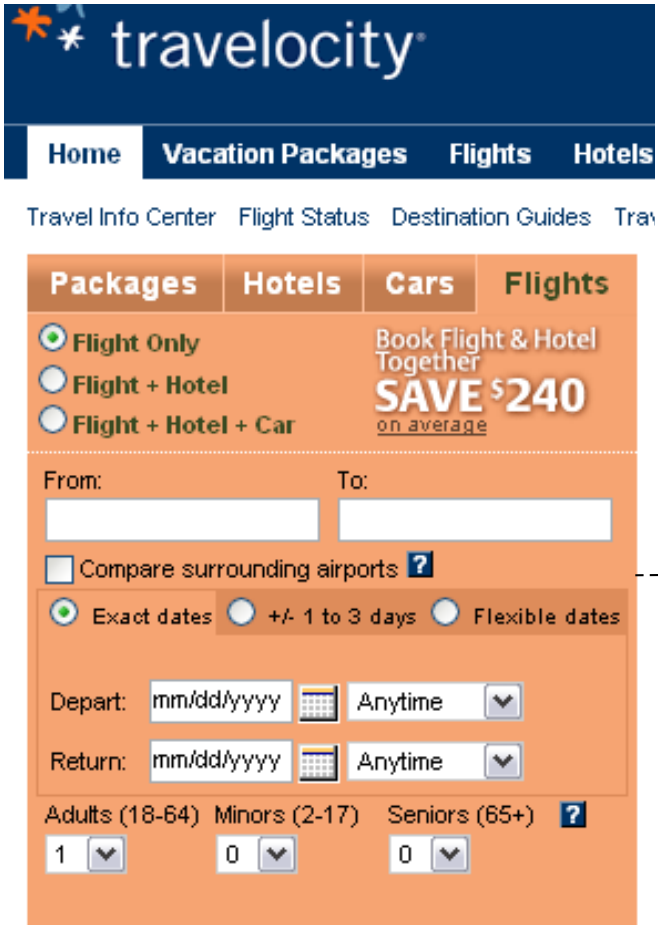
Pair-Wise Testing (2-Way Interaction)

- Only 10 tests needed, if we want to test all interactions of one parameter with one other parameter (pair-wise interaction)

- Pair 3:
  3 x 3 = 9 Combinations

Test	OS	Browser	Protocol	CPU	DBMS
1	XP	IE	IPv4	Intel	MySQL
2	XP	Firefox	IPv6	AMD	Sybase
3	XP	IE	IPv6	Intel	Oracle
4	OS X	Firefox	IPv4	AMD	MySQL
5	OS X	IE	IPv4	Intel	Sybase
6	OS X	Firefox	IPv4	Intel	Oracle
7	RHL	IE	IPv6	AMD	MySQL
8	RHL	Firefox	IPv4	Intel	Sybase
9	RHL	Firefox	IPv4	AMD	Oracle
10	OS X	Firefox	IPv6	AMD	Oracle

