Solution to Homework 3

Does criterion $C_1$ subsume $C_2$? That is, if a test set gives 100% coverage with respect to $C_1$, will it also give 100% coverage with respect to $C_2$?

<table>
<thead>
<tr>
<th>block</th>
<th>statement</th>
<th>condition</th>
<th>decision</th>
<th>CD</th>
<th>MC</th>
<th>c-use</th>
<th>p-use</th>
<th>all-use</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>Yes (1)</td>
<td>No (2)</td>
<td>Yes (3)</td>
<td>Yes (4)</td>
<td>Yes (5)</td>
<td>No (6)</td>
<td>Yes (7)</td>
<td>Yes (8)</td>
</tr>
<tr>
<td>statement</td>
<td>Yes (1)</td>
<td>No (2)</td>
<td>Yes (3)</td>
<td>Yes (4)</td>
<td>Yes (5)</td>
<td>No (6)</td>
<td>Yes (7)</td>
<td>Yes (8)</td>
</tr>
<tr>
<td>condition</td>
<td>No (2)</td>
<td>No (2)</td>
<td>No (9)</td>
<td>Yes (10)</td>
<td>Yes (11)</td>
<td>No (12)</td>
<td>No (13)</td>
<td>No (14)</td>
</tr>
<tr>
<td>decision</td>
<td>No (3)</td>
<td>No (3)</td>
<td>No (9)</td>
<td>Yes (15)</td>
<td>Yes (16)</td>
<td>No (17)</td>
<td>Yes (18)</td>
<td>Yes (19)</td>
</tr>
<tr>
<td>CD</td>
<td>No (4)</td>
<td>No (4)</td>
<td>No (10)</td>
<td>No (15)</td>
<td>Yes (20)</td>
<td>No (21)</td>
<td>No (22)</td>
<td>No (23)</td>
</tr>
<tr>
<td>MC</td>
<td>No (5)</td>
<td>No (5)</td>
<td>No (11)</td>
<td>No (16)</td>
<td>No (20)</td>
<td>No (24)</td>
<td>No (25)</td>
<td>No (26)</td>
</tr>
<tr>
<td>c-use</td>
<td>No (6)</td>
<td>No (6)</td>
<td>No (12)</td>
<td>No (17)</td>
<td>No (21)</td>
<td>No (24)</td>
<td>No (27)</td>
<td>Yes (28)</td>
</tr>
<tr>
<td>p-use</td>
<td>No (7)</td>
<td>No (7)</td>
<td>No (13)</td>
<td>No (18)</td>
<td>No (22)</td>
<td>No (25)</td>
<td>No (27)</td>
<td>Yes (29)</td>
</tr>
<tr>
<td>all-use</td>
<td>No (8)</td>
<td>No (8)</td>
<td>No (14)</td>
<td>No (19)</td>
<td>No (23)</td>
<td>No (26)</td>
<td>No (28)</td>
<td>No (29)</td>
</tr>
</tbody>
</table>

1. block and statement
   - 100% block coverage gives 100% statement coverage? \(\Rightarrow \) Yes
     
     For each statement, it must belong to a block, so covering every block is equivalent to covering every statement.

   - 100% statement coverage gives 100% block coverage? \(\Rightarrow \) Yes
     
     For each block, it contains at least one statement, so covering every statement is equivalent to covering every block.

2. block/statement and condition
   - 100% block/statement coverage gives 100% condition coverage? \(\Rightarrow \) No
     
     \(s_1: \) if \((a > 0 \& \& b > 0)\) {
     \(s_2: \) \(a++;\)
     \(s_3: \) } else {
     \(s_4: \) \(b++;\)
     \(s_5: \) }

     \(t_1: \{a = 1, b = 1\}\) and \(t_2: \{a = 1, b = -1\}\) gives 100% block/statement coverage. To achieve 100% condition coverage, test cases need to make \(a > 0\) as true and false as well as \(b > 0\) as true and false. However, \(t_1\) and \(t_2\) fail to make \(a > 0\) as false so that the 100% condition coverage is not satisfied.

