University of **Sheffield**

COM3529 Software Testing and Analysis

White-Box Coverage Criteria based on
Logic Analysis

Dr José Miguel Rojas
slides by Prof Phil McMinn

Is Branch Coverage Enough?

Consider the following defect in a method that checks if a user is allowed to vote in the UK (voters must be British and 18+):

```
if (nationality != "BRITISH" || age < 21)
    throw new Exception("Cannot vote")
```

Branch coverage can be satisfied by testing only the sub-condition involving the nationality, i.e., the faulty sub-expression may never determine the outcome even if we test both outcomes of the branch.

Logic-based Coverage aims to test individual conditions in compound predicates.

```java
public static int daysBetweenTwoDates(int year1, int month1, int day1,
                                      int year2, int month2, int day2) {
    ...
    if ((year2 < year1) ||
            (year2 == year1 && month2 < month1) ||
            (year2 == year1 && month2 == month1 && day2 < day1)) {
        int t = month2;
        month2 = month1;
        month1 = t;
        t = day2;
        day2 = day1;
        day1 = t;
        t = year2;
        year2 = year1;
        year1 = t;
    }
    ...
```

```
public static int daysBetweenTwoDates(int year1, int month1, int day1,
                                       int year2, int month2, int day2) {
    ...
    if ((year2 < year1) ||
            (year2 == year1 && month2 < month1) ||
            (year2 == year1 && month2 == month1 && day2 < day1)) {
        int t = month2;
        month2 = month1;
        month1 = t;
        t = day2;
        day2 = day1;
        day1 = t;
        t = year2;
        year2 = year1;
        year1 = t;
    }
    ...
```

$$D_1 \lor D_2$$

```java
public static int daysBetweenTwoDates(int year1, int month1, int day1,
                                      int year2, int month2, int day2) {
    ...
    if ((year2 < year1) ||
            (year2 == year1 && month2 < month1) ||
            (year2 == year1 && month2 == month1 && day2 < day1)) {
        int t = month2;
        month2 = month1;
        month1 = t;
        t = day2;
        day2 = day1;
        day1 = t;
        t = year2;
        year2 = year1;
        year1 = t;
    }
    ...
```

Disjunct 1

Disjunct 2

Disjunct 3

Disjunct 1

Disjunct 2

Disjunct 3

```java
public static int daysBetweenTwoDates(int year1, int month1, int day1,
                                      int year2, int month2, int day2) {
    ...
    if ((year2 < year1) ||
        (year2 == year1 && month2 < month1) ||
        (year2 == year1 && month2 == month1 && day2 < day1)) {
        int t = month2;
        month2 = month1;
        month1 = t;
        t = day2;
        day2 = day1;
        day1 = t;
        t = year2;
        year2 = year1;
        year1 = t;
    }
    ...
```

Branch Coverage does not ensure each disjunct is exercised as true and false:

Test Case	Example Input						Disjunct			Branch Predicate
	year1	month1	day1	year2	month2	day2	1	2	3	
1	2019	12	13	2018	4	25	T	F	F	T
2	2018	4	25	2019	12	13	F	F	F	F

Analysing the Logic of a Predicate

Logical Operators

Operator	Math Symbol	Java
not	¬	!
and	∧	&&
or	∨	||

A condition is a boolean expression that is a component of a more complex predicate that does not contain any logical operators
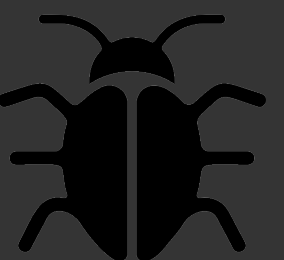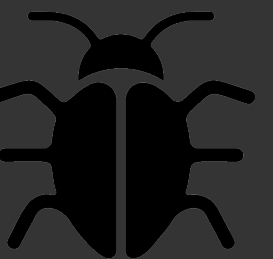
Analysing the Logic of a Predicate

Logical Operators

Operator	Math Symbol	Java
not	¬	!
and	∧	&&
or	∨	||

A condition is a boolean expression that is a component of a more complex predicate that does not contain any logical operators

```
if ((year2 < year1) ||
        (year2 == year1 && month2 < month1) ||
        (year2 == year1 && month2 == month1 && day2 < day1)) {
    ...
}
```

How many conditions are there in this if statement?

Analysing the Logic of a Predicate

```
if ((year2 < year1) ||
       (year2 == year1 && month2 < month1) ||
       (year2 == year1 && month2 == month1 && day2 < day1)) {
  ...
}
```

How many conditions are there in this `if` statement?