Combinatorial Testing – Example 2



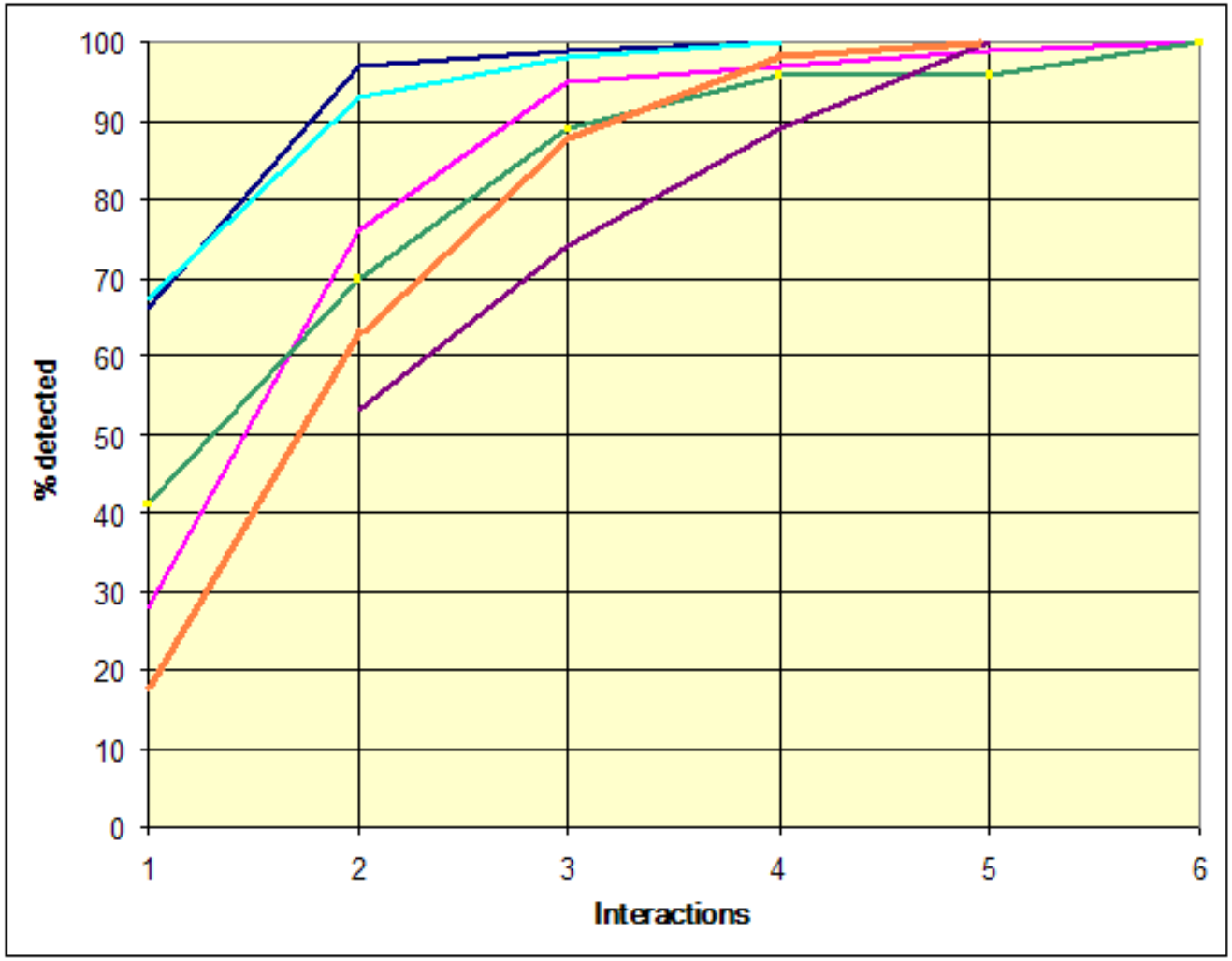
Many variables
Many values per variable
Need to abstract values (equivalence classes, boundary values)

Plan: flt, flt+hotel, flt+hotel+car
From: CONUS, HI, AK, Europe, Asia …
To: CONUS, HI, AK, Europe, Asia …
Compare: yes, no
Date-type: exact, 1to3, flex
Depart: today, tomorrow, 1yr, Sun, Mon …
Return: today, tomorrow, 1yr, Sun, Mon …
Adults: 1, 2, 3, 4, 5, 6
Minors: 0, 1, 2, 3, 4, 5
Seniors: 0, 1, 2, 3, 4, 5

Is Testing 2-Way Interactions Enough?

- Analyses of failure-triggering conditions showed this:

  Medical device (dark blue)
  NASA distrib. DB (light blue)
  Browser (green)
  Web server (magenta)
  Network security (orange)
  TCAS* module (purple)



