معیارهای پوشش منطقی

محمد تنهایی

پوشش معیارهای منطقی

عبارات منطقی در مکان‌های مختلفی دیده می‌شوند

پوشش عبارات منطقی یک نیازمندی برای انجام پروژه‌های هوایی مربوط به دولت فدرال امریکا می‌باشد.

عبارات منطقی می‌توانند از جاهای مختلفی نشات بگیرند:

Decisions in programs

FSMs and statecharts

Requirements

آزمون نرم‌افزاری زیرمجموعه‌ای از تمامی حالات جدول درستی را در نظر می‌گیرد و مورد آزمون قرار می‌دهد.

Logic Predicates and Clauses

predicate یک عبارت است که می‌تواند به یک مقدار **boolean** تبدیل شود.

Predicates می‌تواند شامل:

- **boolean variables**
- عبارات غیر boolean شامل: **>, <, ==, >=, <=, !=**
- **فراخوانی** تابع boolean

ساختار جملات منطقی توسط عملگرهای زیر درست می‌شود:

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

clause یک جمله است که هیچ عملگر منطقی‌ای در آن وجود ندارد.

مثال

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

چهار clause:

relational expression – (a < b)

f (z) – boolean-valued function

D – boolean variable

relational expression – (m >= n*o)

بیشتر جملات شامل Clauseهای محدودی هستند.

البته بهتر است که این مورد بررسی شود!

منابع جملات منطقی

Decisions in programs

Guards in finite state machines

Decisions in UML activity graphs

Requirements, both formal and informal

SQL queries

ترجمه جملات منطقی از زبان طبیعی به زبان ریاضی

“I am interested in SWE 637 and CS 652”

$course = swe637\ OR\ course = cs652$

“If you leave before 6:30 AM, take Braddock to 495, if you leave after 7:00 AM, take Prosperity to 50, then 50 to 495”

$time < 6{:}30 \rightarrow path = Braddock \vee time > 7{:}00 \rightarrow path = Prosperity$

Hmm ... this is incomplete !

$time < 6{:}30 \rightarrow path = Braddock \vee time >= 6{:}30 \rightarrow path = Prosperity$

ترجمه جملات منطقی از زبان طبیعی به زبان ریاضی

"I am interested in SWE 637 and CS 652"

course = swe637 OR course = cs652

Humans have trouble translating from English to Logic

"If you leave before 6:30 AM, take Braddock to 495, if you leave after 7:00 AM, take Prosperity to 50, then 50 to 495"

$time < 6{:}30 \rightarrow path = Braddock \vee time > 7{:}00 \rightarrow path = Prosperity$

Hmm … this is incomplete !

$time < 6{:}30 \rightarrow path = Braddock \vee time >= 6{:}30 \rightarrow path = Prosperity$

آزمون و پوشش مبتنی بر جملات

ما از جملات به صورت زیر در آزمون استفاده می‌کنیم:

ایجاد یک مدل از نرم‌افزار به صورت یک عبارات منطقی (که شامل Clauseها می‌باشد)

ایجاد مجموعه آزمون‌هایی به جهت پوشش این Clauseها

اختصارات:

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 and Clause Coverage

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

Predicate and Clause Coverage

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

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

Predicate and Clause Coverage

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

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 edges, this is equivalent to edge coverage

Predicate and Clause Coverage

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

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 edges, this is equivalent to edge coverage

PC does not evaluate all the clauses, so ...

Predicate and Clause Coverage

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

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 edges, this is equivalent to edge coverage
PC does not evaluate all the clauses, so ...

