

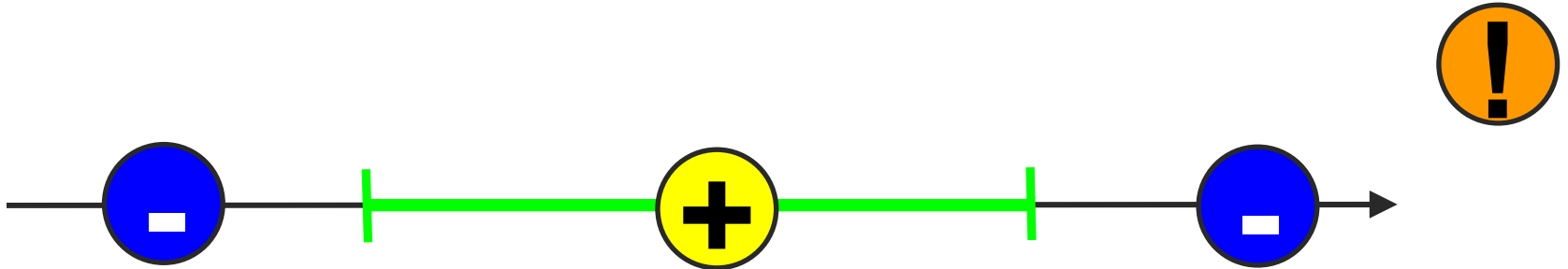
Equivalence Class and Boundary Value
Testing Methods –
Well-Known and Unexplored

Vineta Arnicāne

University of Latvia, Faculty of Computing

vineta.arnicane@lu.lv

Equivalence Classes



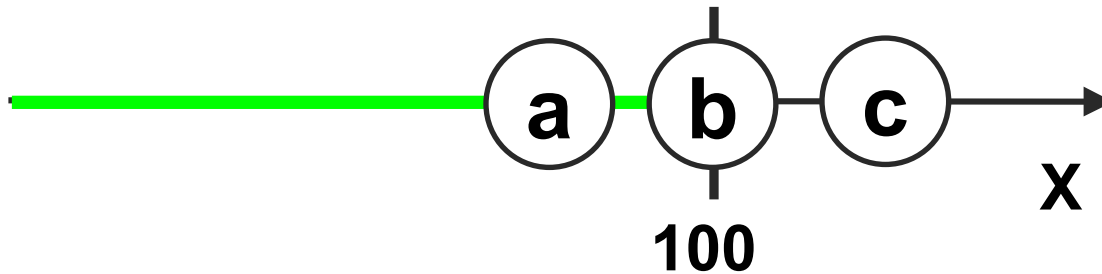
- Equivalence class – subset of program’s input domain which ceases the same type of program’s output or behavior
- Equivalence class testing – typical advice: “take **one** representant from each class”
 - ⊕ - valid values
 - ⊖ - invalid values of the same type
 - ! - invalid values – other

Boundary Values



- Boundary values – boundaries of equivalence classes, points close to them

Boundary Values – Analysis



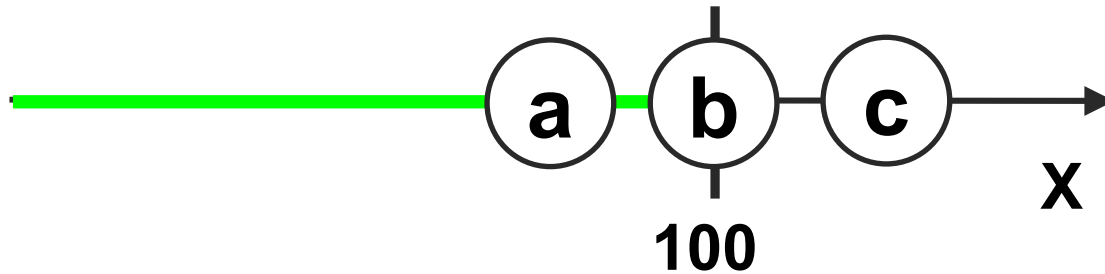
Code fragment:

If $X < 100$ then ...