   - 100% condition coverage gives 100% block/statement coverage? \(\Rightarrow \) No
     
     \(s_1: \) if \((a > 0 \| b > 0)\) {
     \(s_2: \) \(a++;\)
     \(s_3: \) } else {
     \(s_4: \) \(b++;\)
     \(s_5: \) }
\( t_1: \{ a = -1, b = 1 \} \) and \( t_2: \{ a = 1, b = -1 \} \) gives 100\% condition coverage (both \( a > 0 \) and \( b > 0 \) are evaluated as true and false). However, \( s_3 \) and \( s_4 \) are not covered since both test cases will make \( (a > 0 \lor b > 0) \) as true.

3. block/statement and decision

- 100\% block/statement coverage gives 100\% decision coverage? \( \rightarrow \text{No} \)

\[
\begin{align*}
s_1: & \quad \text{if} \ (a > 0 \ \&\& \ b > 0) \\
s_2: & \quad a++; \\
s_3: & \quad b++; \\
s_4: & \quad \}
\end{align*}
\]

\( t_1: \{ a = 1, b = 1 \} \) gives 100\% block/statement coverage. However, the false branch is not taken so that the 100\% decision coverage is not satisfied.

- 100\% decision coverage gives 100\% block/statement coverage? \( \rightarrow \text{Yes} \)

Every block/statement must be part of a branch; or it will be covered by any test cases. So covering both true and false branches of all decisions makes every block/statement to be covered.

4. block/Statement and condition-decision

- 100\% block/statement coverage gives 100\% condition-decision coverage? \( \rightarrow \text{No} \)

Since 100\% block/statement coverage cannot guarantee 100\% condition or decision coverage, it cannot give 100\% condition-decision coverage (which subsumes condition coverage and decision coverage).

- 100\% condition-decision coverage gives 100\% block/statement coverage? \( \rightarrow \text{Yes} \)

Since 100\% decision coverage gives 100\% block/statement coverage, 100\% condition-decision (which subsumes condition coverage and decision coverage) also gives 100\% block/statement coverage.

5. block/Statement and multiple-condition

- 100\% block/statement coverage gives 100\% multiple-condition coverage? \( \rightarrow \text{No} \)

\[
\begin{align*}
s_1: & \quad \text{if} \ (a > 0 \ \&\& \ b > 0) \\
s_2: & \quad a++; \\
s_3: & \quad \} \text{else}\{} \\
s_4: & \quad b++; \\
s_5: & \quad \}
\end{align*}
\]

100\% multiple-condition coverage requires test cases which make \( \{ a > 0 \text{ as true, } b > 0 \text{ as true} \} \), \( \{ a > 0 \text{ as true, } b > 0 \text{ as false} \} \), \( \{ a > 0 \text{ as false, } b > 0 \text{ as true} \} \), and \( \{ a > 0 \text{ as false, } b > 0 \text{ as false} \} \). \( t_1: \{ a = 1, b = 1 \} \) and \( t_2: \{ a =1, b = -1 \} \) gives 100\% block/statement coverage. However, only \( \{ a > 0 \text{ as true, } b > 0 \text{ as true} \} \) and \( \{ a > 0 \text{ as true, } b > 0 \text{ as false} \} \) are covered.
• 100% multiple-condition coverage gives 100% block/statement coverage? \(\Rightarrow\) **Yes**

If 100% multiple-condition coverage is achieved, it means all program paths have been covered. Hence, all block/statement will also be covered.