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

مثالی از پوشش جملات

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

predicate coverage

مثالی از پوشش جملات

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

predicate coverage

Predicate = true

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

= (5 < 10) ∨ true ∧ (1 >= 1*1)

= true ∨ true ∧ TRUE

= true

مثالی از پوشش جملات

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

predicate coverage

Predicate = true

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

$= (5 < 10) \lor true \land (1 >= 1*1)$

$= true \lor true \land TRUE$

$= true$

Predicate = false

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

$= (10 < 5) \lor false \land (1 >= 1*1)$

$= false \lor false \land TRUE$

$= false$

مثالی از پوشش Clause

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

Clause coverage

مثالی از پوشش Clause

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

Clause coverage

(a < b) = true	(a < b) = false
a = 5, b = 10	a = 10, b = 5

مثالی از پوشش Clause

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

Clause coverage

(a < b) = true	(a < b) = false	D = true	D = false
a = 5, b = 10	a = 10, b = 5	D = true	D = false

مثالی از پوشش Clause

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

Clause coverage

(a < b) = true	(a < b) = false	D = true	D = false
a = 5, b = 10	a = 10, b = 5	D = true	D = false

m >= n*o = true	m >= n*o = false
m = 1, n = 1, o = 1	m = 1, n = 2, o = 2

مثالی از پوشش Clause

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

Clause coverage

(a < b) = true	(a < b) = false	D = true	D = false
a = 5, b = 10	a = 10, b = 5	D = true	D = false

m >= n*o = true	m >= n*o = false
m = 1, n = 1, o = 1	m = 1, n = 2, o = 2

Two tests

مثالی از پوشش Clause

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

Clause coverage

(a < b) = true   (a < b) = false   D = true   D = false

a = 5, b = 10   a = 10, b = 5   D = true   D = false

m >= n*o = true   m >= n*o = false

m = 1, n = 1, o = 1   m = 1, n = 2, o = 2

Two tests

true cases

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

مثالی از پوشش Clause

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

Clause coverage

(a < b) = true	(a < b) = false	D = true	D = false
a = 5, b = 10	a = 10, b = 5	D = true	D = false

m >= n*o = true	m >= n*o = false
m = 1, n = 1, o = 1	m = 1, n = 2, o = 2

false cases

true cases

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

مشکلات PC و CC

PC تمامی حالات مختلف Clauseها را بررسی نمی‌کند.

CC همیشه در بردارنده PC نیست

می‌توان به شیوه‌ای CC را پوشش داد که تمامی جمله درست و غلط نشود. یا به عبارتی PC پوشانده نشود.

و این مورد قطعا چیزی که مد نظر ما هست، نیست!

ساده‌ترین راه حل بررسی تمامی حالات Clauseها است.

پوشش همه حالات یا CoC

CoC requires every possible combination

Sometimes called Multiple Condition Coverage

پوشش همه حالات یا CoC

- CoC requires every possible combination
- Sometimes called Multiple Condition Coverage

Combinatorial Coverage (CoC) : For each p in P, TR has test requirements for the clauses in Cp to evaluate to each possible combination of truth values.

پوشش همه حالات یا CoC

CoC requires every possible combination

Sometimes called Multiple Condition Coverage

Combinatorial Coverage (CoC) : For each p in P, TR has test requirements for the clauses in Cp to evaluate to each possible combination of truth values.

	a < b	D	m >= n*o	((a < b) ∨ D) ∧ (m >= n*o)
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

- این روش ساده، سرراست و منطقی است!

Combinatorial Coverage

- **این روش ساده، سرراست و منطقی است!**

- ولی گران!

Combinatorial Coverage

- **این روش ساده، سرراست و منطقی است!**

- ولی گران!
- 2^N tests, where ***N*** is the number of clauses
  - برای بیشتر از ۳ یا ۴ Clause شدنی نیست.

Combinatorial Coverage

- **این روش ساده، سرراست و منطقی است!**

- ولی گران!
- 2^N tests, where ***N*** is the number of clauses
  - برای بیشتر از ۳ یا ۴ Clause شدنی نیست.
- راه‌حل‌های زیادی برای این مساله وجود دارد.