* Traffic Collision Avoiding System

National Institute of Standards and Technology

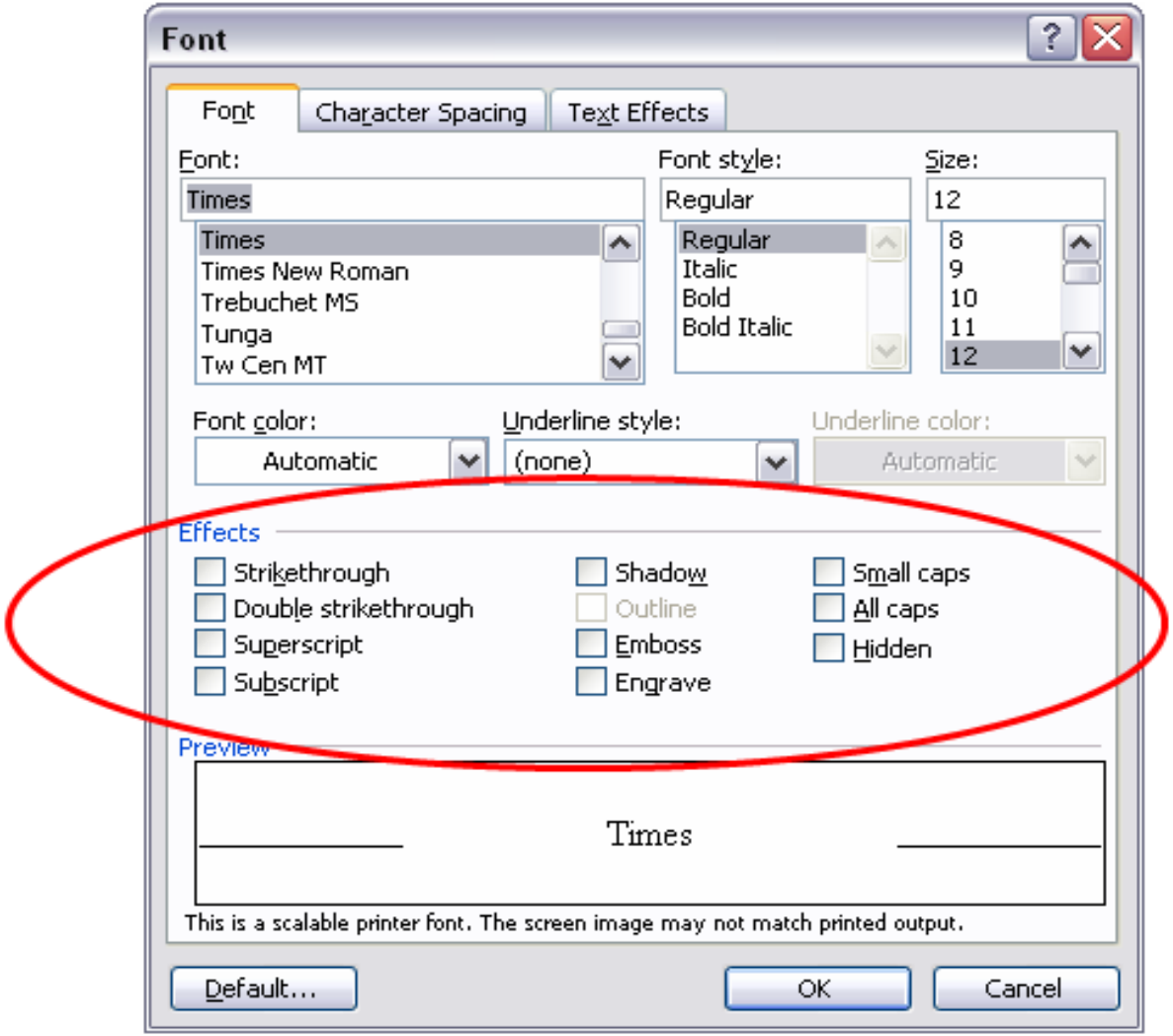
Is Testing 2-Way Interactions Enough?

- Several studies have shown that Pair-wise Testing finds between 50% and 90% of all failures.
- More strengths than 6-way interaction has hardly ever shown to find more defects.
  - Why is that good?

Combinatorial Testing – Example 3

- **How Many Tests Would It Take?**
  - There are 10 effects, each can be on or off
  - All combinations is 2^{10} = 1,024 tests

    too many to visually check …
  - Let's look at all 3-way interactions …



How Many Tests Would it Take?

- There are $\binom{10}{3} = 120$ 3-way interactions.
- Naively 120 x 2^3 = 960 tests.
- Since we can pack 3 triples into each test, we need no more than 320 tests.
- Each test exercises many triples:

0 0 0 1 1 1 0 1 0 1

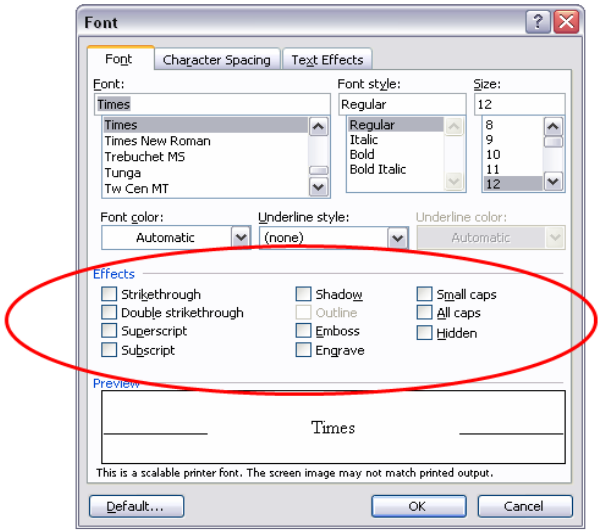
We should be able to pack many in one test, so what's the smallest number we need?

