Introduction to Software Testing
Semantic Logic Coverage (Ch. 8.1)

Software Testing & Maintenance

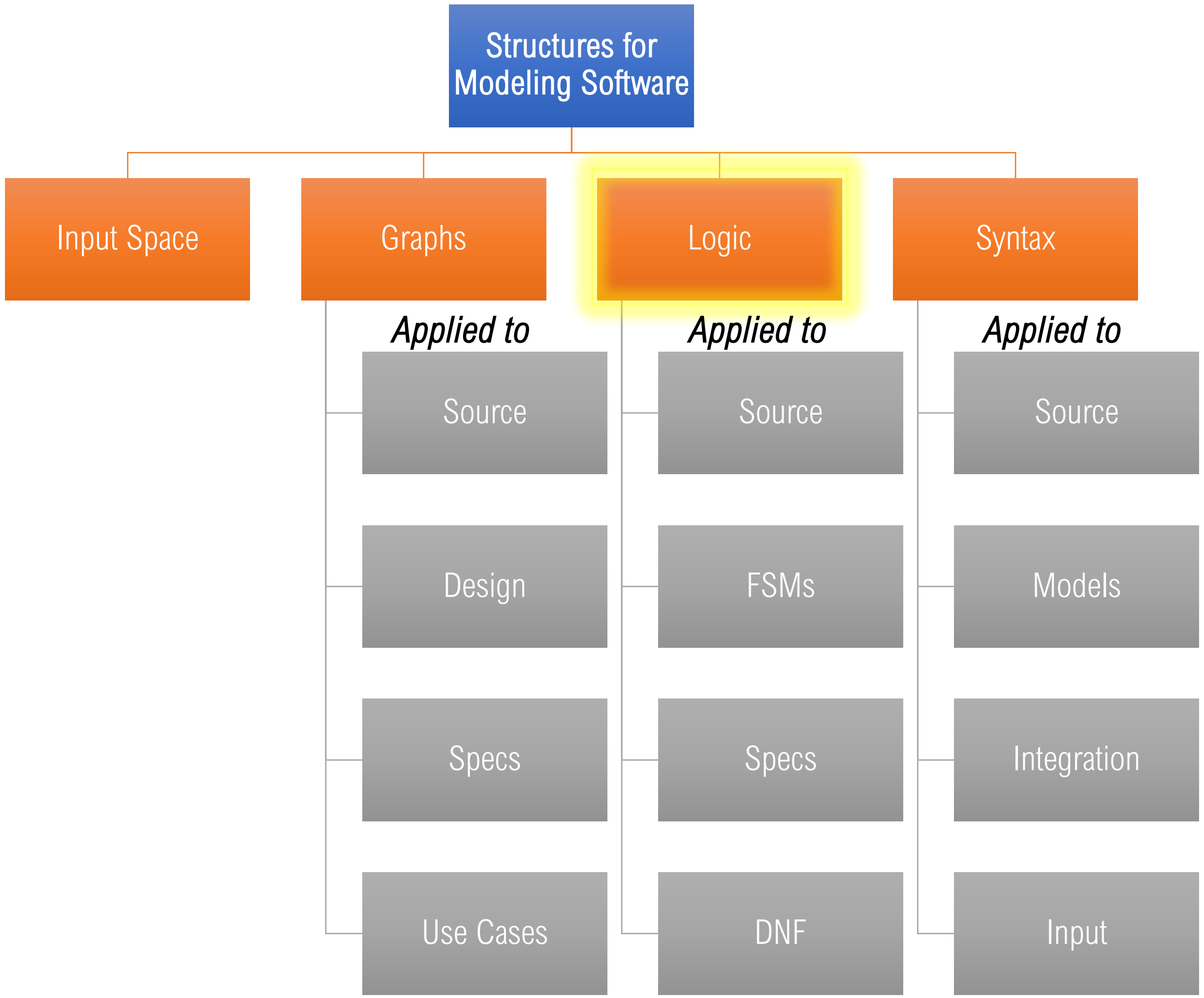
SWE 437/637

go.gmu.edu/SoftwareTestingFall24

Dr. Brittany Johnson-Matthews

(Dr. B for short)

Logic Coverage



Semantic Logic Coverage (8.1)

Logical expressions can come from many sources

- Decisions in programs
- Decisions in **UML** activity graphs and **finite state machines**
- **Requirements**, both formal and informal
- **SQL** queries

Logic coverage is **required by the US Federal Aviation Administration** for safety critical software

- Used by other transportation industries

Used by Electronc Arts (EA) game company

- FIFA, Battlefield, …

Tests are intended to **choose some subset of the total number of truth assignments** to the expressions

Logic predicates and clauses

A ***predicate*** is an expression that evaluates to a **boolean** value.