Combinatorial Coverage

- **این روش ساده، سرراست و منطقی است!**

- ولی گران!
- *2^N* tests, where ***N*** is the number of clauses
  - برای بیشتر از ۳ یا ۴ Clause شدنی نیست.
- راه‌حل‌های زیادی برای این مساله وجود دارد.
- راه حل کلی خیلی ساده است:

Combinatorial Coverage

- **این روش ساده، سرراست و منطقی است!**

- ولی گران!
- *2^N* tests, where ***N*** is the number of clauses
  - برای بیشتر از ۳ یا ۴ Clause شدنی نیست.
- راه‌حل‌های زیادی برای این مساله وجود دارد.
- راه حل کلی خیلی ساده است:

هر Clause را مستقل از Clauseهای دیگر تست می‌کنیم.

Combinatorial Coverage

- **این روش ساده، سرراست و منطقی است!**

- ولی گران!
- 2^N tests, where N is the number of clauses
  - برای بیشتر از ۳ یا ۴ Clause شدنی نیست.
- راه‌حل‌های زیادی برای این مساله وجود دارد.
- راه حل کلی خیلی ساده است:

هر Clause را مستقل از Clauseهای دیگر تست می‌کنیم.

- Getting the details right is hard

Combinatorial Coverage

- **این روش ساده، سرراست و منطقی است!**

- ولی گران!
- *2*^{N} tests, where ***N*** is the number of clauses
  - برای بیشتر از ۳ یا ۴ Clause شدنی نیست.
- راه‌حل‌های زیادی برای این مساله وجود دارد.
- راه حل کلی خیلی ساده است:

هر Clause را مستقل از Clauseهای دیگر تست می‌کنیم.

- Getting the details right is hard
- What exactly does “independently” mean ?

Combinatorial Coverage

- **این روش ساده، سرراست و منطقی است!**

- ولی گران!
- 2^N tests, where ***N*** is the number of clauses
  – برای بیشتر از ۳ یا ۴ Clause شدنی نیست.
- راه‌حل‌های زیادی برای این مساله وجود دارد.
- راه حل کلی خیلی ساده است:

هر Clause را مستقل از Clauseهای دیگر تست می‌کنیم.

- Getting the details right is hard
- What exactly does “independently” mean ?
- The book presents this idea as “*making clauses active*” …

Active Clauses

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

Consider the first test for **clause coverage**, which caused each clause to be true:

$(5 < 10) \lor true \land (1 >= 1*1)$

Only the first clause ***counts*** !

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

This is considered to ***make the clause active***

Active Clauses

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

Consider the first test for **clause coverage**, which caused each clause to be true:

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

Only the first clause ***counts*** !

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

This is considered to ***make the clause active***

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_j are such that changing c_i changes the value of p

Determining Predicates

Determining Predicates

$P = A \underline{\vee} B$

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

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

if $A = false$, B determines p.

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.

Determining Predicates

$P = A \vee B$

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

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

if $A = false$, B determines p.

$P = A \wedge B$

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

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

if $A = true$, B determines p.

هدف : پیدا کردن آزمون برای هر Clause که سبب تصمیم‌گیری آن Clause برای جمله شود.

Determining Predicates

$P = A \vee B$

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

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

if $A = false$, B determines p.

$P = A \wedge B$

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

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

if $A = true$, B determines p.

هدف : پیدا کردن آزمون برای هر Clause که سبب تصمیم‌گیری آن Clause برای جمله شود.

این روش سبب چند معیار پوشانندگی مختلف با تفاوت‌های جزیی می‌شود که در ادامه بررسی می‌کنیم.

Active Clause Coverage

Active Clause Coverage

Active Clause Coverage (ACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci : ci evaluates to true and ci evaluates to false.