	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

Condition Coverage

Exercise each condition as **true** and **false**

```
if ((year2 < year1) ||
        (year2 == year1 && month2 < month1) ||
        (year2 == year1 && month2 == month1 && day2 < day1)) {
  ...
}
```

Condition		Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

Test Case	Condition					Branch Predicate	Example Input					
	1	2	3	4	5		year1	month1	day1	year2	month2	day2
1	T	F	F	T	T	T	2019	1	2	2018	1	1
2	F	T	T	F	F	T	2019	2	1	2019	1	1

Condition Coverage

Exercise each condition as **true** and **false**

```
if ((year2 < year1) ||
        (year2 == year1 && month2 < month1) ||
        (year2 == year1 && month2 == month1 && day2 < day1)) {
  ...
}
```

	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

Test Case	Condition					Branch Predicate	Example Input					
	1	2	3	4	5		year1	month1	day1	year2	month2	day2
1	T	F	F	T	T	T	2019	1	2	2018	1	1
2	F	T	T	F	F	T	2019	2	1	2019	1	1

Multiple Condition Coverage

Exercise each possible combination of truth
values for each condition

n conditions = 2^n test requirements

Multiple Condition Coverage

Exercise each possible combination of truth
values for each condition

n conditions = 2^n test requirements

	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

Multiple Condition Coverage

Exercise each possible combination of truth
values for each condition

n conditions = 2^n test requirements

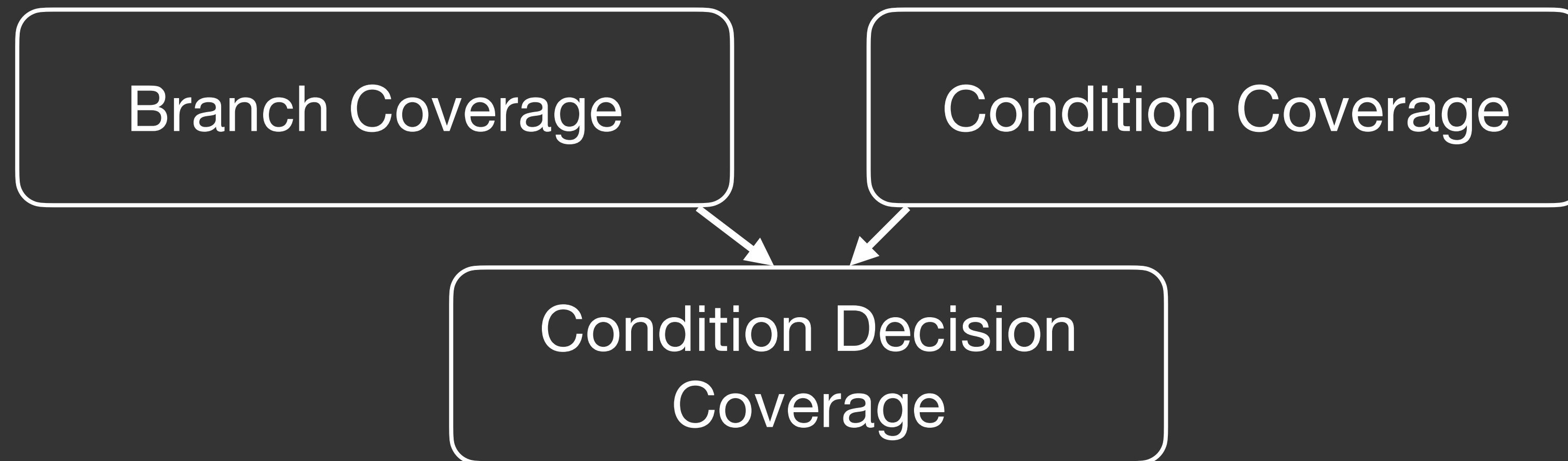|	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

5 unique conditions =
32 test requirements

But not all combinations
are **feasible**

Condition Decision Coverage

Exercise each condition as true and false, as well as the overall branch predicate

Branch Coverage

Condition Coverage

Condition Decision Coverage

Modified Condition Decision Coverage (MCDC)

Exercise each condition as true and false. When the condition flips truth value, the branch predicate must also flip

Modified Condition Decision Coverage (MCDC)

Exercise each condition as true and false. When the condition flips truth value, the branch predicate must also flip
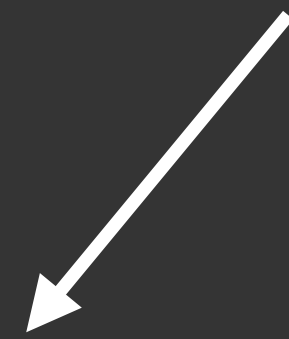
The "major" condition

Truth values of the remaining "minor" conditions set so that flipping the major condition flips the predicate

