- Predicate Testing

Predicate Testing

- Introduction
- Basic Concepts
- Predicate Coverage
- Summary

Motivation

- Predicates are expressions that can be evaluated to a boolean value, i.e., true or false.
- Many decision points can be encoded as a predicate, i.e., which action should be taken under what condition?
- Predicate-based testing is about ensuring those predicates are implemented correctly.

Applications

- Program-based: Predicates can be identified from the branching points of the source code
  - e.g.: if ((a > b) || c) { … } else { … }
- Specification-based: Predicates can be identified from both formal and informal requirements as well as behavioral models such as FSM
  - "if the printer is ON and has paper then send the document for printing"
- Predicate testing is required by US FAA for safety critical avionics software in commercial aircraft

Predicate Testing

- Introduction
- Basic Concepts
- Predicate Coverage
- Summary

Predicate

- A predicate is an expression that evaluates to a Boolean value
- Predicates may contain:
  - Boolean variables
  - Non-boolean variables that are compared with the relational operators $\{>, <, =, \geq, \leq, \neq\}$
  - Boolean function calls
- The internal structure is created by logical operators:

$\neg, \wedge, \vee, \rightarrow, \oplus, \leftrightarrow$

Logical operators

$\neg$ – the *negation* operator
$\wedge$ – the *and* operator
$\vee$ – the *or* operator
$\rightarrow$ – the *implication* operator
$\oplus$ – the *exclusive or* operator
$\leftrightarrow$ – the *equivalence* operator

Clause

- A clause is a predicate that does not contain any of the logical operators
- Example: $(a = b) \vee C \wedge p(x)$ has three clauses:
  - a relational expression $(a = b)$,
  - a boolean variable C,
  - a boolean function call $p(x)$

Predicate Faults

- An incorrect Boolean *operator* is used
- An incorrect Boolean *variable* is used
- Missing or extra Boolean variables
- An incorrect relational operator is used
- Parentheses are used incorrectly

Example

- Assume that $(a < b) \vee (c > d) \wedge e$ is a correct Boolean expression:
  - $(a < b) \wedge (c > d) \wedge e$
  - $(a < b) \vee (c > d) \wedge f$
  - $(a < b) \vee (c > d)$
  - $(a = b) \vee (c > d) \wedge e$
  - $(a = b) \vee (c \leq d) \wedge e$
  - $(a < b \vee c > d) \wedge e$

Predicate Testing

- Introduction
- Basic Concepts
- Predicate Coverage
- Program-Based Predicate Testing
- Summary

Abbreviations

- P is the set of predicates
- p is a single predicate in P
- C is the set of clauses in P
- C_p is the set of clauses in predicate p
- c is a single clause in C

Predicate Coverage (PC)

- The first (and simplest) two criteria require that each predicate and each clause be evaluated to both true and false and each clause be evaluated to both true and false

- For each predicate p, TR contains two requirements: p evaluates to true, and p evaluates to false.
- Example: $p = ((a > b) \vee C) \wedge p(x))$

	a	b	C	p(x)
1	5	4	true	true
2	5	6	false	false

Predicate Coverage Example

$$p = ((a < b) \vee D) \wedge (m >= n*o)$$

Predicate = true

a = 5, b = 10, D = true, m = 1, n = 1, o = 1

$= (5 < 10) \vee true \wedge (1 >= 1*1)$

$= true \vee true \wedge$ TRUE

= true

Predicate = false

a = 10, b = 5, D = false, m = 1, n = 1, o = 1

$= (10 < 5) \vee false \wedge (1 >= 1*1)$

$= false \vee false \wedge$ TRUE

= false

Clause Coverage (CC)