Possible faults:

- $X \leq 100$
- $X = 100$
- $X > 100$
- $X \geq 100$

Boundary Values – Analysis



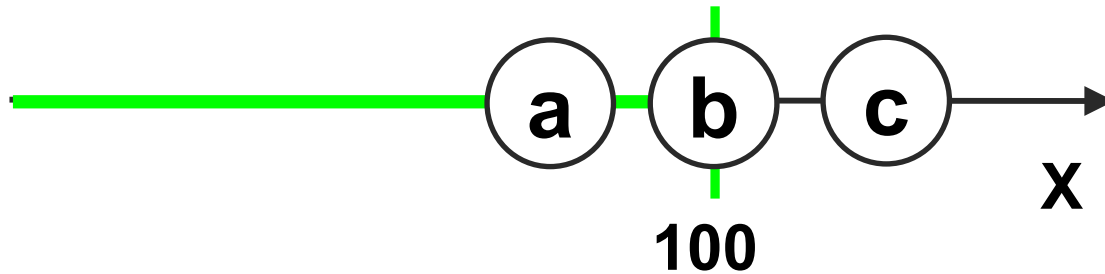
Code fragment:
If $X < 100$ then ...

| Ability of testcases to reveal faults | | | | |
|---------------------------------------|--------------|------|-------|-------|
| Testcases | | a=99 | b=100 | c=101 |
| Faults | $X \leq 100$ | No | Yes | No |
| | $X = 100$ | Yes | Yes | No |
| | $X > 100$ | Yes | No | Yes |
| | $X \geq 100$ | Yes | Yes | Yes |

Effective test suites:

- b and a
- b and c

Boundary Values – Analysis

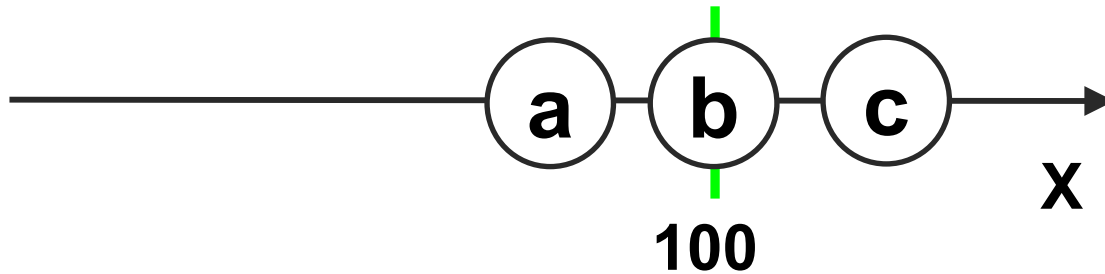


Code fragment:
If $X \leq 100$ then ...

| Ability of testcases to reveal faults | | | | |
|---------------------------------------|--------------|------|-------|-------|
| Testcases | | a=99 | b=100 | c=101 |
| Faults | $X < 100$ | No | Yes | No |
| | $X = 100$ | Yes | No | No |
| | $X > 100$ | Yes | Yes | Yes |
| | $X \geq 100$ | Yes | No | Yes |

Effective
testsuite:
➤ b and a

Boundary Values – Analysis















Code fragment:
If **X=100** then ...

| Ability of testcases to reveal faults | | | | |
|---------------------------------------|---------|------|-------|-------|
| Testcases | | a=99 | b=100 | c=101 |
| Faults | X < 100 | Yes | Yes | No |
| | X ≤ 100 | Yes | No | No |
| | X > 100 | No | Yes | Yes |
| | X ≥ 100 | No | No | Yes |

Effective testsuite:

➤ **a** and **c**

Testsuites that Reveal All Faults

| Code\Testsuite | (a,b) | (a,c) | (b,c) | (a,b,c) |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $X < 100$ |   | |  |   |
| $X \leq 100$ |   | | |   |
| $X = 100$ | |  | |   |