Active Clause Coverage

Active Clause Coverage (ACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci : ci evaluates to true and ci evaluates to false.

p = a v b

a = true, b = false

a = false, b = false

a = false, b = true

a = false, b = false

Active Clause Coverage

Active Clause Coverage (ACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci : ci evaluates to true and ci evaluates to false.

p = a v b

a = true, b = false

a = false, b = false

a is major clause

---

a = false, b = true

a = false, b = false

Active Clause Coverage

Active Clause Coverage (ACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci : ci evaluates to true and ci evaluates to false.

p = a v b

a = true, b = false

a = false, b = false

a = false, b = true

a = false, b = false

a is major clause

b is major clause

Active Clause Coverage

Active Clause Coverage (ACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci : ci evaluates to true and ci evaluates to false.

p = a v b

a = true, b = false

a = false, b = false

a = false, b = true

~~a = false, b = false~~

a is major clause

b is major clause

Duplicate

Active Clause Coverage

Active Clause Coverage (ACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, $j \,!= i$, so that ci determines p. TR has two requirements for each ci : ci evaluates to true and ci evaluates to false.

p = a v b

a = true, b = false

a = false, b = false

a = false, b = true

~~a = false, b = false~~

a is major clause

b is major clause

Duplicate

Ambiguity : Do the minor clauses have to have the **same values** when the major clause is true and false?

Resolving the Ambiguity

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

Major clause : a

a = true, b = false, c = true

a = false, b = false, c = false

Resolving the Ambiguity

$p = a \lor (b \land c)$

Major clause : a

a = true, b = false, c = true

a = false, b = false, c = false

Is this allowed ?

Resolving the Ambiguity

$p = a \lor (b \land c)$

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

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:

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

Resolving the Ambiguity

$p = a \lor (b \land c)$

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
- Minor clauses **do** need to be the same

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
- Minor clauses **do** need to be the same
- Minor clauses **force the predicate** to become both true and false

General Active Clause Coverage

General Active Clause Coverage

General Active Clause Coverage (GACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, $j \mathrel{!}= i$, so that ci determines p. TR has two requirements for each ci : ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj do not need to be the same when ci is true as when ci is false, that is, $cj(ci = true) = cj(ci = false)$ for all cj OR $cj(ci = true) \mathrel{!}= cj(ci = false)$ *for all* cj.

General Active Clause Coverage

General Active Clause Coverage (GACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, $j != i$, so that ci determines p. TR has two requirements for each ci : ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj do not need to be the same when ci is true as when ci is false, that is, $cj(ci = true) = cj(ci = false)$ for all ci OR $cj(ci = true) != cj(ci = false)$ *for all* cj.

This is complicated !

General Active Clause Coverage

General Active Clause Coverage (GACC) : For each p in P and each major clause c_i in C_p, choose minor clauses c_j, $j \neq i$, so that c_i determines p. TR has two requirements for each c_i : c_i evaluates to true and c_i evaluates to false. The values chosen for the minor clauses c_j do not need to be the same when c_i is true as when c_i is false, that is, $c_j(c_i = true) = c_j(c_i = false)$ for all c_j OR $c_j(c_i = true) \neq c_j(c_i = false)$ *for all* c_j.

This is complicated!

It is possible to satisfy GACC without satisfying predicate coverage

General Active Clause Coverage

General Active Clause Coverage (GACC) : For each *p* in *P* and each major clause *ci* in *Cp*, choose minor clauses *cj*, *j != i*, so that *ci* determines *p*. TR has two requirements for each *ci* : *ci* evaluates to true and *ci* evaluates to false. The values chosen for the minor clauses *cj* do not need to be the same when *ci* is true as when *ci* is false, that is, *cj(ci = true) = cj(ci = false)* for all *ci* OR *cj(ci = true) != cj(ci = false) for all cj*.

This is complicated !

It is possible to satisfy GACC without satisfying predicate coverage

We really want to cause predicates to be both true and false !

Restricted Active Clause Coverage

Restricted Active Clause Coverage

Restricted Active Clause Coverage (RACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, $j \mathrel{!}= i$, so that ci determines p. TR has two requirements for each ci: ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj must be the same when ci is true as when ci is false, that is, it is required that $cj(ci = true) = cj(ci = false)$ for all cj.