Predicates can contain:

- boolean variables
- non-boolean variables that contain >, <, ==, >=, <=, !=
- boolean function calls

Internal structure is created by *logical operators*

- ¬ or ! – the negation operator
- ∧ or & – the and operator
- ∨ or | – the or operator
- → – the implication operator
- ⊕ or **xor** – the exclusive or operator
- ↔ – the equivalence operator

A **clause** is a predicate with no logical operators.

Example

P = (a & (b | c))

P has three clauses:

a, b, and c

A more complex example

$$(a < b) \vee f(z) \wedge d \vee (m >= n*o)$$

This predicate has four clauses:

- **(a < b)** : a relational expression
- **f(z)** : a boolean function
- **d** : a boolean variable
- **(m >= n*o)** : a relational expression

Most predicates only have 1 clause!

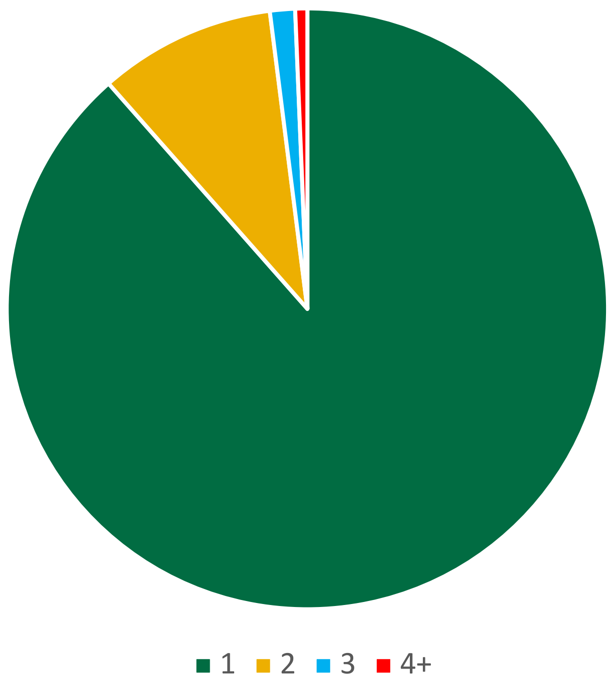
From a study of 63 open-source programs which contained > 400,000 predicates:

88.5% had only 1 clause

9.5% had 2 clauses

1.35% had 3 clauses

Only **0.65%** had 4 or more clauses



Logic Coverage Criteria (8.1.1)

We use predicates in testing to:

Develop a model of the software as one or more predicates

Require tests to satisfy some combination of clauses

Abbreviations:

P is a set of predicates

p is a predicate in ***P***

C is the set of clauses in ***P***

Cp is the set of clauses in predicate ***p***

c is a clause in ***C***

Predicate Coverage

The first (and simplest) criterion requires that each predicate be evaluated to both true and false

DEFINITION

Predicate Coverage (PC) – For each *p* in *P*, *TR* contains two requirements: *p* evaluates to true, and *p* evaluates to false

When predicates come from conditions on graph edges, this is equivalent to *edge coverage*

Clause Coverage

Predicate coverage does not require evaluation of all clauses, so a more complete criterion is clause coverage

DEFINITION

Clause Coverage (CC) – For each ***c*** in *C*, *TR* contains two requirements: ***c*** evaluates to true, and ***c*** evaluates to false

Does clause coverage subsume predicate coverage?

Clause Coverage

Predicate coverage does not require evaluation of all clauses, so a more complete criterion is clause coverage

DEFINITION

Clause Coverage (CC) – For each ***c*** in ***C***, ***TR*** contains two requirements: ***c*** evaluates to true, and ***c*** evaluates to false

Does clause coverage subsume predicate coverage?

No! Consider $\boldsymbol{a} \vee \boldsymbol{b}$, the clauses can be satisfied with (*a*=true, *b*=false) and (*a*=false, *b*=true) but the predicate is always true

In-class Exercise

$(a < b) \lor d \land (m >= n*o)$

Give predicate coverage (**PC**) and clause coverage (**CC**) abstract tests for our example predicate.