- testsuite reveals all 4 faults

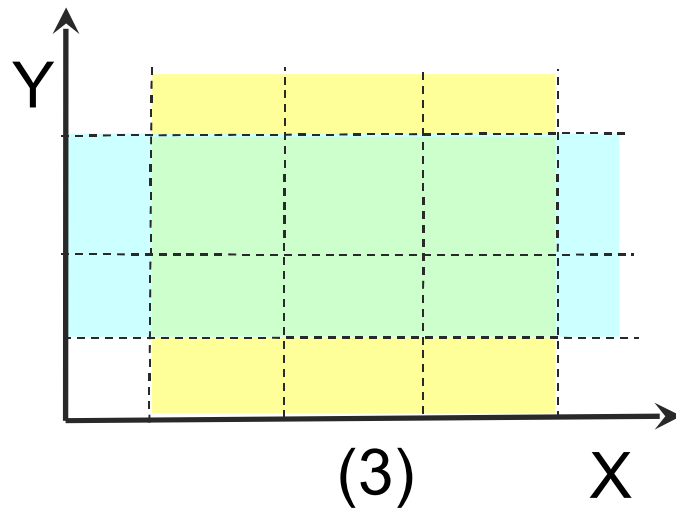
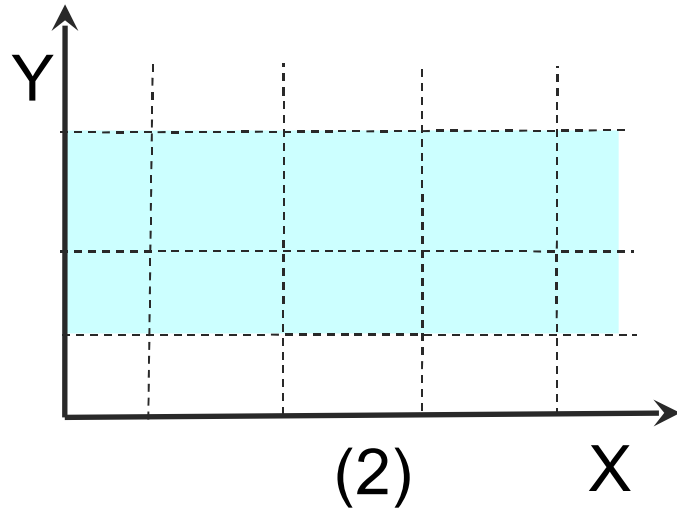
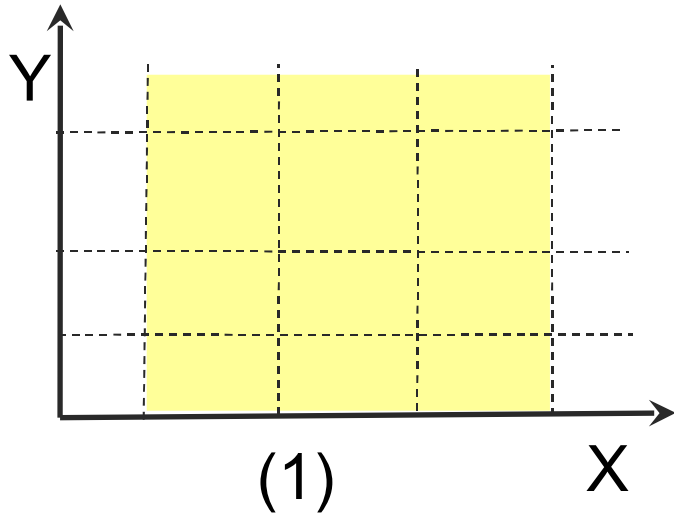


– testsuite has testcase with valid value

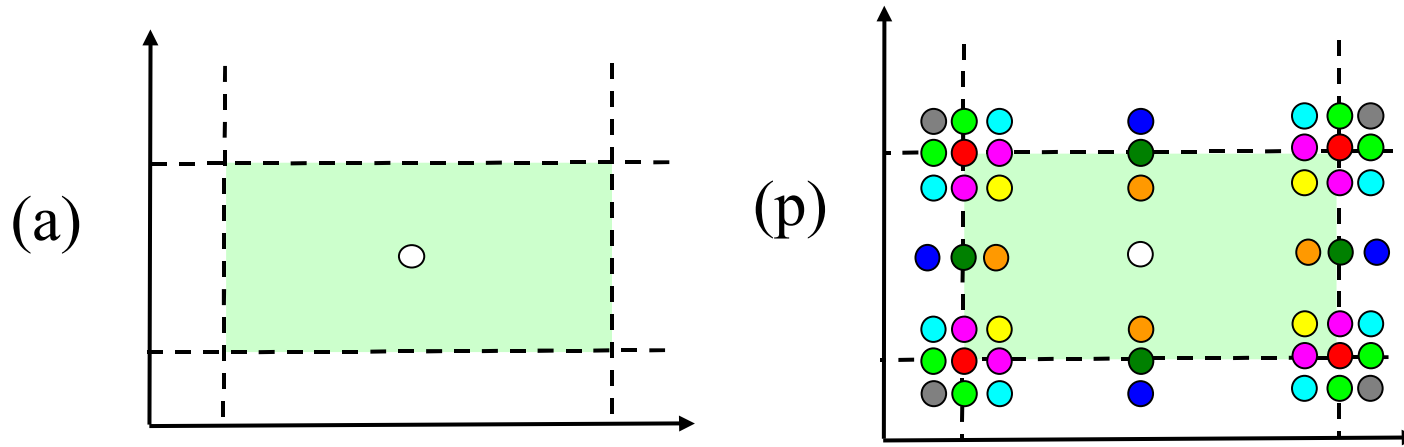
Three testcases at boundary **are necessary**

- only for equality expression and
- only if tester have to show that program works properly