Restricted Active Clause Coverage

Restricted Active Clause Coverage (RACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, $j != i$, so that ci determines p. TR has two requirements for each ci: ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj must be the same when ci is true as when ci is false, that is, it is required that $cj(ci = true) = cj(ci = false)$ for all cj.

This has been a **common interpretation** by aviation developers

Restricted Active Clause Coverage

Restricted Active Clause Coverage (RACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci: ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj must be the same when ci is true as when ci is false, that is, it is required that $cj(ci = true) = cj(ci = false)$ for all cj.

This has been a **common interpretation** by aviation developers

RACC often leads to **infeasible test requirements**

Restricted Active Clause Coverage

Restricted Active Clause Coverage (RACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci: ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj must be the same when ci is true as when ci is false, that is, it is required that $cj(ci = true) = cj(ci = false)$ for all cj.

This has been a **common interpretation** by aviation developers

RACC often leads to **infeasible test requirements**

There is **no logical reason** for such a restriction

Correlated Active Clause Coverage

Correlated Active Clause Coverage

Correlated Active Clause Coverage (CACC)：For each p in P and each major clause ci in Cp, choose minor clauses cj, $j \mathrel{!=} i$, so that ci determines p. TR has two requirements for each ci: ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj must cause p to be true for one value of the major clause ci and false for the other, that is, it is required that $p(ci = true) \mathrel{!=} p(ci = false)$.

Correlated Active Clause Coverage

Correlated Active Clause Coverage (CACC)：For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci: ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj must cause p to be true for one value of the major clause ci and false for the other, that is, it is required that

$$p(ci = true) \; != p(ci = false).$$

A **more recent** interpretation

Correlated Active Clause Coverage

Correlated Active Clause Coverage (CACC)：For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci: ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj must cause p to be true for one value of the major clause ci and false for the other, that is, it is required that

$$p(ci = true) \mathrel{!=} p(ci = false).$$

A **more recent** interpretation

Implicitly allows minor clauses to have different values

Correlated Active Clause Coverage

Correlated Active Clause Coverage (CACC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, j != i, so that ci determines p. TR has two requirements for each ci: ci evaluates to true and ci evaluates to false. The values chosen for the minor clauses cj must cause p to be true for one value of the major clause ci and false for the other, that is, it is required that

$$p(ci = true) \,!\!= p(ci = false).$$

A **more recent** interpretation

Implicitly allows minor clauses to have different values

Explicitly satisfies (**subsumes**) predicate coverage

CACC and RACC

CACC and RACC

	a	b	c	a ∧ (b ∨ 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

CACC and RACC

	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

major clause

CACC and RACC

	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

major clause

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

CACC and RACC

	a	b	c	a ∧ (b ∨ 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

	a	b	c	a ∧ (b ∨ 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

major clause

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

CACC and RACC

	a	b	c	a ∧ (b ∨ 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

major clause

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
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

major clause

CACC and RACC

	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

major clause

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	$a \wedge (b \vee c)$
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

major clause

RACC can only be satisfied by one of the three pairs above

Inactive Clause Coverage

The 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

Inactive Clause Coverage

The 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

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

General and Restricted ICC

- Unlike ACC, the notion of correlation is not relevant
  - ci does not determine p, so cannot correlate with p
- Predicate coverage is always guaranteed

General and Restricted ICC

- Unlike ACC, the notion of correlation is not relevant
  - c_i does not determine p, so cannot correlate with p
- Predicate coverage is always guaranteed