The major condition will then determine the predicate

	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

Restricted MCDC

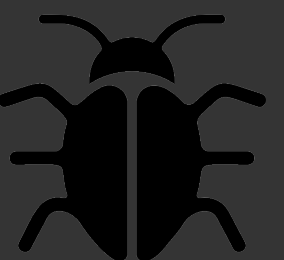The truth values of the minor conditions are fixed as the major condition and the predicate flip

Condition 1 is major:

Test Case	Condition					Branch Predicate	Example Input					
	1	2	3	4	5		year1	month1	day1	year2	month2	day2
1	T	F	F	F	F	T	2019	1	1	2018	2	1
2	F	F	F	F	F	F	2018	1	1	2019	2	1

Condition 2 is major:

Test Case	Condition					Branch Predicate	Example Input					
	1	2	3	4	5		year1	month1	day1	year2	month2	day2
3	F	T	T	F	F	T	2019	2	1	2019	1	1
4	F	F	T	F	F	F	2018	2	1	2019	1	1

	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

Restricted MCDC

The truth values of the minor conditions are fixed as the major condition and the predicate flip
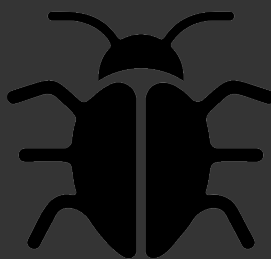
Condition 1 is major:

Test Case	Condition 1	2	3	4	5	Branch Predicate	year1	month1	day1	year2	month2	day2
1	T	F	F	F	F	T	2019	1	1	2018	2	1
2	F	F	F	F	F	F	2018	1	1	2019	2	1

Condition 2 is major:

Test Case	Condition 1	2	3	4	5	Branch Predicate	year1	month1	day1	year2	month2	day2
3	F	T	T	F	F	T	2019	2	1	2019	1	1
4	F	F	T	F	F	F	2018	2	1	2019	1	1

	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

Restricted MCDC

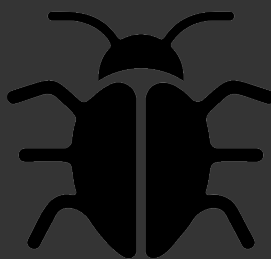The truth values of the minor conditions are fixed as the major condition and the predicate flip

Condition 1 is major:

Test Case	Condition 1	2	3	4	5	Branch Predicate	year1	month1	day1	year2	month2	day2
1	T	F	F	F	F	T	2019	1	1	2018	2	1
2	F	F	F	F	F	F	2018	1	1	2019	2	1

Condition 2 is major:

Test Case	Condition 1	2	3	4	5	Branch Predicate	year1	month1	day1	year2	month2	day2
3	F	T	T	F	F	T	2019	2	1	2019	1	1
4	F	F	T	F	F	F	2018	2	1	2019	1	1

	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

Correlated MCDC

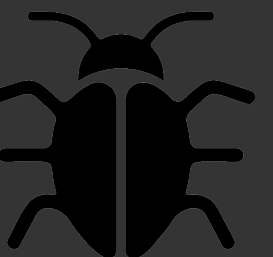The truth values of the minor conditions do not need to be fixed as the major condition and the predicate flip

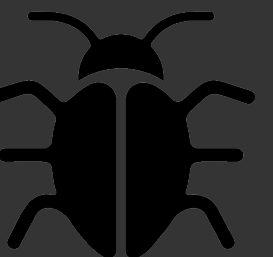| Test Case	Condition					Branch Predicate	Example Input					
	1	2	3	4	5		year1	month1	day1	year2	month2	day2
1	T	F	F	T	T	T	2019	1	2	2018	1	1
2	F	T	T	F	F	T	2019	2	1	2019	1	1
3	F	F	F	F	F	F	2018	1	1	2019	2	1

Correlated MCDC
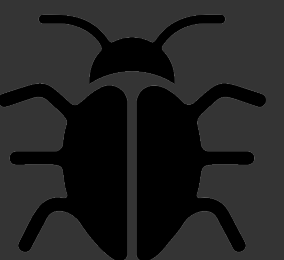
The truth values of the minor conditions do not need to
be fixed as the major condition and the predicate flip

Test Case	Condition					Branch Predicate	Example Input					
	1	**2**	**3**	**4**	**5**		year1	month1	day1	year2	month2	day2
1	T	F	F	T	T	T	2019	1	2	2018	1	1
2	F	T	T	F	F	T	2019	2	1	2019	1	1
3	F	F	F	F	F	F	2018	1	1	2019	2	1

Correlated MCDC

The truth values of the minor conditions do not need to
be fixed as the major condition and the predicate flip