Example of Equivalence Classes for 2 Parameters



Resource Consumption of Testing Methods

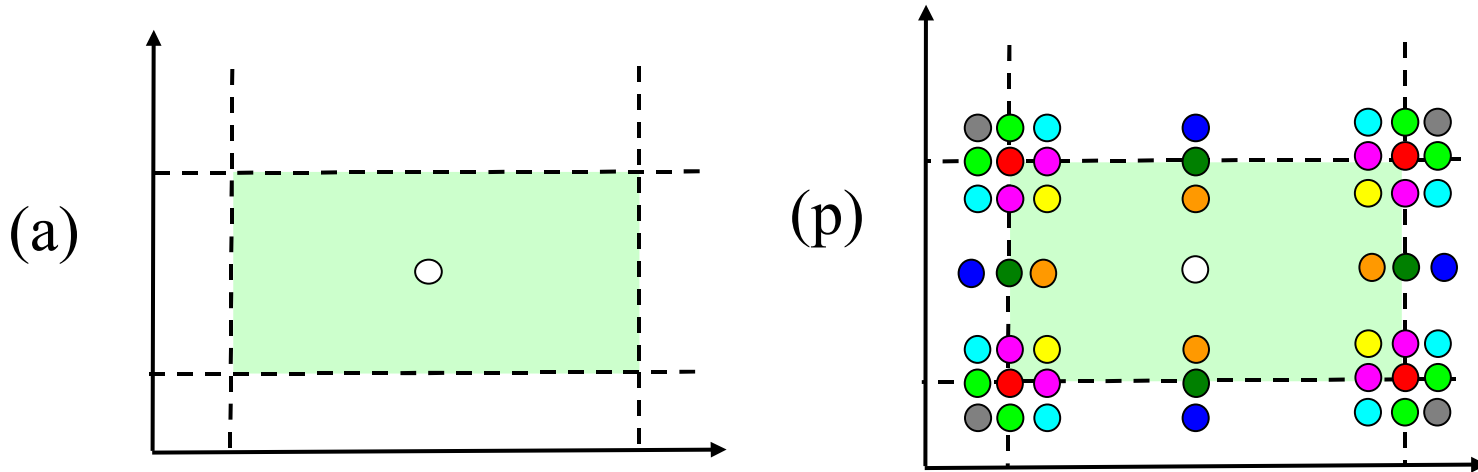


➤ **Assumption** – design and documenting time of one testcase – **10 minutes**, then overall design time of testsuite:

(a) **10 minutes**

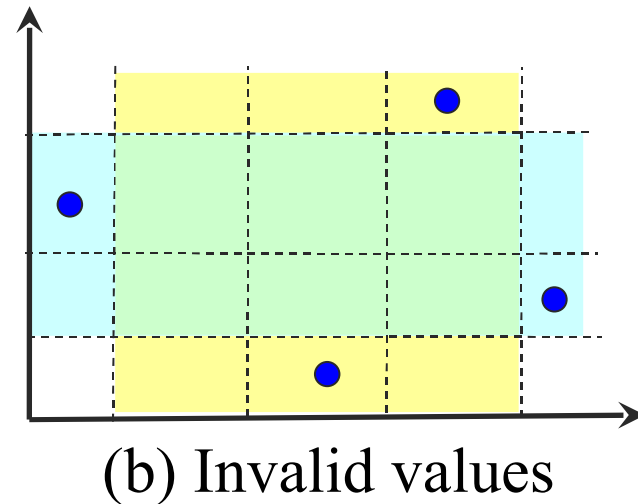
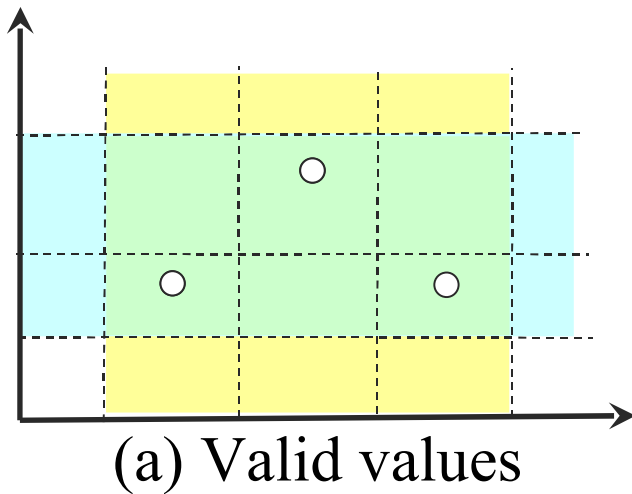
(p) **490 minutes** or more than 8 hours

Testing Intensity of Different Methods for Two Parameters



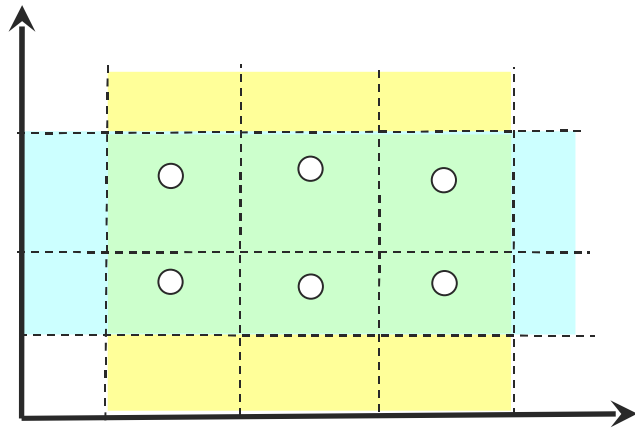
- Which values to test – valid and/or invalid?
- How combine chosen values of different parameters of program?