- For each clause c, TR contains two requirements: c evaluates to true, and c evaluates to false.
- Example: $((a > b) \vee C) \wedge p(x))$

	a	b	C	p(x)
1	5	4	true	true
2	5	6	false	false

"condition coverage" in literature

Clause Coverage Example

$P = ((a < b) \vee D) \wedge (m >= n*o)$

(a < b) = true

a = 5, b = 10

(a < b) = false

a = 10, b = 5

D = true

D = true

D = false

D = false

m >= n*o = true

m = 1, n = 1, o = 1

m >= n*o = false

m = 1, n = 2, o = 2

Two tests

1) a = 5, b = 10, D = true, m = 1, n = 1, o = 1

2) a = 10, b = 5, D = false, m = 1, n = 2, o = 2

Predicate vs. Clause Coverage

- Does predicate coverage subsume clause coverage? Does clause coverage subsume predicate coverage?
- Example: $p = a \vee b$

	a	b	$a \vee b$
1	T	T	T
2	T	F	T
3	F	T	T
4	F	F	F

Naturally, we want to test both the predicate and individual clauses.

Predicate and Clause Coverage

- CC does not always ensure PC
- This is, we can satisfy CC without causing the predicate to be both true or false
- This is definitely not what we want!!
  - We need to come up with other approaches

Combinatorial Coverage (CoC)

- For each predicate p, TR has test requirements for the clauses in p to evaluate to each possible combination of truth values
- Example: $(a \vee b) \wedge c$

CoC requires every possible combination

	a	b	c	$(a \vee b) \wedge c$
1	T	T	T	T
2	T	T	F	F
3	T	F	T	T
4	T	F	F	F
5	F	T	T	T
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

Combinatorial Coverage (CoC) – Exercise -1

Write all the clauses and the CoC of the given predicate:

$$P = ((a > b) \vee C) \wedge p(x)$$

Combinatorial Coverage (CoC) – Exercise -1

$P = ((a > b) \vee C) \wedge p(x)$

$P = (X \vee Y) \wedge Z$

	X	Y	Z	Predicate
1	F	F	F	F
2	F	F	T	F
3	F	T	F	F
4	F	T	T	T
5	T	F	F	F
6	T	F	T	T
7	T	T	F	F
8	T	T	T	T

Z is more important clause in this predicate than the others

Combinatorial Coverage (CoC)

What is the problem with combinatorial coverage ?

Combinatorial coverage is very expensive if we have multiple clauses in the predicate.

- 2^n possibilities, which n is number of independent clauses.

Active Clause

- Major clause
  - The clause which is being focused upon;
- Minor clause
  - All other clauses in the predicate (everything else).
- Determination:
  - A clause c_i in predicate p, called the major clause, determines p if and only if the values of the remaining minor clauses c_i are such that changing c_i changes the value of p (c_i Controls the behavior)

- *In the previous example, if we chose Z as Major clause, when it has value of "False", it doesn't matter what the other clauses are, but when it is "True", it does matter what other clauses are*

Determining Predicates

$P = A \lor B$

if B = *true*, p is always true.

so if B = *false*, A determines p.

if A = *false*, B determines p.

$P = A \land B$

if B = *false*, p is always false.

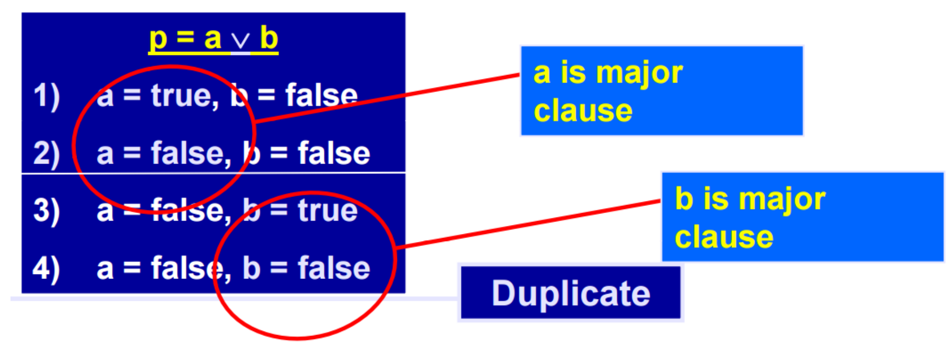
so if B = *true*, A determines p.

if A = *true*, B determines p.