6. **block/statement and c-use**

• 100% block/statement coverage gives 100% c-use coverage? \(\Rightarrow\) **No**

```plaintext
s_1: \quad \text{if} \ (a > 0) \ \{
\quad s_2: \quad x = 1
\quad s_3: \quad \} \ \text{else} \{
\quad s_4: \quad x = 2;
\quad s_5: \quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
\quad \}
• 100% p-use coverage gives 100% block/statement coverage? → Yes

\[\begin{array}{l}
s_1: \quad \text{input} (x, y); \\
s_2: \quad \text{if} (x > 0) \\
s_3: \quad x++; \\
s_4: \quad \\
s_5: \quad \text{print}(x); \\
s_6: \quad \text{if} (y > 0) \\
s_7: \quad y++; \\
s_8: \quad \\
s_9: \quad \text{print}(y); \\
\end{array}\]

100% p-use coverage guarantees that define/p-use pairs, (s_1, (s_2, s_3)) and (s_1, (s_2, s_5)), of variable \(x\) are covered and define/p-use pairs, (s_1, (s_6, s_7)) and (s_1, (s_6, s_9)), of variable \(y\) are covered. It implies that all program branches are covered. Hence, all block/statements are also covered.

8. block/statement and all-use

• 100% block/statement coverage gives 100% all-use coverage? → No

Since 100% block/statement coverage does not give 100% c-use coverage or 100% p-use coverage, it cannot give 100% all-use coverage (which subsumes c-use and p-use coverage).

• 100% all-use coverage gives 100% block/statement coverage? → Yes

Since 100% p-use coverage give 100% block/statement coverage, 100% all-use coverage also gives 100% block/statement coverage.

9. condition and decision

• 100% condition coverage gives 100% decision coverage? → No

\[\begin{array}{l}
s_1: \quad \text{if} (a > 0 \lor b > 0) \\
s_2: \quad a++; \\
s_3: \quad \} \text{else} \\
s_4: \quad b++; \\
s_5: \quad \\
\end{array}\]

t_1: \{a = -1, b = 1\} and t_2: \{a = 1, b = -1\} gives 100% condition coverage (both \(a > 0\) and \(b > 0\) are evaluated as true and false). However, these test cases cannot make \((a > 0 \lor b > 0)\) as false so that 100% decision coverage cannot be achieved.

• 100% decision coverage gives 100% condition coverage? → No

\[\begin{array}{l}
s_1: \quad \text{if} (a > 0 \lor b > 0) \\
s_2: \quad a++; \\
s_3: \quad } \text{else} \\
s_4: \quad b++; \\
s_5: \quad \\
\end{array}\]
$t_1: \{a = -1, b = 1\}, \, t_2: \{a = -1, b = -1\}$ gives 100% decision coverage ($a > 0 \, \| \, b > 0$ is evaluated as true and false). However, they cannot make condition $a > 0$ as true so that 100% condition coverage cannot be achieved.

10. condition and condition-decision

- 100% condition coverage gives 100% condition-decision coverage? → **No**

Since 100% condition cannot give 100% decision coverage, it also cannot give 100% condition-decision coverage (which subsumes condition and decision coverage).

- 100% condition-decision coverage gives 100% condition coverage? → **Yes**

Condition-decision coverage subsumes condition coverage and decision coverage.

11. condition and multiple-condition

- 100% condition coverage gives 100% multiple-condition coverage? → **No**

```
s_1: \text{if} \ (a > 0 \, \| \, b > 0) \{
    s_2: \ a++; 
    s_3: \ )\text{else}\{
    s_4: \ b++; 
    s_5: \ }
```

100% Multiple-condition coverage requires test cases which cover {\(a > 0\) as true, \(b > 0\) as true}, {\(a > 0\) as true, \(b > 0\) as false}, {\(a > 0\) as false, \(b > 0\) as true}, and {\(a > 0\) as false, \(b > 0\) as false}. $t_1: \{a = -1, b = 1\}$ and $t_2: \{a = 1, b = -1\}$ gives 100% condition coverage (both $a > 0$ and $b > 0$ are evaluated as true and false). However, they cannot cover {\(a > 0\) as true, \(b > 0\) as true} and {\(a > 0\) as false, \(b > 0\) as false}.

- 100% multiple-condition coverage gives 100% condition coverage? → **Yes**