Equivalence Class Testing Methods

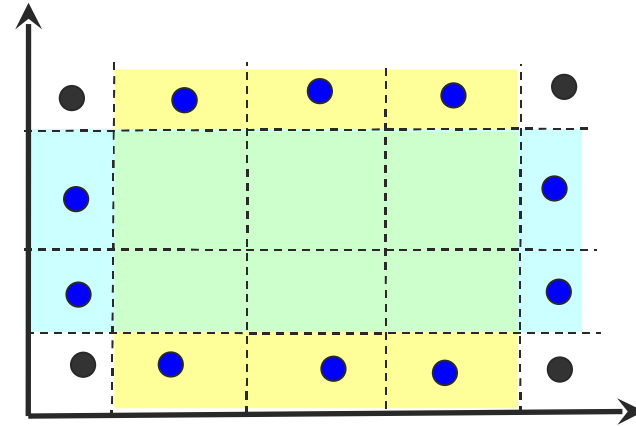


- Each class used coverage
- Single fault assumption

Equivalence Class Testing Methods



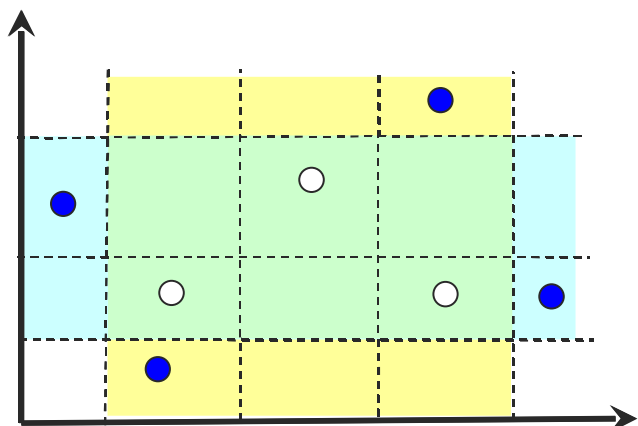
(c) Valid values



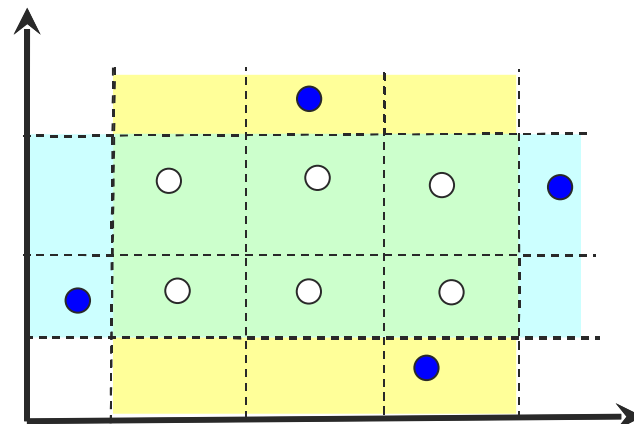
(d) Invalid values

- Pair-wise coverage of classes
- Multiple fault assumption for invalid values

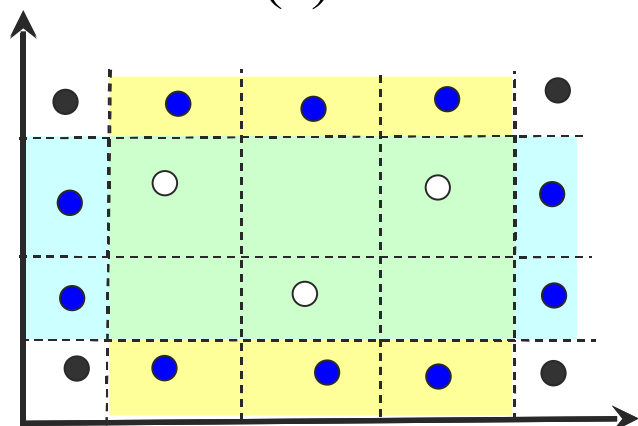
Combinations of Equivalence Class Testing Methods