Active Clause Coverage (ACC)

- For each predicate p and each major clause c of p, choose minor clauses so that c determines p. TR has two requirements for each c: c evaluates to true and c evaluates to false.

	a	b
$c_i = a$	**T**	f
	F	f
$c_i = b$	f	**T**
	f	**F**



Two of these requirements are identical, so we end up with **three distinct** test requirements for active clause coverage for the predicate a ∨ b, namely, {(a = true, b = false), (a = false, b = true), (a = false, b = false)}

Active Clause Coverage (ACC) – Example (1)

```java
public static void printHonorRollStatus(double cummulativeGPA,
        double termGPA, int creditsCompleted, boolean fullTimeStatus) {
    // Determine if the student is on the deans list.
    if ((creditsCompleted > 30) && (cummulativeGPA > 3.20)
            && (fullTimeStatus == true) && (termGPA > 2.0)) {
        System.out.println("You are on the dean's list.");
    } else if ((creditsCompleted > 30) && (cummulativeGPA > 3.70)
            && (fullTimeStatus == true) && (termGPA > 2.0)) {
        System.out.println("You are on the high honors dean's list.");
    } else if ((creditsCompleted > 30) && (cummulativeGPA > 2.0)
            && (fullTimeStatus == true) && (termGPA > 3.2)) {
        System.out.println("You are on the honor list.");
    } else {
```

Active Clause Coverage (ACC) – Example (2)

CC	CGPA	FT	TGPA	Predicate
29	3.3	True	2.4	False
31	3.3	True	2.4	True
31	3.0	True	2.4	False
31	3.3	True	2.4	True
31	3.3	False	2.4	False
31	3.3	True	2.4	True
31	3.3	True	1.9	False
31	3.3	True	2.4	True

- The Green cells indicate active clauses,
- The Orange color cells indicate minor clauses
- We can test this with only 5 test cases

Resolving the Ambiguity

$p = a \vee (b \wedge c)$

Major clause : a

a = true, b = false, c = true

a = false, b = false, c = false

Is this allowed ?

- This question caused confusion among testers for years
- Considering this carefully leads to three separate criteria :
  - Minor clauses do not need to be the same (GACC)
  - Minor clauses do need to be the same (RACC)
  - Minor clauses force the predicate to become both true and false (CACC)

General Active Clause Coverage (GACC)

- The same as ACC, and it does not require the minor clauses have the same values when the major clause evaluates to true and false.
- Does GACC subsume predicate coverage?

	a	b	c	$a \wedge (b \vee c)$
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

Correlated Active Clause Coverage (CACC)

- The same as ACC, but it requires the entire predicate to be true for one value of the major clause and false for the other.

CACC (2)

- Example: $p = a \wedge (b \vee c)$

For *a* to determine the value of p, the expression b v c must be true

This can be achieved in three ways: b true and c false, b false and c true, and both b and c true

	a	b	c	$a \wedge (b \vee c)$
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

	a	b	c	$a \wedge (b \vee c)$
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F

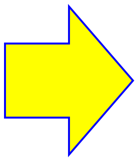
Rows 4 and 8 are missing because a is not active in the two rows.

(1,5) (1,6) (1,7)
(2,5)(2,6)(2,7)
(3,5)(3,6)(3,7)

Restricted Active Clause Coverage (RACC)

- The same as ACC, but it requires the minor clauses have the same values when the major clause evaluates to true and false.
- Example: $p = a \wedge (b \vee c)$

	a	b	c	$a \wedge (b \vee c)$
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F



	a	b	c	$a \wedge (b \vee c)$
1	T	T	T	T
5	F	T	T	F
2	T	T	F	T
6	F	T	F	F
3	T	F	T	T
7	F	F	T	F

RACC can only be satisfied by one of the three pairs
(1,5) (2,6) (3,7)

CACC and RACC

	a	b	c	a ∧ (b ∨ c)
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

major clause

P_a : b=true or c = true

CACC can be satisfied by choosing any of rows 1, 2, 3 AND any of rows 5, 6, 7 – a total of nine pairs

	a	b	c	a ∧ (b ∨ c)
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

RACC can only be satisfied by row pairs (1, 5), (2, 6), or (3, 7)