```
s_1: \text{if} \ (a > 0 \, \| \, b > 0) \{
    s_2: \ a++; 
    s_3: \ )\text{else}\{
    s_4: \ b++; 
    s_5: \ }
```

100% Multiple-condition coverage requires test cases which cover {\(a > 0\) as true, \(b > 0\) as true}, {\(a > 0\) as true, \(b > 0\) as false}, {\(a > 0\) as false, \(b > 0\) as true}, and {\(a > 0\) as false, \(b > 0\) as false}. If 100% multiple-condition coverage is satisfied then each condition must be evaluated as true and false.
12. condition and c-use

- 100% condition coverage gives 100% c-use coverage? \(\rightarrow \text{No}\)

\[ \begin{align*}
\text{s}_1: &\quad \text{input}(a, b); \ x = 0; \ y = 0; \\
\text{s}_2: &\quad \text{if}(a > 0 \text{ and } b > 0)\
\text{s}_3: &\quad a = x + 1; \\
\text{s}_4: &\quad \text{else} \\
\text{s}_5: &\quad b = y + 1; \\
\text{s}_6: &\quad \}
\end{align*} \]

The define/c-use pair of variable \(x\) contains \((s_1, s_3)\). The define/c-use pair of variable \(y\) contains \((s_1, s_5)\).

\(t_1: \{a = -1, b = 1\}\) and \(t_2: \{a = 1, b = -1\}\) gives 100% condition coverage. However, the define/c-use pair of \(y\) is not covered.

- 100% c-use coverage gives 100% condition coverage? \(\rightarrow \text{No}\)

\[ \begin{align*}
\text{s}_1: &\quad \text{input}(a, b); \\
\text{s}_2: &\quad \text{if}(a > 0)\
\text{s}_3: &\quad \text{print}(a + b); \\
\text{s}_4: &\quad \text{else} \\
\text{s}_5: &\quad \text{print}(\text{"Hello World!"}); \\
\text{s}_6: &\quad \}
\end{align*} \]

\(s_5\) does not contain c-use of any variable. \(t_1: \{a = 1, b = 1\}\) gives 100% c-use coverage. However, \(t_1\) cannot make \(a > 0\) as false.

13. condition and p-use

- 100% condition coverage gives 100% p-use coverage? \(\rightarrow \text{No}\)

\[ \begin{align*}
\text{s}_1: &\quad \text{input}(a, b); \\
\text{s}_2: &\quad \text{if}(a > 0 \text{ and } b > 0)\
\text{s}_3: &\quad x++; \\
\text{s}_4: &\quad \text{else} \\
\text{s}_5: &\quad y++; \\
\text{s}_6: &\quad \}
\end{align*} \]

100% p-use coverage requires define/p-use pairs, \((s_1, (s_2, s_3))\) and \((s_1, (s_2, s_5))\), of variable \(a\) and define/p-use pairs, \((s_1, (s_2, s_3))\) and \((s_1, (s_2, s_3))\), of variable \(b\) to be covered.

\(t_1: \{a = -1, b = 1\}\), \(t_2: \{a = 1, b = -1\}\) gives 100% condition coverage. However, only \((s_1, (s_2, s_3))\) of \(a\) and \((s_1, (s_2, s_3))\) of \(b\) are covered.

- 100% p-use coverage gives 100% condition coverage? \(\rightarrow \text{No}\)

\[ \begin{align*}
\text{s}_1: &\quad \text{input}(a, b); \\
\text{s}_2: &\quad \text{if}(a > 0 \text{ and } b > 0)\
\text{s}_3: &\quad x++; \\
\text{s}_4: &\quad \text{else} \\
\end{align*} \]
100% p-use coverage requires define/p-use pairs, \((s_1, (s_2, s_3))\) and \((s_1, (s_2, s_5))\), of variable \(a\) and define/p-use pairs, \((s_1, (s_2, s_3))\) and \((s_1, (s_2, s_5))\), of variable \(b\) to be covered.

\(t_1: \{a = -1, b = 1\}, t_2: \{a = -1, b = -1\}\) gives 100% p-use coverage. However, \(a > 0\) cannot be true using these test cases.

14. condition and all-use

- 100% condition coverage gives 100% all-use coverage? \(\Rightarrow \text{No}\)

Since 100% condition cannot give 100% p-use or 100% c-use coverage, it also cannot give 100% all-use coverage (which subsumes c-use and p-use coverage).

- 100% all-use coverage gives 100% condition coverage? \(\Rightarrow \text{No}\)

100% p-use coverage requires define/p-use pairs, \((s_1, (s_2, s_3))\) and \((s_1, (s_2, s_5))\), of variable \(a\) and define/p-use pairs, \((s_1, (s_2, s_3))\) and \((s_1, (s_2, s_5))\), of variable \(b\) to be covered.

The define/c-use pair of variable \(x\) contains \((s_1, s_3)\). The define/c-use pair of variable \(y\) contains \((s_1, s_5)\).

\(t_1: \{a = -1, b = 1\}, t_2: \{a = -1, b = -1\}\) gives 100% all-use (100% c-use and 100% p-use) coverage. However, \(a > 0\) cannot be true using these test cases.

15. decision and condition-decision

- 100% decision coverage gives 100% condition-decision coverage? \(\Rightarrow \text{No}\)