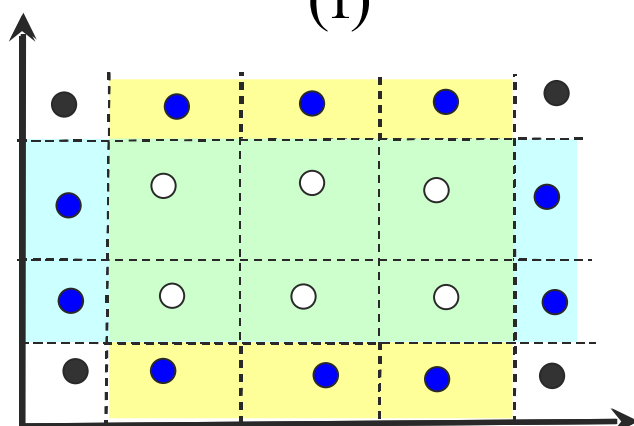
(e)



(f)

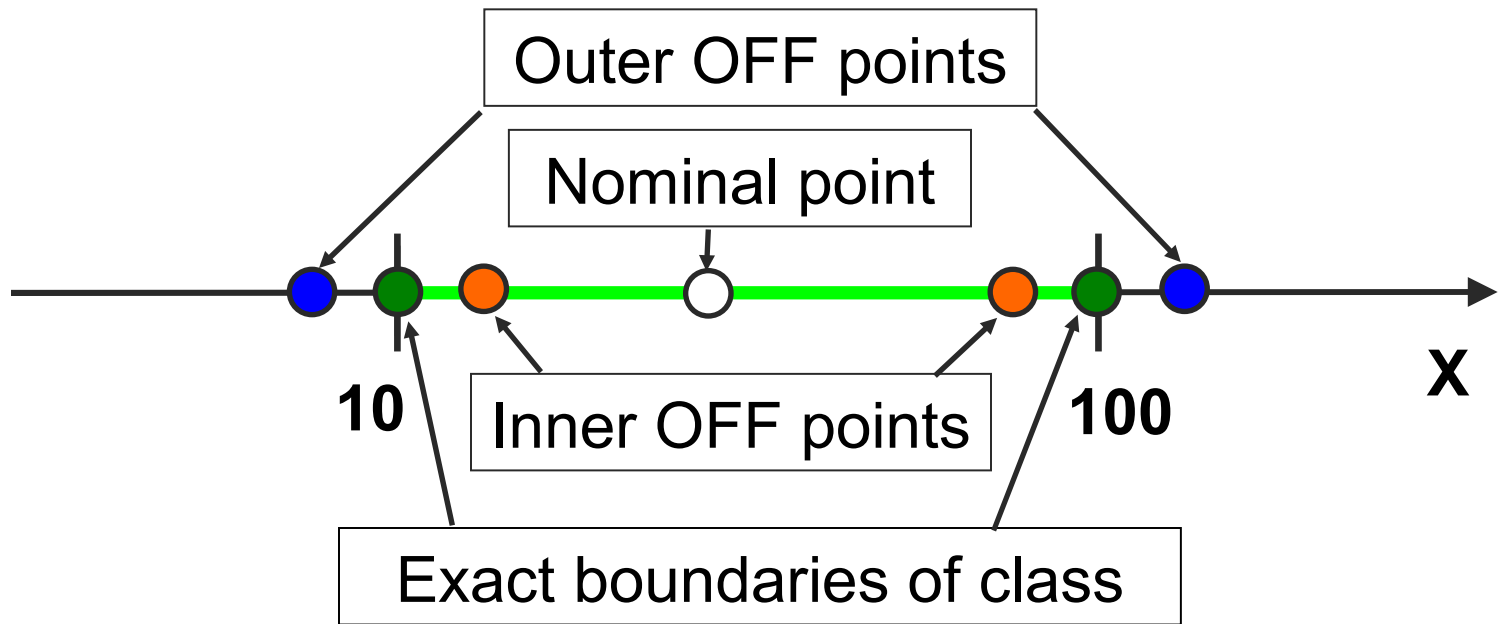


(g)



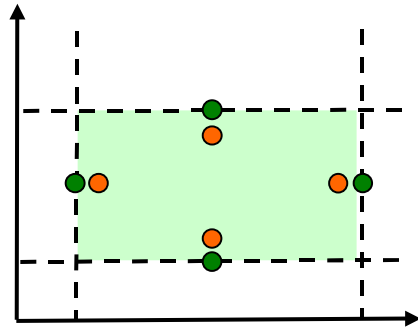
(h)

Boundary Values

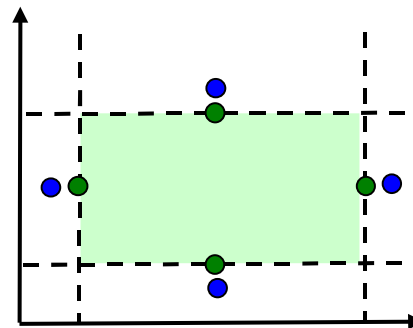


- Boundary values – boundaries of equivalence classes, points close to them

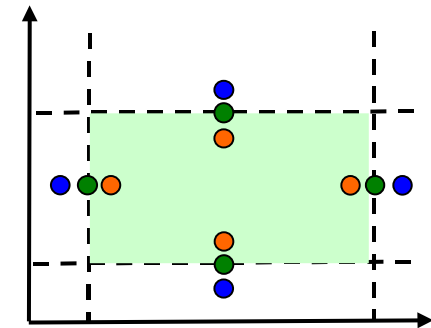
Combinations of Exact Boundary Points, OFF Points and Nominal Point