Covering Arrays

Each row is a test:

0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	0	0	0	0	1
1	0	1	1	0	1	0	1	0	0
1	0	0	0	1	1	1	0	0	0
0	1	1	0	0	1	0	0	1	0
0	0	1	0	1	0	1	1	1	0
1	1	0	1	0	0	1	0	1	0
0	0	0	1	1	1	0	0	1	1
0	0	1	1	0	0	1	0	0	1
0	1	0	1	1	0	0	1	0	0
1	0	0	0	0	0	0	1	1	1
0	1	0	0	0	1	1	1	0	1

Each column is a parameter:



All 3-way interactions in only 13 tests

http://dx.doi.org/10.6028/NIST.SP.800-142

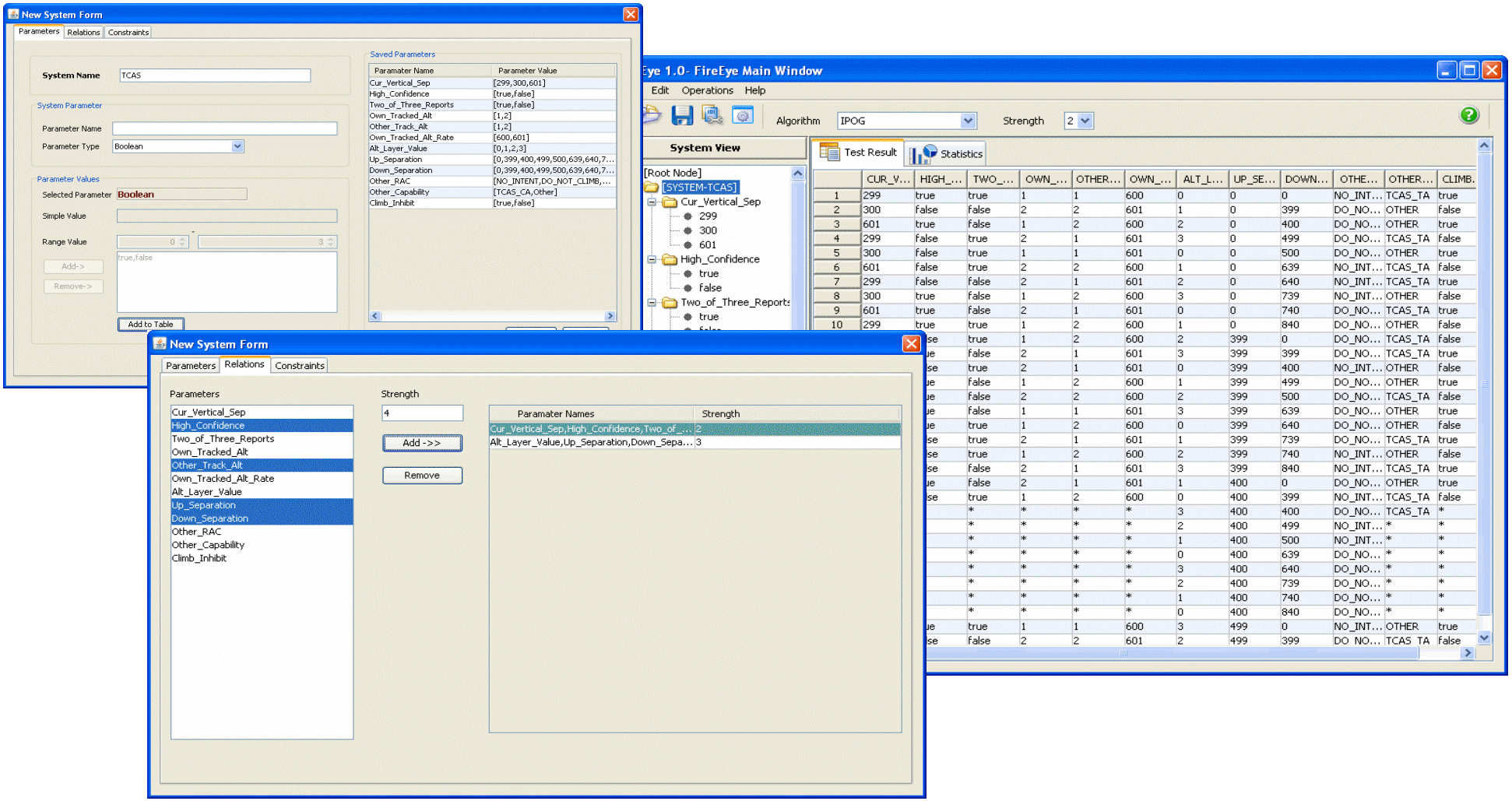
How to Generate Covering Arrays?