"Abstract tests" include truth assignments for each clause, for example:
a = true

Predicate Coverage Example

$$(a < b) \lor d \land (m >= n*o)$$

Predicate must evaluate to true

Predicate must evaluate to false

Predicate Coverage Example

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

Predicate must evaluate to true

Example test case:

a=5, *b*=10, *d*=true, *m*=1, *n*=1, *o*=1

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

$= (\text{true} \vee \text{true}) \wedge (\text{true})$

= true

Predicate Coverage Example

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

Predicate must evaluate to true

Example test case:

a=5, *b*=10, *d*=true, *m*=1, *n*=1, *o*=1

= ((5 < 10) ∨ true) ∧ (1 >= 1*1)

= (true ∨ true) ∧ (true)

= true

Predicate must evaluate to false

Example test case:

a=10, *b*=5, *d*=false, *m*=1, *n*=1, *o*=1

= ((10 < 5) ∨ false) ∧ (1 >= 1*1)

= (false ∨ false) ∧ (true)

= false

Clause Coverage Example

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

Clauses must evaluate to true

In this case we arbitrarily choose to set all clauses to true at the same time, but that is not required

Clause Coverage Example

$$(a < b) \lor d \land (m >= n*o)$$

Clauses must evaluate to true

In this case we arbitrarily choose to set all clauses to true at the same time, but that is not required

$(a<b)$: $a=5$, $b=10$

d : true

$(m >= n*o)$: $m=1$, $n=1$, $o=1$

Clause Coverage Example

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

Clauses must evaluate to true

In this case we arbitrarily choose to set all clauses to true at the same time, but that is not required

$(a<b)$: a=5, b=10

d : true

$(m >= n*o)$: m=1, n=1, o=1

Clauses must evaluate to false

$(a<b)$: a=10, b=5

d : false

$(m >= n*o)$: m=1, n=2, o=2

Problems with PC and CC

PC does not **fully exercise** all the clauses, especially in the presence of short circuit evaluation

CC does not always **ensure PC**

- That is, we can satisfy CC without causing the predicate to be both true and false
- This is *definitely* not what we want !

The simplest solution is to test **all combinations** …

Combinatorial Coverage (CoC)

DEFINITION

Combinatorial Coverage (CoC) – For each p in P, TR contains a test requirement for the clauses in C_p to evaluate to each possible combination of truth values

Combinatorial coverage is conceptually simple and complete, but very expensive

- Results in 2^N *tests*, where N is the number of clauses

In-class Exercise

P = (a & (b | c))

Give abstract tests to satisfy combinatorial coverage (**CoC**) for the above predicate.

Hint: There should be 8

In-class Exercise

P = (a & (b | c))

Give abstract tests to satisfy combinatorial coverage (**CoC**) for our example predicate.

CoC

a=true, b=true, c=true
a=f, b=t, c=f
a !b c
a !b !c
!a b c
!a b !c
!a !b c
!a !b !c

Hint: There should be 8

Combinatorial Coverage (CoC)

This is simple, neat, clean, and comprehensive …

But can be **expensive**

- Impractical for predicates with more than 3 or 4 clauses

The literature has lots of suggestions – some confusing

The general idea is simple:

Test each clause independently from the other clauses

Getting the details right is hard

What exactly does "independently" mean ?

The book presents this idea as "***making clauses active***" …

Active Clauses (8.1.2)

Clause coverage has a **weakness** : The values do not always make a difference

Consider the CC tests for P = (a & (b | c)):

Test 1: (true & (true | true))

Test 2: (false & (false | false))

Clauses b and c are ignored!

To really test the results of a clause, the clause should be the **determining factor** in the value of the predicate

Active Clauses – Determination

Clause c_i determines the value of its predicate when the other clauses have certain values.

If c_i is changed, the value of the predicate changes

c_i is called the *major clause*

Other clauses are *minor clauses*

This is called *making the clause active*.

Determining Predicates

$P = A \vee B$	$P = A \wedge B$
if ***B* = *true***, *p* is always true.	if ***B* = *false***, *p* is always false.
so if ***B* = *false***, *A* determines *p*.	so if ***B* = *true***, *A* determines *p*.
if ***A* = *false***, *B* determines *p*.	if ***A* = *true***, *B* determines *p*.

Goal : Find tests for each clause when the clause determines the value of the predicate.

This is formalized in a **family of criteria** that have subtle, but very important, differences.

Active Clause Coverage

DEFINITION

Active Clause Coverage (ACC) – For each p in P and each major clause c_i in C_p, choose minor clauses c_j (j!=i) such that c_i determines p. *TR* has two requirements for c_i: c_i evaluates to true and c_i evaluates to false

Active Clause Coverage

$$p = a \lor b$$

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

Active Clause Coverage

$$p = a \lor b$$

	a	b	a ∨ b
1	T	T	T
2	T	F	T
3	F	T	T
4	F	F	F

Select ***a*** as the major clause, then choose values for minor clause ***b*** such that changing ***a*** changes ***p***, so ***a*** determines ***p*** and ***a*** is the *active clause*

Active Clause Coverage

$$p = a \vee b$$

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

Select ***a*** as the major clause, then choose values for minor clause ***b*** such that changing ***a*** changes ***p***, so ***a*** determines ***p*** and ***a*** is the *active clause*

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

Select ***b*** as the major clause, then choose values for minor clause ***a*** such that changing ***b*** changes ***p***, so ***b*** determines ***p*** and ***b*** is the *active clause*

In-class Exercise

Making clauses active

P = (a & (b | c))

Write truth values for **b and c** that make clause **a** active

For example: Pa : b=?? or c=??