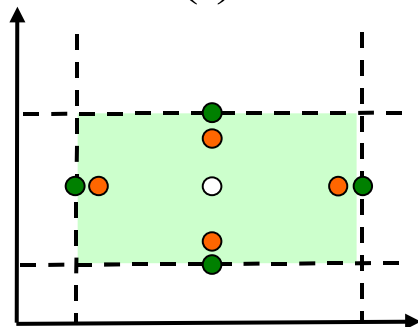
(i)



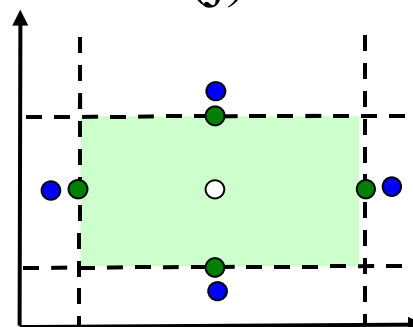
(j)



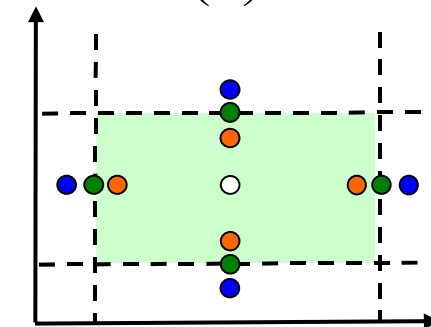
(k)



(l)



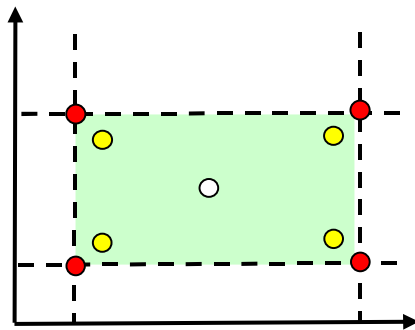
(m)



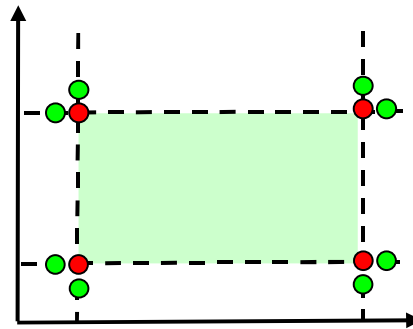
(n)

➤ Single fault assumption

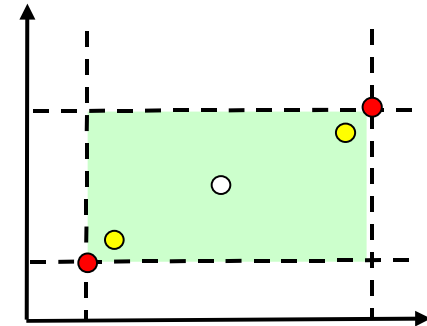
Combinations in Boundary Value Testing



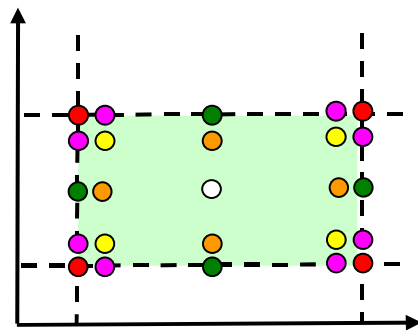
(o)



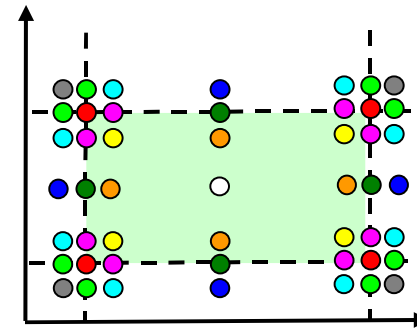
(p)



(q)



(r)