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

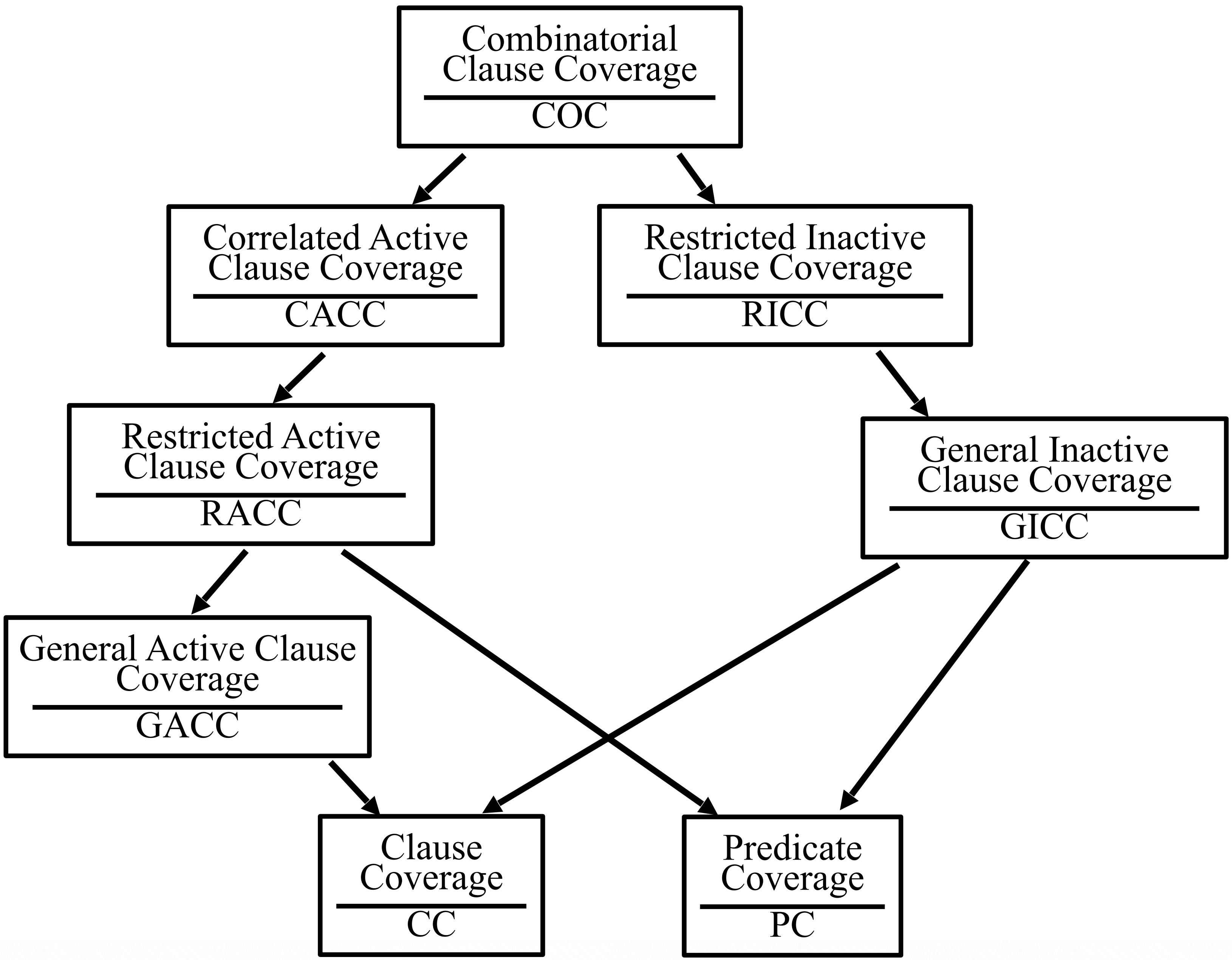
General and Restricted ICC

- Unlike ACC, the notion of correlation is not relevant
  - ci does not determine p, so cannot correlate with p
- Predicate coverage is always guaranteed

General Inactive Clause Coverage (GICC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, $j \mathrel{!}= i$, so that ci does not determine p. The values chosen for the minor clauses cj do not need to be the same when ci is true as when ci is false, that is, $cj(ci = true) = cj(ci = false)$ for all cj OR $cj(ci = true) \mathrel{!}= cj(ci = false)$ for all cj.