Write truth values for **a and c** that make clause **b** active

Write truth values for **a and b** that make clause **c** active

In-class Exercise

Making clauses active

P = (a & (b | c))

Pa : (b=true or c=true
compactly: (b or c)
Pb : (a and !c)
Pc : (a and !b)

Write truth values for **b and c** that make clause **a** active

For example: Pa : b=?? or c=??

Write truth values for **a and c** that make clause **b** active

Write truth values for **a and b** that make clause **c** active

Calculating Determination

Finding values for minor clauses is easy for simple predicates, but hard for complex ones

Definitional approach:

$\boldsymbol{p_{c=\mathbf{true}}}$ is predicate ***p*** with every occurrence of clause ***c*** replaced by **true**

$\boldsymbol{p_{c=\mathbf{false}}}$ is predicate ***p*** with every occurrence of clause ***c*** replaced by false

To find values for minor clauses, *exclusive or* $\boldsymbol{p_{c=\mathbf{true}}}$ and $\boldsymbol{p_{c=\mathbf{false}}}$

$\boldsymbol{p_c} = \boldsymbol{p_{c=\mathbf{true}}} \oplus \boldsymbol{p_{c=\mathbf{false}}}$

After solving, $\boldsymbol{p_c}$ describes exactly the values needed for ***c*** to determine ***p***

Determination Example

Consider $p = a \vee b$

$p_a = p_{a=\text{true}} \oplus p_{a=\text{false}}$

$p_a = (\text{true} \vee b) \oplus (\text{false} \vee b)$ [identity law]

$p_a = \text{true} \oplus b$

$p_a = \neg b$

Thus a determines p when b=false.

Determination Example

Consider $p = a \wedge b$

$p_a = p_{a=\text{true}} \oplus p_{a=\text{false}}$

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

$p_a = b \oplus \text{false}$

$p_a = b$

Thus a determines p when $b = \text{true}$

Determination Example

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

$p_a = p_{a=\text{true}} \oplus p_{a=\text{false}}$

$p_a = (\text{true} \vee (b \wedge c)) \oplus (\text{false} \vee (b \wedge c))$ [identity law]

$p_a = \text{true} \oplus b \wedge c$

$p_a = \neg(b \wedge c)$ [De Morgan's Law]

$p_a = \neg b \vee \neg c$

Thus a determines p when b=false or c=false

Determination (tabular method)

A truth table can sometimes be easier than logic evaluation

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

Determination (tabular method)

A truth table can sometimes be easier than logic evaluation

Row	a	b	c	$a \lor (b \land c)$	p_a	p	
1	T	T	T	T			
2	T	T	F	T	✓		
3	T	F	T	T			
4	T	F	F	T			
5	F	T	T	T			
6	F	T	F	F	✓		
7	F	F	T	F			
8	F	F	F	F			

Select ***a*** as the major clause, then choose values for minor clauses ***b*** and ***c*** such that changing only ***a*** changes ***p***, thus ***a*** determines ***p***

Determination (tabular method)

A truth table can sometimes be easier than logic evaluation

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

Select ***a*** as the major clause, then choose values for minor clauses ***b*** and ***c*** such that changing only ***a*** changes ***p***, thus ***a*** determines ***p***

Determination (tabular method)

A truth table can sometimes be easier than logic evaluation

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

Select ***a*** as the major clause, then choose values for minor clauses ***b*** and ***c*** such that changing only ***a*** changes ***p***, thus ***a*** determines ***p***

Determination (tabular method)

A truth table can sometimes be easier than logic evaluation

Row	a	b	c	$a \lor (b \land c)$	p_a	p_b	
1	T	T	T	T			
2	T	T	F	T	✓		
3	T	F	T	T	✓		
4	T	F	F	T	✓		
5	F	T	T	T		✓	
6	F	T	F	F	✓		
7	F	F	T	F	✓	✓	
8	F	F	F	F	✓		

Select ***b*** as the major clause, then choose values for minor clauses ***a*** and ***c*** such that changing only ***b*** changes ***p***, thus ***b*** determines ***p***

Determination (tabular method)

A truth table can sometimes be easier than logic evaluation

Row	a	b	c	$a \lor (b \land c)$	p_a	p_b	p_c
1	T	T	T	T			
2	T	T	F	T	✓		
3	T	F	T	T	✓		
4	T	F	F	T	✓		
5	F	T	T	T		✓	
6	F	T	F	F	✓		
7	F	F	T	F	✓	✓	
8	F	F	F	F	✓		