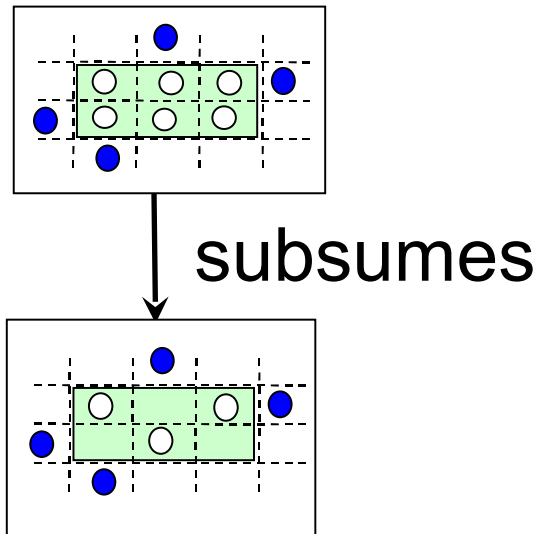


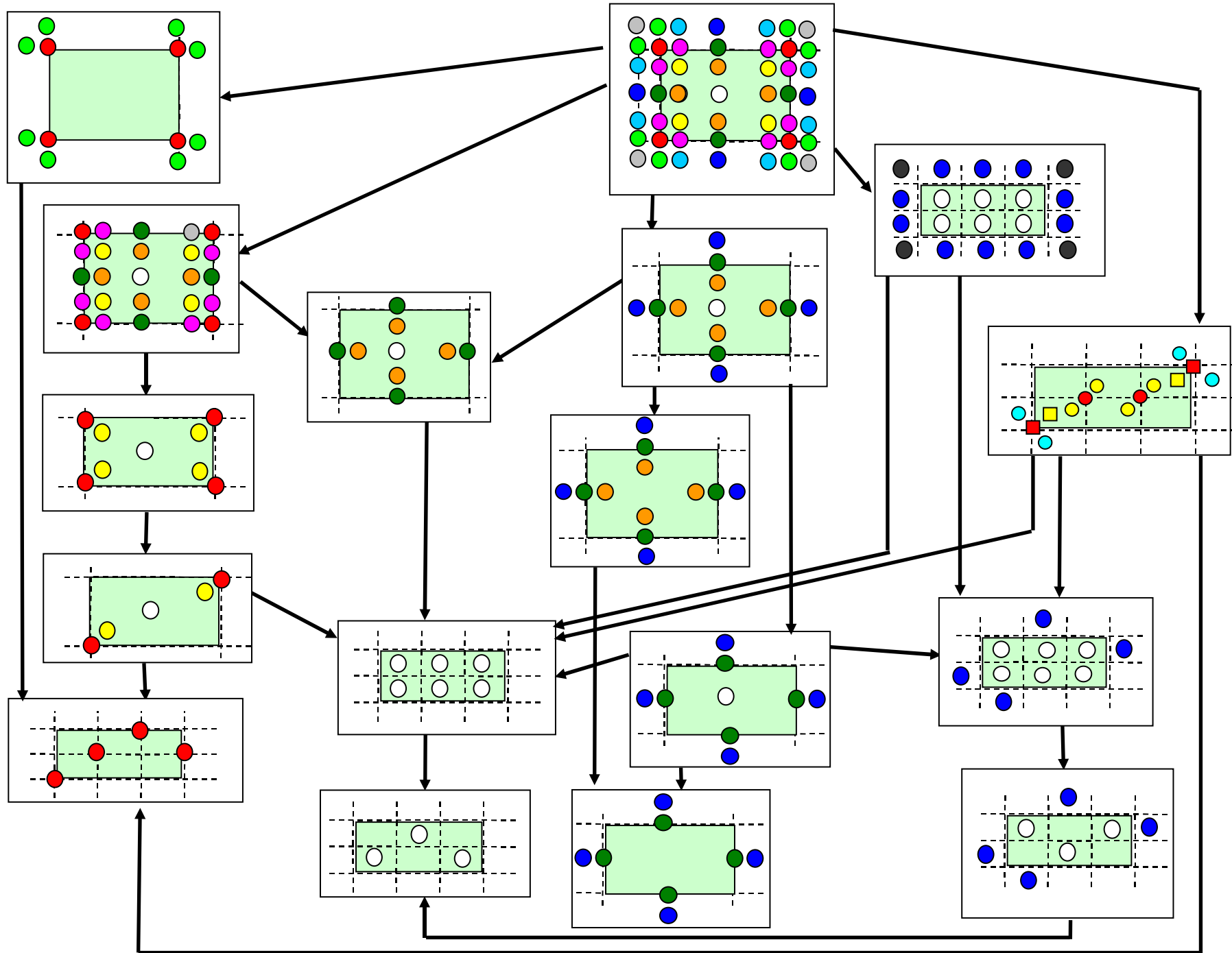
(s)

- Multiple fault assumption (o,p,q)
- Combination of single and multiple fault assumptions (r,s)

Subsume Relation

- If testing method C1 subsumes testing method C2 then every test suite generated by C1 is adequate for C2





Thanks!