Only three pairs

Inactive Clause Coverage (ICC)

- Active clause coverage criteria ensure that “major” clauses do affect the predicates
- Inactive clause coverage takes the opposite approach - major clauses do not affect the predicates

- For each predicate p and each major clause c of p, choose minor clauses so that c does not determine p. TR has four requirements for each clause c:
  - 1) c evaluates to true with p = true
  - 2) c evaluates to false with p = true
  - 3) c evaluates to true with p = false
  - 4) c evaluates to false with p = false.

General and Restricted ICC

- GICC does not require the values of the minor clauses to be the same when the major clause evaluates to true and false.

- RICC requires the values the minor clauses to be the same when the major clause evaluates to true and false.

Making Clauses Determine a Predicate

- Let p be a predicate and c a clause. Let $p_{c=true}$ (or $p_{c=false}$) be the predicate obtained by replacing every occurrence of c with true (or false)
- The following expression describes the exact conditions under which the value of c determines the value of p:

$$\boxed{p_c = p_{c=true} \oplus p_{c=false}}$$

Example-1

$\underline{p = a \vee b}$

$p_a = p_{a=true} \oplus p_{a=false}$
$= (true \vee b) \text{ XOR } (false \vee b)$
$= true \text{ XOR } b$
$= \neg b$

$\underline{p = a \wedge b}$

$p_a = p_{a=true} \oplus p_{a=false}$
$= (true \wedge b) \oplus (false \wedge b)$
$= b \oplus false$
$= b$

$\underline{p = a \vee (b \wedge c)}$

$p_a = p_{a=true} \oplus p_{a=false}$
$= (true \vee (b \wedge c)) \oplus (false \vee (b \wedge c))$
$= true \oplus (b \wedge c)$
$= \neg (b \wedge c)$
$= \neg b \vee \neg c$

"NOT b v NOT c" means either b or c can be false

Example-2

Compute (and simplify) the conditions under which each of the clauses determines predicate *p.*

$p = a \wedge (\neg b \vee c)$

$p_a = \neg b \vee c$

$p_b = a \wedge \neg c$

$p_c = a \wedge b$

Note that the last step in the simplification may not be immediately obvious. If it is not, try constructing the truth table. For instance, for $(a \wedge c) \oplus a$. The computation for p_c is equivalent and yields the solution $a \wedge \neg c$

Example-3

Compute (and simplify) the conditions under which each of the clauses determines predicate *p.*

Consider $p = (a \wedge b) \vee (a \wedge \neg b)$

$p_a = \ldots$

$p_b = \ldots$

Example-3(1)

$\underline{p = (a \wedge b) \vee (a \wedge \neg b)}$

$$
\begin{aligned}
p_a &= p_{a=true} \oplus p_{a=false} \\
&= ((true \wedge b) \vee (true \wedge \neg b)) \oplus ((false \wedge b) \vee (false \wedge \neg b)) \\
&= (b \vee \neg b) \oplus false \\
&= true \oplus false \\
&= true
\end{aligned}
$$

$\underline{p = (a \wedge b) \vee (a \wedge \neg b)}$

$$
\begin{aligned}
p_b &= p_{b=true} \oplus p_{b=false} \\
&= ((a \wedge true) \vee (a \wedge \neg true)) \oplus ((a \wedge false) \vee (a \wedge \neg false)) \\
&= (a \vee false) \oplus (false \vee a) \\
&= a \oplus a \\
&= false
\end{aligned}
$$

- ***a*** **always determines the value of this predicate**
- ***b*** **never determines the value – *b* is irrelevant !**

Example-4

$p = a \wedge (\neg b \vee c)$

	a	b	c	p	p_a	p_b	p_c
1	T	T	T	T	T	F	T
2	T	T	F	F	F	T	T
3	T	F	T	T	T	F	F
4	T	F	F	T	T	T	F
5	F	T	T	F	T	F	F
6	F	T	F	F	F	F	F
7	F	F	T	F	T	F	F
8	F	F	F	F	T	F	F