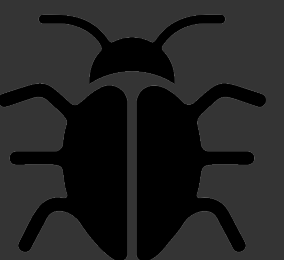| Test Case	Condition					Branch Predicate	Example Input					
	1	2	3	4	5		year1	month1	day1	year2	month2	day2
1	T	F	F	T	T	T	2019	1	2	2018	1	1
2	F	T	T	F	F	T	2019	2	1	2019	1	1
3	F	F	F	F	F	F	2018	1	1	2019	2	1

Correlated MCDC

The truth values of the minor conditions do not need to be fixed as the major condition and the predicate flip

Test Case	Condition 1	2	3	4	5	Branch Predicate	year1	month1	day1	year2	month2	day2
1	T	F	F	T	T	T	2019	1	2	2018	1	1
2	F	T	T	F	F	T	2019	2	1	2019	1	1
3	F	F	F	F	F	F	2018	1	1	2019	2	1

	Condition	Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

```
if ((year2 < year1) ||
        (year2 == year1 && month2 < month1) ||
        (year2 == year1 && month2 == month1 && day2 < day1)) {
   ...
}
```

Disjunct 3 not actually exercised as true

Correlated MCDC

The truth values of the minor conditions do not need to
be fixed as the major condition and the predicate flip

Test Case	Condition					Branch Predicate	Example Input					
	1	2	3	4	5		year1	month1	day1	year2	month2	day2
1	T	F	F	T	T	T	2019	1	2	2018	1	1
2	F	T	T	F	F	T	2019	2	1	2019	1	1
3	F	F	F	F	F	F	2018	1	1	2019	2	1

Condition		Disjunct
1	year2 < year1	1
2	year2 == year1	2
3	month2 < month1	2
=2	year2 == year1	3
4	month2 == month1	3
5	day2 < day1	3

```
if ((year2 < year1) ||
        (year2 == year1 && month2 < month1) ||
        (year2 == year1 && month2 == month1 && day2 < day1)) {
    ...
}
```

Disjunct 3 not actually exercised as true

Some General Observations

Some General Observations

- MCDC variants offer the best trade-off in terms of number of test requirements v. test thoroughness

Some General Observations

- MCDC variants offer the best trade-off in terms of number of test requirements v. test thoroughness

- Correlated MCDC tends to result in smaller test suites than Restricted

Some General Observations

- MCDC variants offer the best trade-off in terms of number of test requirements v. test thoroughness

- Correlated MCDC tends to result in smaller test suites than Restricted

- Multiple Condition Coverage yields too many test requirements

Some General Observations

- MCDC variants offer the best trade-off in terms of number of test requirements v. test thoroughness

- Correlated MCDC tends to result in smaller test suites than Restricted

- Multiple Condition Coverage yields too many test requirements

- Condition Coverage does not subsume Branch Coverage

Consider this…

```
...
boolean yearSame = year2 == year1;
boolean yearAndMonthSame = yearSame && month2 == month1;

boolean secondDateBeforeFirstByYear = year2 < year1;
boolean secondDateBeforeFirstByMonth = yearSame && month2 < month1;
boolean secondDateBeforeFirstByDay = yearAndMonthSame && day2 < day1;

boolean secondDateBefore = secondDateBeforeFirstByYear ||
        secondDateBeforeFirstByMonth || secondDateBeforeFirstByDay;

if (secondDateBeforeFirst) {
    ...
}
...
```

MCDC and Testing Standards

MCDC and Testing Standards

- ISO 26262 – "Road Vehicles – Functional Safety"
  https://en.wikipedia.org/wiki/ISO_26262

MCDC and Testing Standards

- **ISO 26262** – "Road Vehicles – Functional Safety"
https://en.wikipedia.org/wiki/ISO_26262

- **EN 50128** – a functional safety standard used in the rail industry
https://www.adacore.com/industries/rail/en50128

MCDC and Testing Standards

- ISO 26262 – "Road Vehicles – Functional Safety"
  https://en.wikipedia.org/wiki/ISO_26262

- EN 50128 – a functional safety standard used in the rail industry
  https://www.adacore.com/industries/rail/en50128

- DO-178B and DO-178C – "Software Considerations in Airborne Systems and Equipment Certification"
  https://en.wikipedia.org/wiki/DO-178B
  https://en.wikipedia.org/wiki/DO-178C

MCDC and Testing Standards

- ISO 26262 – "Road Vehicles – Functional Safety"
  https://en.wikipedia.org/wiki/ISO_26262

- EN 50128 – a functional safety standard used in the rail industry
  https://www.adacore.com/industries/rail/en50128

- DO-178B and DO-178C – "Software Considerations in Airborne Systems and Equipment Certification"
  https://en.wikipedia.org/wiki/DO-178B
  https://en.wikipedia.org/wiki/DO-178C

- IEC 61508, a basic functional safety standard applicable to all industries.
  https://en.wikipedia.org/wiki/IEC_61508