Since 100% decision cannot give 100% condition coverage, it also cannot give 100% condition-decision coverage (which subsumes condition and decision coverage).

- 100% condition-decision coverage gives 100% condition coverage? \(\Rightarrow \text{Yes}\)

Condition-decision coverage subsumes condition coverage and decision coverage.

16. decision and multiple-condition

- 100% decision coverage gives 100% multiple-condition coverage? \(\Rightarrow \text{No}\)

\(s_1: \quad \text{if } (a > 0 \parallel b > 0)\{
\quad a++;\)

\(s_2: \quad y++;\)

\(s_6: \quad \}

100% Multiple-condition coverage requires test cases which cover \{a > 0 as true, b > 0 as true\}, \{a > 0 as true, b > 0 as false\}, \{a > 0 as false, b > 0 as true\}, and \{a > 0 as false, b > 0 as false\}.

t: \{a = -1, b = 1\}, t_2: \{a = -1, b = -1\} gives 100% decision coverage. However, they cannot cover \{a > 0 as true, b > 0 as true\} and \{a > 0 as true, b > 0 as false\}.

- 100% multiple-condition coverage gives 100% decision coverage? → Yes

17. decision and c-use

- 100% decision coverage gives 100% c-use coverage? → No

The define/c-use pair of variable x contains (s_2, s_7), (s_2, s_9), (s_4, s_7), (s_4, s_9).

t_1: \{a = -1, b = 1\}, t_2: \{a = 1, b = -1\} gives 100% decision coverage. However, only two pairs are covered (s_2, s_9), (s_4, s_7).

- 100% c-use coverage gives 100% decision coverage? → No
does not contain c-use of any variable. \( t_1: \{ a = 1, b = 1 \} \) gives 100% c-use coverage. However, \( t_1 \) cannot make \( a > 0 \) as false.

18. decision and p-use

- 100% decision coverage gives 100% p-use coverage? \( \rightarrow \text{No} \)

\[
\begin{align*}
s_1: & \quad \text{input } (a); \\
s_2: & \quad \text{if } (a > 2) \{ \\
s_3: & \quad x = a \times 2; \\
s_4: & \quad \text{else} \\
s_5: & \quad x = a + 6; \\
s_6: & \quad \} \\
s_7: & \quad \text{if } (x > 7) \{ \\
s_8: & \quad y = x + 1; \\
s_9: & \quad \text{else} \\
s_{10}: & \quad y = x - 1; \\
s_{11}: & \quad \}
\end{align*}
\]

100% p-use coverage requires define/p-use pairs, \((s_3, (s_7, s_8)), (s_3, (s_7, s_{10})), (s_5, (s_7, s_8)), \) and \((s_5, (s_7, s_{10}))\) of variable \( x \) are covered.

\( t_1: \{ a = 4 \}, t_2: \{ a = 1 \} \) gives 100% decision coverage.

\( t_1 \) covers \((s_3, (s_7, s_8))\) of \( x \).

\( t_2 \) covers \((s_5, (s_7, s_{10}))\) of \( x \).

\((s_3, (s_7, s_{10}))\) and \((s_5, (s_7, s_8))\) are not covered.

- 100% p-use coverage gives 100% decision coverage? \( \rightarrow \text{Yes} \)

\[
\begin{align*}
s_1: & \quad \text{input } (a); \\
s_2: & \quad \text{if } (a > 2) \{ \\
s_3: & \quad x = a \times 2; \\
s_4: & \quad \text{else} \\
s_5: & \quad x = a + 6; \\
s_6: & \quad \} \\
s_7: & \quad \text{if } (x > 7) \{ \\
s_8: & \quad y = x + 1; \\
s_9: & \quad \text{else} \\
s_{10}: & \quad y = x - 1; \\
s_{11}: & \quad \}
\end{align*}
\]

100% p-use coverage requires define/p-use pairs, \((s_3, (s_7, s_8)), (s_3, (s_7, s_{10})), (s_5, (s_7, s_8)), \) and \((s_5, (s_7, s_{10}))\) of variable \( x \) are covered.

\( t_1: \{ a = 4 \}, t_2: \{ a = 1 \}, t_3: \{ a = 3 \}, t_4: \{ a = 2 \} \) gives 100% p-use coverage. They also guarantee 100% decision coverage.

Generally speaking, for p-use coverage, every p-use must independently contribute to the decision taking both a true and false branch. So it guarantees decision coverage.

19. decision and all-use

- 100% decision coverage gives 100% all-use coverage? \( \rightarrow \text{No} \)
Since 100% decision coverage cannot give 100% p-use or 100% c-use coverage, it also cannot give 100% all-use coverage (which subsumes c-use and p-use coverage).

- 100% all-use coverage gives 100% decision coverage? → Yes

Since 100% p-use coverage gives 100% decision coverage, 100% all-use coverage (which subsumes p-use coverage) also gives 100% decision coverage.

20. condition-decision and multiple-condition

- 100% condition-decision coverage gives 100% multiple-condition coverage? → No