- Conditions under which each of the clauses determines p
  - p_a: $(\neg b \vee c)$
  - p_b: $a \wedge \neg c$
  - p_c: $a \wedge b$

- All pairs of rows satisfying GACC
  - a: {1,3,4} x {5,7,8}, b: {(2,4)}, c:{(1,2)}
- All pairs of rows satisfying CACC
  - Same as GACC
- All pairs of rows satisfying RACC
  - a: {(1,5),(3,7),(4,8)}
  - Same as CACC pairs for b, c
- GICC
  - a: {(2,6)} for p=F, no feasible pair for p=T
  - b: {5,6}x{7,8} for p=F, {(1,3) for p=T
  - c: {5,7}x{6,8} for p=F, {(3,4)} for p=T
- RICC
  - a: same as GICC
  - b: {(5,7),(6,8)} for p=F, {(1,3)} for p=T
  - c: {(5,6),(7,8)} for p=F, {(3,4)} for p=T

Boolean Algebra Laws

- ***Commutativity Laws***

$$a \vee b = b \vee a$$
$$a \wedge b = b \wedge a$$
$$a \oplus b = b \oplus a$$

- ***Associativity Laws***

$$(a \vee b) \vee c = a \vee (b \vee c)$$
$$(a \wedge b) \wedge c = a \wedge (b \wedge c)$$
$$(a \oplus b) \oplus c = a \oplus (b \oplus c)$$

- ***Distributive Laws***

$$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$$
$$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$$

- ***DeMorgan's Laws***

$$\neg(a \vee b) = \neg a \wedge \neg b$$
$$\neg(a \wedge b) = \neg a \vee \neg b$$

- ***Negation Laws***

$$\neg(\neg a) = a$$
$$\neg a \vee a = true$$
$$\neg a \wedge a = false$$
$$a \vee \neg a \wedge b = a \vee b$$

- ***AND Identity Laws***

$$false \wedge a = false$$
$$true \wedge a = a$$
$$a \wedge a = a$$
$$a \wedge \neg a = false$$

- ***OR Identity Laws***

$$false \vee a = a$$
$$true \vee a = true$$
$$a \vee a = a$$
$$a \vee \neg a = true$$

- ***XOR Identity Laws***

$$false \oplus a = a$$
$$true \oplus a = \neg a$$
$$a \oplus a = false$$
$$a \oplus \neg a = true$$

- ***XOR Equivalence Laws***

$$a \oplus b = (a \wedge \neg b) \vee (\neg a \wedge b)$$
$$a \oplus b = (a \vee b) \wedge (\neg a \vee \neg b)$$
$$a \oplus b = (a \vee b) \wedge \neg(a \wedge b)$$

Infeasible Test Requirements

- Consider the predicate: $(a > b \wedge b > c) \vee c > a$

$(a > b) = true, (b > c) = true, (c > a) = true$ is infeasible

- As with graph-based criteria, infeasible test requirements have to be recognized and ignored

- Recognizing infeasible test requirements is hard, and in general, undecidable

- Software testing is inexact – engineering, not science

Finding Satisfying Values

How to choose values that satisfy a given coverage goal?

Example (1)

- Consider $p = (a \vee b) \wedge c$:

a	x < y
b	done
c	List.contains(str)

How to choose values to satisfy predicate coverage?

Example (2)