Restricted Inactive Clause Coverage (RICC) : For each p in P and each major clause ci in Cp, choose minor clauses cj, $j \mathrel{!}= i$, so that ci does not determine p. The values chosen for the minor clauses cj must be the same when ci is true as when ci is false, that is, it is required that $cj(ci = true) = cj(ci = false)$ for all cj.

Logic Coverage Criteria Subsumption



Making Clauses Determine a Predicate

Finding values for minor clauses c_j is easy for simple predicates

But how to find values for more complicated predicates ?

Definitional approach:

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

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

To find values for the minor clauses, connect $p_{c=true}$ and $p_{c=false}$ with exclusive OR

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

After solving, p_c describes exactly the values needed for c to determine p

مثال‌ها

مثالها

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

مثالها

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

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

$$p = a \vee b$$

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

$$= (true \vee b) \text{ XOR } (false \vee b)$$

مثالها

$$p = a \lor b$$

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

$$= (true \lor b) \text{ XOR } (false \lor b)$$

$$p = a \land b$$

مثالها

$$p = a \lor b$$

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

$$= (true \lor b) \text{ XOR } (false \lor b)$$

$$p = a \land b$$

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

مثالها

$p = a \vee b$

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

$= (true \vee b) \text{ XOR } (false \vee b)$

$p = a \wedge b$

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

$= (true \wedge b) \oplus (false \wedge b)$

مثال‌ها

$$p = a \vee b$$

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

$$= (true \vee b) \text{ XOR } (false \vee b)$$

$$p = a \wedge b$$

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

$$= (true \wedge b) \oplus (false \wedge b)$$

$$= b \oplus false$$

مثالها

$$p = a \lor b$$

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

$$= (true \lor b) \text{ XOR } (false \lor b)$$

$$p = a \land b$$

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

$$= (true \land b) \oplus (false \land b)$$

$$= b \oplus false$$

$$= b$$

مثال

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

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

$$= (true \vee b) \text{ XOR } (false \vee b)$$

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

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

$$= (true \wedge b) \oplus (false \wedge b)$$

$$= b \oplus false$$

$$= b$$

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

مثلا

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

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

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

مثال

$$p = a \lor b$$

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

$$= (true \lor b) \text{ XOR } (false \lor b)$$

$$p = a \land b$$

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

$$= (true \land b) \oplus (false \land b)$$

$$= b \oplus false$$

$$= b$$

$$p = a \lor (b \land c)$$

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

$$= (true \lor (b \land c)) \oplus (false \lor (b \land c))$$

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

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

$$p = a \lor (b \land c)$$
$$p_a = p_{a=true} \oplus p_{a=false}$$
$$= (true \lor (b \land c)) \oplus (false \lor (b \land c))$$
$$= true \oplus (b \land c)$$

مثال

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

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

$$p = a \lor (b \land c)$$
$$p_a = p_{a=true} \oplus p_{a=false}$$
$$= (true \lor (b \land c)) \oplus (false \lor (b \land c))$$
$$= true \oplus (b \land c)$$
$$= \neg (b \land c)$$

أمثلة

$$p = a \lor b$$

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

$$= (true \lor b) \text{ XOR } (false \lor b)$$

$$p = a \land b$$

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

$$= (true \land b) \oplus (false \land b)$$

$$= b \oplus false$$

$$= b$$

$$p = a \lor (b \land c)$$

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

$$= (true \lor (b \land c)) \oplus (false \lor (b \land c))$$

$$= true \oplus (b \land c)$$

$$= \neg (b \land c)$$

$$= \neg b \lor \neg c$$

مثلا

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

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

$$p = a \lor (b \land c)$$
$$p_a = p_{a=true} \oplus p_{a=false}$$
$$= (true \lor (b \land c)) \oplus (false \lor (b \land c))$$
$$= true \oplus (b \land c)$$
$$= \neg (b \land c)$$
$$= \neg b \lor \neg c$$