```plaintext
s_1: if (a > 0 || b > 0){
    s_2: a++;
    s_3: }else{
    s_4: b++;
    s_5: }
```

100% Multiple-condition coverage requires test cases which cover \{a > 0 as true, b > 0 as true\}, \{a > 0 as true, b > 0 as false\}, \{a > 0 as false, b > 0 as true\}, and \{a > 0 as false, b > 0 as false\}. 
\( t_1: \{a = -1, b = 1\}, t_2: \{a = -1, b = -1\}, t_3: \{a = 1, b = -1\} \) gives 100% condition-decision coverage. However, they cannot cover \{a > 0 as true, b > 0 as false\}.

- 100% multiple-condition coverage gives 100% condition-decision coverage? → Yes

```plaintext
s_1: if (a > 0 || b > 0){
    s_2: a++;
    s_3: }else{
    s_4: b++;
    s_5: }
```

100% Multiple-condition coverage requires test cases which cover \{a > 0 as true, b > 0 as true\}, \{a > 0 as true, b > 0 as false\}, \{a > 0 as false, b > 0 as true\}, and \{a > 0 as false, b > 0 as false\}. If 100% multiple-condition coverage is satisfied, it implies all possible combinations of condition outcomes are covered so that 100% condition-decision coverage is achieved.

21. condition-decision and c-use

- 100% condition-decision coverage gives 100% c-use coverage? → No