Determination (tabular method)

A truth table can sometimes be easier than logic evaluation

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

> Select ***c*** as the major clause, then choose values for minor clauses ***a*** and ***b*** such that changing only ***c*** changes ***p***, thus ***c*** determines ***p***

Determination (tabular method)

A truth table can sometimes be easier than logic evaluation

Row	a	b	c	$a \lor (b \land c)$	p_a	p_b	
1	T	T	T	T			
2	T	T	F	T	✓(1)		
3	T	F	T	T	✓(2)		
4	T	F	F	T	✓(3)		
5	F	T	T	T		✓(4)	✓(5)
6	F	T	F	F	✓(1)		✓(5)
7	F	F	T	F	✓(2)	✓(4)	
8	F	F	F	F	✓(3)		

Instead of color-coding, we can tag the matching pairs with ID numbers

ACC Ambiguity

Do the minor clauses have to retain the same values while the major clause changes between true and false?

Example: given $\boldsymbol{p} = \boldsymbol{a} \vee (\boldsymbol{b} \wedge \boldsymbol{c})$, if a is the major clause then when we vary the major clause:

a=true, ***b***=false, ***c***=true

a=false, ***b***=false, ***c***=false

Is this allowed, or not?

Vary the major clause

This question has caused confusion among safety-critical testers for years

Resolving the ambiguity

Three possible answers (which leads to three different coverage criteria)

- Minor clauses *do not* need to be the same
- Minor clauses can *force the predicate* to become both true and false
- Minor clauses *do* need to be the same

General Active Clause Coverage

Minor clauses *do not* need to be the same

DEFINITION

General Active Clause Coverage (GACC) – For each p in P and each major clause c_i in C_p, choose minor clauses c_j (j!=i) such that c_i determines p. TR has two requirements for c_i: c_i evaluates to true and c_i evaluates to false. The values chosen for minor clauses c_j do not need to be the same when c_i is true as when c_i is false, and the value of p does not need to change.

It is possible to satisfy GACC without satisfying predicate coverage

Correlated Active Clause Coverage

Minor clauses can *force the predicate*

DEFINITION

Correlated Active Clause Coverage (CACC) – For each p in P and each major clause c_i in C_p, choose minor clauses c_j (j!=i) such that c_i determines p. TR has two requirements for c_i: c_i evaluates to true and c_i evaluates to false. The values chosen for minor clauses c_j must cause p to be true for one value of major clause c_i and false for the other value of c_i.

Subsumes predicate coverage

This is "masking MCDC"*

* DOT/FAA/AR-01/18 "An Investigation of Three Forms of the Modified Condition Decision Coverage (MCDC) Criterion", April 2001

Restrictive Active Clause Coverage

Minor clauses do need to be the same

DEFINITION

Restricted Active Clause Coverage (RACC) – For each p in P and each major clause c_i in C_p, choose minor clauses c_j (j!=i) such that c_i determines p. TR has two requirements for c_i: c_i evaluates to true and c_i evaluates to false. The values chosen for minor clauses c_j must must be the same when c_i is true as when c_i is false.

This is "unique-cause MCDC", the common interpretation of MCDC*

Often leads to infeasible test requirements

* DOT/FAA/AR-01/18 "An Investigation of Three Forms of the Modified Condition Decision Coverage (MCDC) Criterion", April 2001

Active Clause Comparison

To satisfy an active clause coverage criterion, pick a pair of tests where…

Criterion	Major clause determines p	Major clause changes value	Changing major clause changes p	Minor clauses are held the same
GACC	✓	✓		
CACC	✓	✓	✓	
RACC	✓	✓	✓	✓

GACC vs. CACC vs. RACC Example

Evaluation process

1. Select a major clause
2. Determine the conditions for the minor clauses where the major clause determines the predicate
3. For GACC, select a pair of conditions where the value of the major clause changes (the value of the minor clauses and *p* may or may not change)
4. For CACC, select a pair of conditions where the value of the major clause changes and the value of *p* changes (the minor clauses may or may not change)
5. For RACC, select a pair of conditions where the value of the major clause changes, the values of the minor clauses are held constant, and the value of p changes