- "*NOT b* ∨ *NOT c*" means either *b* or *c* can be false

$$p = a \lor b$$

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

$$= (true \lor b) \text{ XOR } (false \lor b)$$

$$p = a \land b$$

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

$$= (true \land b) \oplus (false \land b)$$

$$= b \oplus false$$

$$= b$$

$$p = a \lor (b \land c)$$

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

$$= (true \lor (b \land c)) \oplus (false \lor (b \land c))$$

$$= true \oplus (b \land c)$$

$$= \neg (b \land c)$$

$$= \neg b \lor \neg c$$

- "*NOT b* ∨ *NOT c*" means either *b* or *c* can be false
- RACC requires the same choice for both values of *a*, CACC does not

مثال‌ها

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

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

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

- "*NOT b* ∨ *NOT c*" means either *b* or *c* can be false
- RACC requires the same choice for both values of *a*, CACC does not

مثالها

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

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

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

- "*NOT b* ∨ *NOT c*" means either *b* or *c* can be false
- RACC requires the same choice for both values of *a*, CACC does not

Repeated Variables

The definitions in this chapter yield the same tests no matter how the predicate is expressed

$(a \lor b) \land (c \lor b) == (a \land c) \lor b$

$(a \land b) \lor (b \land c) \lor (a \land c)$

Only has 8 possible tests, not 64

Use the simplest form of the predicate, and ignore contradictory truth table assignments

A More Subtle Example

A More Subtle Example

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

A More Subtle Example

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

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

A More Subtle Example

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

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

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

A More Subtle Example

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

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

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

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

A More Subtle Example

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

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

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

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

A More Subtle Example

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

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

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

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

A More Subtle Example

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

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

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

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

$$= (a \vee false) \oplus (false \vee a)$$

A More Subtle Example

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

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

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

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

$$= (a \vee false) \oplus (false \vee a)$$

- *a* always determines the value of this predicate

A More Subtle Example

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

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

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

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

$$= (a \vee false) \oplus (false \vee a)$$

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

A More Subtle Example

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

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

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

$$= (b \vee \neg b) \oplus false$$

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

$$= (a \vee false) \oplus (false \vee a)$$

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

A More Subtle Example

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

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

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

$$= (b \vee \neg b) \oplus false$$

$$= true \oplus false$$

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

$$= (a \vee false) \oplus (false \vee a)$$

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

A More Subtle Example

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

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

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

$$= (b \vee \neg b) \oplus false$$

$$= true \oplus false$$

$$= true$$

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

$$= (a \vee false) \oplus (false \vee a)$$

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

A More Subtle Example

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

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

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

$$= (b \vee \neg b) \oplus false$$

$$= true \oplus false$$

$$= true$$

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

$$= (a \vee false) \oplus (false \vee a)$$

$$= a \oplus a$$

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

A More Subtle Example

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

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

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

$$= (b \vee \neg b) \oplus false$$

$$= true \oplus false$$

$$= true$$

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

$$p_b = p_{b=true} \oplus p_{b=false}$$

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

$$= (a \vee false) \oplus (false \vee a)$$

$$= a \oplus a$$

$$= false$$

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

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

خلاصه پوشش منطقی

خلاصه پوشش منطقی

Predicates are often very simple

- PC may be enough
- CoC is practical

خلاصه پوشش منطقی

Predicates are often very simple

- PC may be enough
- CoC is practical

خلاصه پوشش منطقی

Predicates are often very simple

- PC may be enough
- CoC is practical

Control software often has many complicated predicates, with lots of clauses

خلاصه پوشش منطقی

Predicates are often very simple

- PC may be enough
- CoC is practical

Control software often has many complicated predicates, with lots of clauses

Question ... why don’t complexity metrics count the number of clauses in predicates?

پایان