- Combinatorial testing requires many tests
- Number of tests: Suppose we want all 4-way combinations of 30 parameters, 5 values each → 3,800 tests
- May need 10^3 to 10^7 tests for realistic systems
- Generating covering arrays is a hard problem, one reason why anything beyond pair-wise testing is rarely done

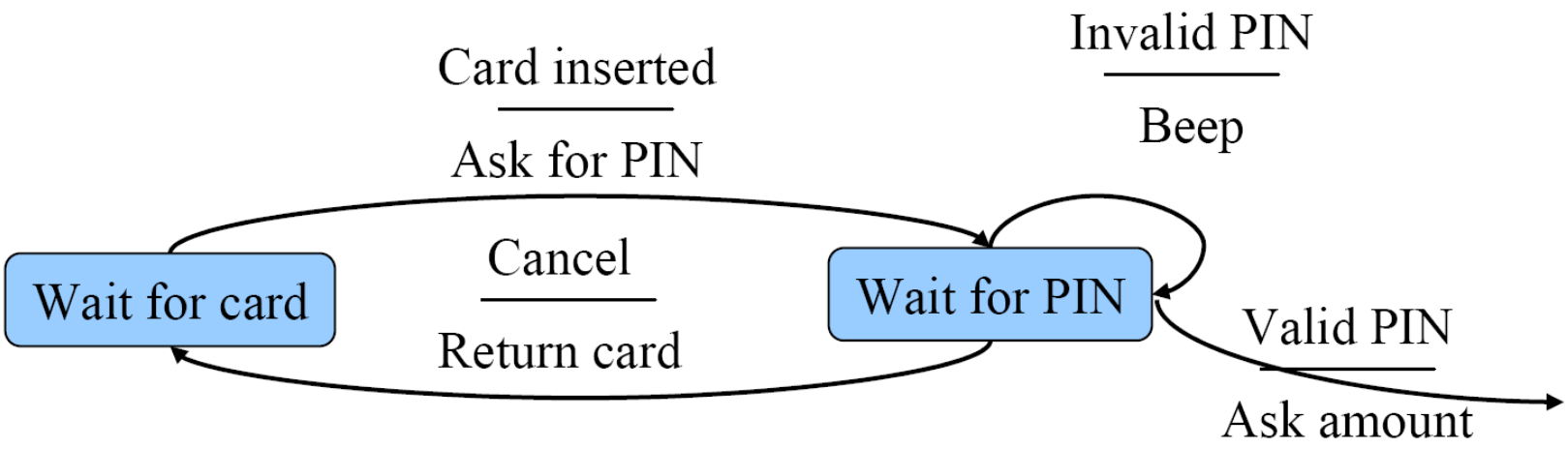
<u>Solution:</u>

- New algorithms can produce large covering arrays, quickly

ACTS Tool (NIST & UT Arlington)



State-Transition Testing



Create a set of test cases that triggers each event at least once

Error Guessing

- Exploratory testing, happy testing, ...
- Always worth including
- Can detect some failures that systematic techniques miss
- Consider
  - Past failures (fault models)
  - Intuition
  - Experience
  - Brain storming
  - ”What is the craziest thing we can do?”
  - Lists in literature

Recommended Exercises

- Chapter 6
  - 1, 3, 4, 5, 7, 8
- Chapter 9
  - 2, 4, 5, 6, 7, 8, 9

This Week

- Project
  - Find/read literature
- Lab 1
  - Thursday: White-box testing

Next Week

- Project
  - Report outline (Jan 31)
- Lab 2
  - Thursday : Black-box testing