GACC vs. CACC vs. RACC Example

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

1. Select a major clause -- *a*

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

GACC vs. CACC vs. RACC Example

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

2. Determine the conditions for the minor clauses where the major clause determines the predicate

Using the XOR approach:

$$p_a = p_{a=\text{true}} \oplus p_{a=\text{false}}$$
$$\mathrm{p}_a = (\mathrm{T} \wedge b) \vee (\neg \mathrm{T} \wedge \neg b \wedge c) \oplus (\mathrm{F} \wedge b) \vee (\neg \mathrm{F} \wedge \neg b \wedge c)$$
$$\mathrm{p}_a = b \vee (\mathrm{F} \wedge \neg b \wedge c) \oplus \mathrm{F} \vee (\mathrm{T} \wedge \neg b \wedge c)$$
$$\mathrm{p}_a = b \oplus \neg b \wedge c$$
$$p_a = b \vee c$$

GACC vs. CACC vs. RACC Example

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

2. Determine the conditions for the minor clauses where the major clause determines the predicate

Using the truth table approach:

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	
3	T	F	T	F	
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	
7	F	F	T	T	
8	F	F	F	F	

Select inputs such that ***a*** changes, ***b*** and ***c*** do not change, and ***p*** changes… thus ***a*** determines ***p*** when ***b*** $\wedge$ ***c***

GACC vs. CACC vs. RACC Example

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

2. Determine the conditions for the minor clauses where the major clause determines the predicate

Using the truth table approach:

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	
8	F	F	F	F	

Select inputs such that ***a*** changes, ***b*** and ***c*** do not change, and ***p*** changes… thus ***a*** determines ***p*** wh en ***b*** $\wedge \neg$***c***

GACC vs. CACC vs. RACC Example

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

2. Determine the conditions for the minor clauses where the major clause determines the predicate

Using the truth table approach:

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

Select inputs such that ***a*** changes, ***b*** and ***c*** do not change, and ***p*** changes… thus ***a*** determines ***p*** when $\neg b \wedge c$

GACC vs. CACC vs. RACC Example

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

2. Determine the conditions for the minor clauses where the major clause determines the predicate

Using the truth table approach:

	a	b	c	$p=(a\wedge b)\vee(\neg a\wedge\neg b\wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

Select inputs such that ***a*** changes, ***b*** and ***c*** do not change… but ***p*** DOES NOT change, thus ***a*** DOES NOT determines ***p*** when $\neg b \wedge \neg c$

GACC vs. CACC vs. RACC Example

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

2. Determine the conditions for the minor clauses where the major clause determines the predicate

Using the truth table approach:

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	√1
2	T	T	F	T	√2
3	T	F	T	F	√3
4	T	F	F	F	
5	F	T	T	F	√1
6	F	T	F	F	√2
7	F	F	T	T	√3
8	F	F	F	F	

$p_a = b \vee c$

GACC vs. CACC vs. RACC Example

To satisfy an active clause coverage criterion, pick a pair of tests where…

Criterion	Major clause determines p	Major clause changes value	Changing major clause changes p	Minor clauses are held the same
GACC	✓	✓		
CACC	✓	✓	✓	
RACC	✓	✓	✓	✓

GACC vs. CACC vs. RACC Example

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

3. For GACC, select a pair of conditions where the value of the major clause changes (the value of the minor clauses and *p* may or may not change)

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

GACC Pairs:
(1,5)

GACC vs. CACC vs. RACC Example

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

3. For GACC, select a pair of conditions where the value of the major clause changes (the value of the minor clauses and *p may or may not change*)

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

GACC Pairs: (1,5) or (1,6) or (1,7) okay because p does not need to change

GACC vs. CACC vs. RACC Example

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

3. For GACC, select a pair of conditions where the value of the major clause changes (the value of the minor clauses and *p* may or may not change)

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

GACC Pairs: any one of (1,5), (1,6), (1,7), (2,5), (2,6), (2,7), (3,5), (3,6), (3,7)

GACC vs. CACC vs. RACC Example

To satisfy an active clause coverage criterion, pick a pair of tests where…

Criterion	Major clause determines p	Major clause changes value	Changing major clause changes p	Minor clauses are held the same
GACC	✓	✓		
CACC	✓	✓	✓	
RACC	✓	✓	✓	✓

GACC vs. CACC vs. RACC Example

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

4. For CACC, select a pair of conditions where the value of the major clause changes and the value of *p* changes (the minor clauses may or may not change)

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

CACC Pairs: (1,5)

GACC vs. CACC vs. RACC Example

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

4. For CACC, select a pair of conditions where the value of the major clause changes and the value of *p* changes (the minor clauses may or may not change)

	a	b	c	p=(a∧b)∨(¬a∧¬b∧c)	p_a
1	T	T	T	T	√1
2	T	T	F	T	√2
3	T	F	T	F	√3
4	T	F	F	F	
5	F	T	T	F	√1
6	F	T	F	F	√2
7	F	F	T	T	√3
8	F	F	F	F	

CACC Pairs:
(1,5) or (1,6)