```plaintext
s_1: if (a > 0) {
    s_2: x = 1;
    s_3: }else{
    s_4: x = 2;
    s_5: }
    s_6: if (b > 0) {
    s_7: y = x + 1;
    s_8: }else{
    s_9: y = x - 1;
```
The define/c-use pair of variable x contains (s₂, s₇), (s₂, s₉), (s₄, s₇), (s₄, s₉).

$t_1$: \{a = -1, b = 1\}, $t_2$: \{a = 1, b = -1\} gives 100% condition-decision coverage. However, only two pairs are covered (s₂, s₉), (s₄, s₇).

- 100% c-use coverage gives 100% condition-decision coverage? \(\Rightarrow\) No

$s₁$: input (a, b);
$s₂$: if (a > 0) {
$s₃$: print (a + b);
$s₄$: } else {
$s₅$: print (“Hello World!”);
$s₆$: }

$s₁$ does not contain c-use of any variable. $t₁$: \{a = 1, b = 1\} gives 100% c-use coverage. However, $t₁$ cannot make $a > 0$ as false.

22. condition-decision and p-use

- 100% condition-decision coverage gives 100% p-use coverage? \(\Rightarrow\) No

$s₁$: input (a);
$s₂$: if (a > 2) {
$s₃$: x = a * 2;
$s₄$: } else {
$s₅$: x = a + 6;
$s₆$: }
$s₇$: if (x > 7) {
$s₈$: y = x + 1;
$s₉$: } else {
$s₁₀$: y = x - 1;
$s₁₁$: }

100% p-use coverage requires define/p-use pairs, (s₃, (s₇, s₈)), (s₃, (s₇, s₁₀)), (s₅, (s₇, s₈)), and (s₅, (s₇, s₁₀)) of variable x are covered.
$t₁$: \{a = 4\}, $t₂$: \{a = 1\} gives 100% condition-decision coverage.
$t₁$ covers (s₃, (s₇, s₈)) of x.
$t₂$ covers (s₅, (s₇, s₁₀)) of x.
(s₃, (s₇, s₁₀)) and (s₅, (s₇, s₈)) are not covered.

- 100% p-use coverage gives 100% condition-decision coverage? \(\Rightarrow\) No

$s₁$: input (a, b);
$s₂$: if (a > 0 || b > 0) {
$s₃$: x ++;
$s₄$: } else {
$s₅$: y ++;
$s₆$: }
100% p-use coverage requires define/p-use pairs, \((s_1, (s_2, s_3))\) and \((s_1, (s_2, s_5))\), of variable \(a\) and define/p-use pairs, \((s_1, (s_2, s_3))\) and \((s_1, (s_2, s_5))\), of variable \(b\) are covered. 

\(t_1: \{a = -1, b = 1\}\), \(t_2: \{a = -1, b = -1\}\) gives 100% p-use coverage. However, \(a > 0\) cannot be true using these test cases. So 100% condition-decision coverage is not satisfied.

23. condition-decision and all-use

- 100% condition-decision coverage gives 100% all-use coverage? \(\rightarrow\) **No**

  Since 100% condition-decision coverage cannot give 100% p-use or 100% c-use coverage, it also cannot give 100% all-use coverage (which subsumes c-use and p-use coverage).

- 100% all-use coverage gives 100% condition-decision coverage? \(\rightarrow\) **No**

  Since 100% all-use coverage cannot give 100% condition coverage, Hence 100% all-use coverage also cannot give 100% condition-decision coverage (which subsumes decision and condition coverage).

24. multiple-condition and c-use

- 100% multiple-condition coverage gives 100% c-use coverage? \(\rightarrow\) **No**

  \(s_1: \) if \((a > 0)\) {
  \(s_2: \) \(x = 1\);
  \(s_3: \) }else{
  \(s_4: \) \(x = 2;\)
  \(s_5: \) }
  \(s_6: \) if \((b > 0)\) {
  \(s_7: \) \(y = x + 1;\)
  \(s_8: \) }else{
  \(s_9: \) \(y = x - 1;\)
  \(s_{10}: \) }

  The define/c-use pair of variable \(x\) contains \((s_2, s_7), (s_2, s_9), (s_4, s_7), (s_4, s_9)\).
  \(t_1: \{a = -1, b = 1\}, t_2: \{a = 1, b = -1\}\) gives 100% multiple-condition coverage. However, only two pairs are covered \((s_2, s_9), (s_4, s_7)\).

- 100% c-use coverage gives 100% multiple-condition coverage? \(\rightarrow\) **No**

  \(s_1: \) input \((a, b)\);
  \(s_2: \) if \((a > 0)\) {
  \(s_3: \) print \((a + b)\);
  \(s_4: \) }else{
  \(s_5: \) print (“Hello World!”);
  \(s_6: \) }

  \(s_5\) does not contain c-use of any variable. \(t_1: \{a = 1, b = 1\}\) gives 100% c-use coverage. However, \(t_1\) cannot make \(a > 0\) as false.
25. multiple-condition and p-use

- 100% multiple-condition coverage gives 100% p-use coverage? → No