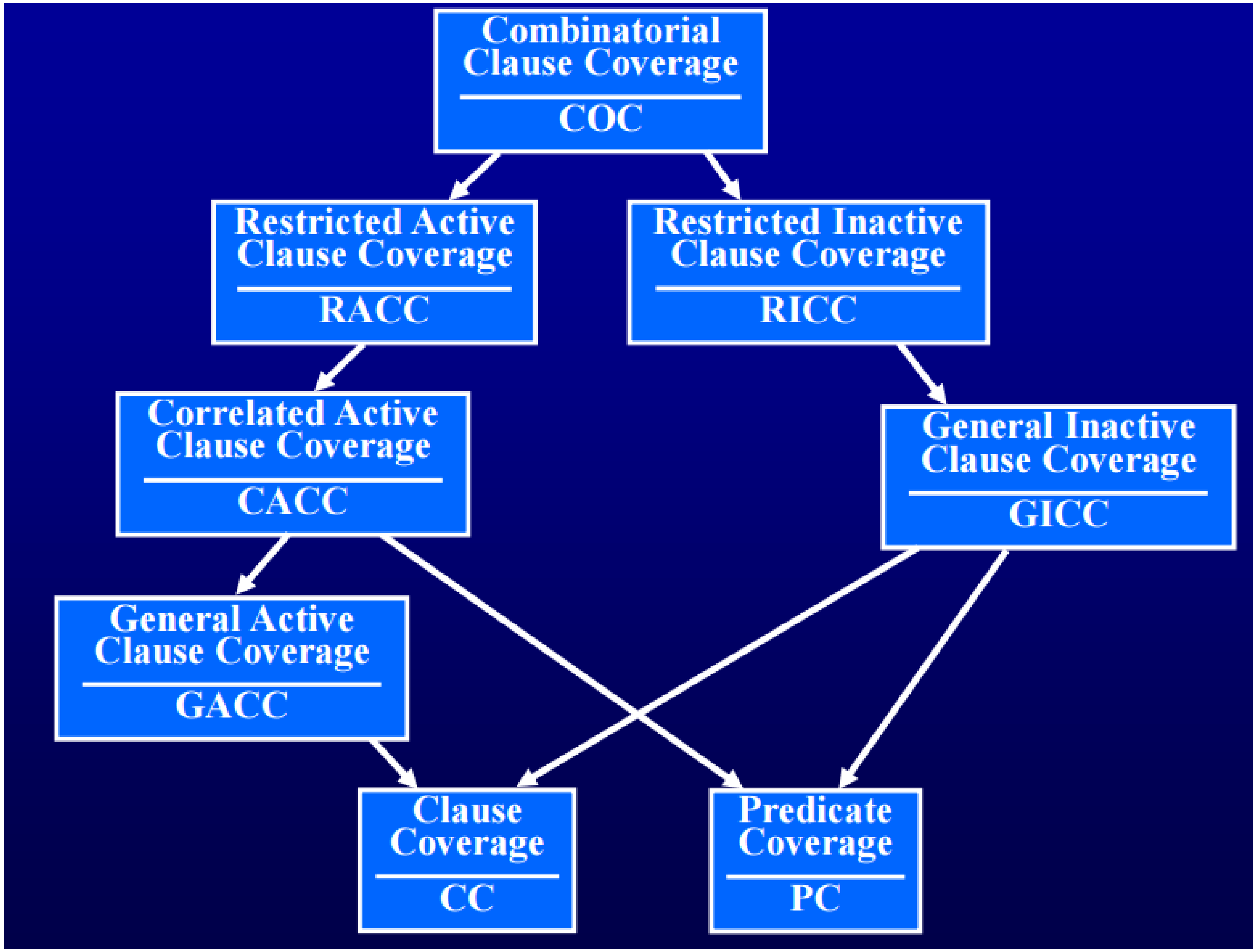
	a	b	c	p
1	t	t	t	t
2	t	t	f	f
3	t	f	t	t
4	t	f	f	f
5	f	t	t	t
6	f	t	f	f
7	f	f	t	f
8	f	f	f	f

$\{1, 3, 5\} \times \{2, 4, 6, 7, 8\}$

Suppose we choose {1, 2}.

a	b	c
x = 3 y = 5	done = true	List = [“Rat”, “cat”, “dog”] str = “cat”
x = 0, y = 7	done = true	List = [“Red”, “White”] str = “Blue”

Logic Coverage Criteria Subsumption



Recap

- Predicate testing is about ensuring that each decision point is implemented correctly.
- If we flip the value of an active clause, we will change the value of the entire predicate.
- Different active clause criteria are defined to clarify the requirements on the values of the minor clauses.