GACC vs. CACC vs. RACC Example

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

4. For CACC, select a pair of conditions where the value of the major clause changes and the value of *p* changes (the minor clauses may or may not change)

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

CACC Pairs: (1,5) or (1,6) ***but not (1,7) because p doesn't change!***

GACC vs. CACC vs. RACC Example

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

4. For CACC, select a pair of conditions where the value of the major clause changes and the value of p changes (the minor clauses may or may not change)

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

CACC Pairs: any one of (1,5), (1,6), (2,5), (2,6) *but not (2,7)*

GACC vs. CACC vs. RACC Example

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

4. For CACC, select a pair of conditions where the value of the major clause changes and the value of *p* changes (the minor clauses may or may not change)

	a	b	c	$p=(a \wedge b) \vee (\neg a \wedge \neg b \wedge c)$	p_a
1	T	T	T	T	√1
2	T	T	F	T	√2
3	T	F	T	F	√3
4	T	F	F	F	
5	F	T	T	F	√1
6	F	T	F	F	√2
7	F	F	T	T	√3
8	F	F	F	F	

CACC Pairs:
any one of
(1,5), (1,6),
(2,5), (2,6),
(3,7) ***but not***
(3,5) or (3,6)

GACC vs. CACC vs. RACC Example

To satisfy an active clause coverage criterion, pick a pair of tests where…

Criterion	Major clause determines p	Major clause changes value	Changing major clause changes p	Minor clauses are held the same
GACC	✓	✓		
CACC	✓	✓	✓	
RACC	✓	✓	✓	✓

GACC vs. CACC vs. RACC Example

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

5. For RACC, select a pair of conditions where the value of the major clause changes, the values of the minor clauses are held constant, and the value of p changes

	a	b	c	p=(a∧b)∨(¬a∧¬b∧c)	p_a
1	T	T	T	T	√1
2	T	T	F	T	√2
3	T	F	T	F	√3
4	T	F	F	F	
5	F	T	T	F	√1
6	F	T	F	F	√2
7	F	F	T	T	√3
8	F	F	F	F	

RACC Pairs: (1,5)

GACC vs. CACC vs. RACC Example

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

5. For RACC, select a pair of conditions where the value of the major clause changes, the values of the minor clauses are held constant, and the value of p changes

	a	b	c	$p=(a\wedge b)\vee(\neg a\wedge\neg b\wedge c)$	p_a
1	T	T	T	T	✓1
2	T	T	F	T	✓2
3	T	F	T	F	✓3
4	T	F	F	F	
5	F	T	T	F	✓1
6	F	T	F	F	✓2
7	F	F	T	T	✓3
8	F	F	F	F	

RACC Pairs:
(1,5) or (2,6)

GACC vs. CACC vs. RACC Example

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

5. For RACC, select a pair of conditions where the value of the major clause changes, the values of the minor clauses are held constant, and the value of p changes

	a	b	c	$p=(a\wedge b)\vee(\neg a\wedge\neg b\wedge c)$	p_a
1	T	T	T	T	√1
2	T	T	F	T	√2
3	T	F	T	F	√3
4	T	F	F	F	
5	F	T	T	F	√1
6	F	T	F	F	√2
7	F	F	T	T	√3
8	F	F	F	F	

RACC Pairs:
(1,5) or
(2,6) or
(3,7)

Extra Credit!

Making clauses active

P = ((a&b) | c | (d&e))

Pick any **one** of the 5 clauses (call it $\mathbf{c_i}$)

Solve for $\mathbf{c_i}$

Answer by giving truth values for the other 4 clauses that make your $\mathbf{c_i}$ determine the value of the predicate

Show your work!

Inactive Clause Coverage

Taking the opposite approach – major clauses do not affect the predicates

DEFINITION

Inactive Clause Coverage (ICC) – For each p in P and each major clause c_i in C_p, choose minor clauses c_j (j!=i) such that c_i *does not* determine p. *TR* has four requirements for c_i: (1) c_i evaluates to true with p true, (2) c_i evaluates to false with p true, (3) c_i evaluates to true with p false, and (4) c_i evaluates to false with p false.

Why bother? It's useful for testing safety interlock systems to ensure that during certain circumstances a control variable does not have any effect on operation

Inactive Clause Coverage

DEFINITION

General Inactive Clause Coverage (GICC) – For each p in P and each major clause c_i in C_p, choose minor clauses c_j (j!=i) such that c_i *does not* determine p. The values chosen for minor clauses c_j *do not* need to be the same when c_i is true as when c_i is false.

DEFINITION

Restricted Inactive Clause Coverage (RICC) – For each p in P and each major clause c_i in C_p, choose minor clauses c_j (j!=i) such that c_i *does not* determine p. The values chosen for minor clauses c_j *must be* the same when c_i is true as when c_i is false.