```java
s1: input (a);
s2: if (a > 2) {
    s3: x = a * 2;
s4: } else {
    s5: x = a + 6;
}s6: }
s7: if (x > 7) {
    s8: y = x + 1;
s9: } else {
    s10: y = x - 1;
s11: }
```

100% p-use coverage requires define/p-use pairs, (s3, (s7, s8)), (s3, (s7, s10)), (s5, (s7, s8)), and (s5, (s7, s10)) of variable x are covered.

- 100% p-use coverage requires define/p-use pairs, (s1, (s2, s3)) and (s1, (s2, s5)), of variable a and define/p-use pairs, (s1, (s2, s3)) and (s1, (s2, s5)), of variable b are covered.

```java
s1: input (a, b);
s2: if (a > 0 || b > 0) {
    s3: x++;  
s4: } else {
    s5: y++;  
s6: }
```

26. multiple-condition and all-use

- 100% multiple-condition coverage gives 100% all-use coverage? → No

Since 100% multiple-condition coverage cannot give 100% p-use or 100% c-use coverage, it also cannot give 100% all-use coverage (which subsumes c-use and p-use coverage).

- 100% all-use coverage gives 100% multiple-condition coverage? → No

Since 100% all-use coverage cannot give 100% condition coverage, Hence 100% all-use coverage also cannot give 100% multiple-condition coverage.
27. c-use and p-use

- 100% c-use coverage gives 100% p-use coverage?  → No

\[s_1: \text{input} (a, b);
\]
\[s_2: \text{if} (a > 0) \{
\]
\[s_3: \text{print} (a + b);
\]
\[s_4: \} \text{else} \{
\]
\[s_5: \text{print} ("Hello World!");
\]
\[s_6: \}
\]

\[s_5\] does not contain c-use of any variable. \(t_1: \{a = 1, b = 1\}\) gives 100% c-use coverage. However, define/p-use pair of variable \(a, (s_1, (s_2, s_5))\), is not covered.

- 100% p-use coverage gives 100% c-use coverage?  → No

\[s_1: \text{input} (a, b);
\]
\[s_2: \text{if} (a > 0) \{
\]
\[s_3: x = 1
\]
\[s_4: \} \text{else} \{
\]
\[s_5: x = 2;
\]
\[s_6: \}
\]
\[s_7: \text{if} (b > 0) \{
\]
\[s_8: y = x + 1;
\]
\[s_9: \} \text{else} \{
\]
\[s_{10}: y = x - 1;
\]
\[s_{11}: \}
\]

The define/c-use pair of variable \(x\) contains \((s_3, s_9), (s_3, s_{10}), (s_5, s_8), (s_5, s_{10})\).
The define/p-use pair of variable \(a\) contains \((s_1, (s_2, s_3)), (s_1, (s_2, s_5))\)
The define/p-use pair of variable \(b\) contains \((s_1, (s_7, s_8)), (s_1, (s_7, s_{10}))\)
\(t_1: \{a = -1, b = 1\}, t_2: \{a = 1, b = -1\}\) gives 100% p-use coverage. However, two define/c-use pairs of \(x\) are not covered \((s_3, s_8), (s_5, s_{10})\).

28. c-use and all-use

- 100% c-use coverage gives 100% all-use coverage?  → No

100% c-use does not give 100% p-use (which is subsumed by all-use).

- 100% all-use coverage gives 100% c-use coverage?  → Yes

All-use subsumes c-use.

29. p-use and all-use

- 100% c-use coverage gives 100% all-use coverage?  → No

100% p-use does not give 100% c-use (which is subsumed by all-use).
• 100% all-use coverage gives 100% c-use coverage? \( \Rightarrow \text{Yes} \)

All-use subsumes p-use.