Infeasibility

Infeasible test requirements are common

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

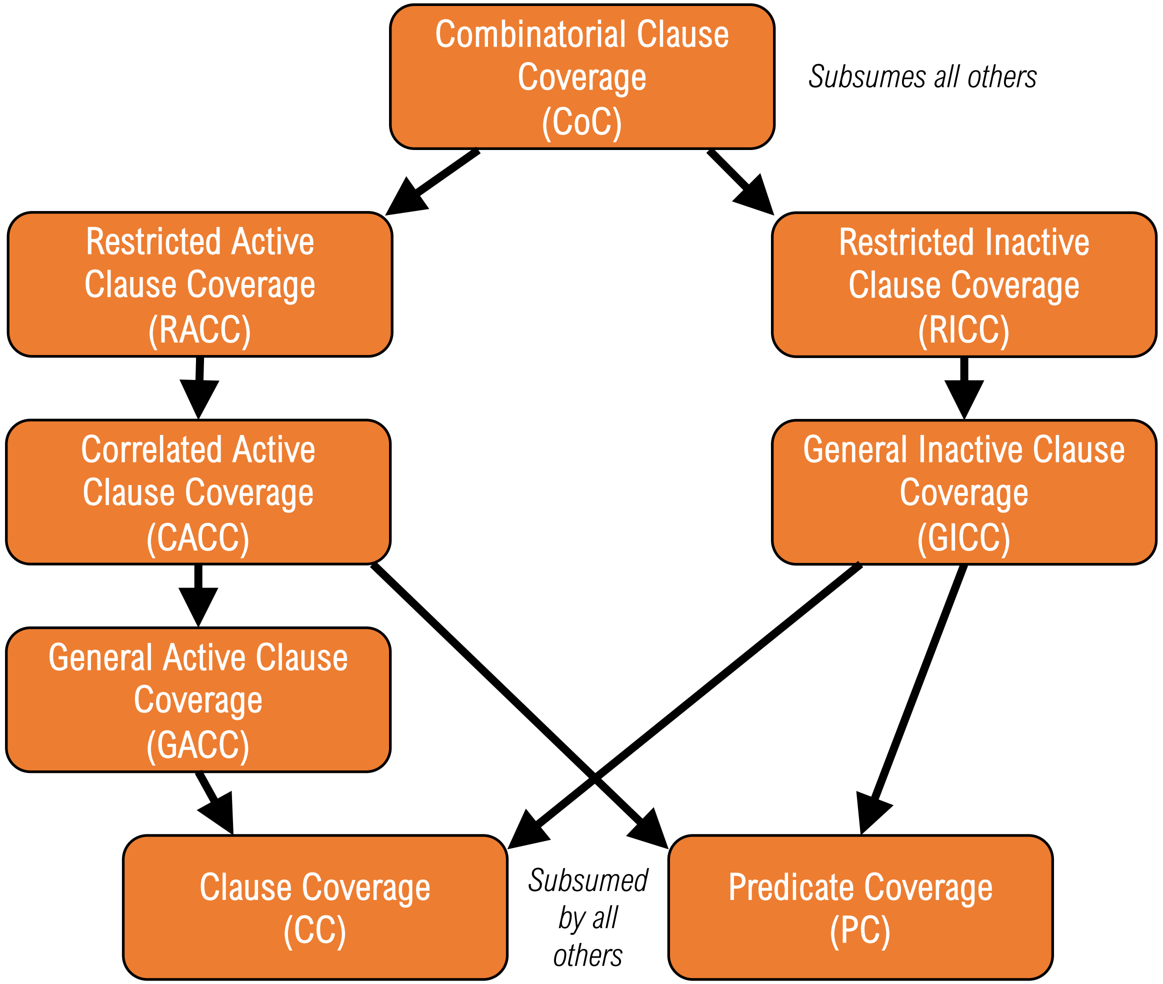
If $(a > b)$=**true** and $(b > c)$=**true**, then $(c > a)$=**true** is infeasible

As with ISP and graph criteria, infeasible test requirements are generally be *recognized and ignored*

Logic Criteria Subsumption

DEFINITION

A test criterion *C1* **subsumes** *C2* if and only if every set of test cases that satisfies criterion *C1* also satisfies *C2*



Logic Coverage Summary

Predicates are often very simple – in practice, most have fewer than 3 clauses

- That's good news, because fewer clauses *significantly* simplifies testing
- With only one clause, predicate coverage is sufficient
- With 2 or 3 clauses, combinatorial coverage may be practical
- With more complex clauses, ACC and ICC criteria are practical

Testing safety-critical software often requires MCDC (or RACC or CACC)