Data Flow Coverage

Today's Reading Assignment


- Background reading
**New reading assignment**


- **Background**

**Control-Flow-Graph-Based Coverage Criteria**

- Statement Coverage
- Branch Coverage
- Hidden Paths
- Path Coverage
- Loop Guidelines
  - General
  - Boundary - Interior
Paths for Boundary Interior Example

Boundary paths
1,2,3,5,7       a
1,2,3,6,7       b
1,2,4,5,7       c
1,2,4,6,7       d

Interior paths
(for 2 executions of the loop)
  a,a
  a,b
  a,c
  a,d
  b,a
  b,b
  ...
  x,y for x,y = a, b, c, d

Need Control Flow AND Data Dependence

x := ...
y := x
z := ...

1 2 3 4 5 6 7 8
**Simple Non-looping Example**

\[ x := 1 \]

\[ 2 \rightarrow 3 \]

\[ 4 \]

\[ 5 \rightarrow 6 := x \]

\[ 7 \]

All branches 1, 2, 4, 5, 7
1, 3, 4, 6, 7

does not exercise the relationship between the definition of \( X \) in statement 2 and the reference to \( X \) in statement 6.

**Definitions**

- \( d_n(x) \) denotes that variable \( x \) is assigned a value at node \( n \) (*defined*)
- \( u_m(y) \) denotes that variable \( y \) is used (referenced at node \( m \))
  - a definition clear path \( p \) with respect to (wrt) \( x \) is a subpath where \( x \) is not defined at any of the nodes in \( p \)
  - a definition \( d_m(x) \) reaches a use \( u_n(x) \) iff there is a subpath \( (m) \cdot p \cdot (n) \) such that \( p \) is definition clear wrt \( x \)
Data Flow Path Selection

- Rapps and Weyuker
  - definition-clear subpaths from definitions to uses

- Ntafos
  - chains of alternating definitions and uses linked by definition-clear subpaths

- Laski and Korel
  - combinations of definitions that reach uses at a node via a subpath

Assumptions: every control flow graph is well formed

- single start and single final node
- no edges of the form \((n, n_s)\) or \((n_f, n)\)
- no edges of the form \((n, n)\)
- there is at most one edge \((m, n)\) for all \(m, n\)
- graph is connected
  - Can syntactically reach all nodes from the start node
- every loop has a single entry and a single exit
**More assumptions**

- at least one variable is associated with a node representing a predicate
- no variable definitions are associated with a node representing a predicate
- every definition of a variable reaches at least one use of that variable
- every use is reached by at least one definition
- every control graph contains at least one variable definition
- no variable uses or definitions are associated with $n_s$ and $n_f$

---

**Rapps' and Weyuker's Data Flow Criteria**

*Foundation:*
- Definition-clear subpaths from each definition to {some|all} use(s)

*All-Defs*
- Some definition-clear subpath from each definition to some use reached by that definition

![Diagram](attachment:diagram.png)
**Rapps’ and Weyuker’s Data Flow Criteria**

**All-Uses**
- Some definition-clear subpath from each definition to each use reached by that definition and each successor node of the use.

![Diagram showing definition-clear paths]

**C-use** is a “computation use”

**P-use** is a “predicate use”

**All-C-Uses, Some-P-Uses**
- either All-C-Uses for $d_m(x)$ or at least one P-Use

**All-P-Uses, Some-C-Uses**
- either All-P-Uses for $d_m(x)$ or at least one C-Use
Rapps’ and Weyuker’s Data Flow Criteria

All-Du-Paths

- All definition-clear subpaths that are cycle-free or simple-cycles from each definition to each use reached by that definition and each successor node of the use

\[ x := x \]

Difference between All-Uses and All-DU-paths

\[ x := x \]

\[ := x \]
Difference between All-Uses and All-DU-paths

\[ x := X \]

…

def-clear

…

def-clear

…

def-clear

\[ := X \]

\[ := X \]


cycle-free or simple-cycles

def-clear

…

cycle-free or simple-cycles

def-clear

…

cycle-free or simple-cycles

def-clear

…

cycle-free or simple-cycles

def-clear

…

cycle-free or simple-cycles

def-clear

…

cycle-free or simple-cycles

def-clear

…

cycle-free or simple-cycles

def-clear
**Example**

![Diagram](image)

**All-Defs**

Requires:
- $d_0(x)$ to a use

Satisfactory Path:
- $0, 1, 2, 4, 6$
**All-Uses**

Requires:
- \(a: d_0(x) \text{ to } u_2(x)\)
- \(b: d_0(x) \text{ to } u_3(x)\)
- \(c: d_0(x) \text{ to } u_5(x)\)

Satisfactory Paths:
- \(0, 1, 2, 4, 5, 6\) (satisfies \(a, c\))
- \(0, 1, 3, 4, 6\) (satisfies \(b\))

**All-Du-Paths**

Requires:
- \(a: d_0(x) \text{ to } u_2(x)\)
- \(b: d_0(x) \text{ to } u_3(x)\)
- \(c: d_0(x) \text{ to } u_5(x)\) [T path]
- \(d: d_0(x) \text{ to } u_5(x)\) [F path]

Satisfactory Paths:
- \(0, 1, 2, 4, 5, 6\) (satisfies \(a, c\))
- \(0, 1, 3, 4, 5, 6\) (satisfies \(b, d\))
- \(0, 1, 2, 4, 6\)
  (satisfies \((4, 6)\) branch coverage)
All-Du-Paths

Requires:
- a: \( d_0(x) \) to \( u_2(x) \)
- b: \( d_0(x) \) to \( u_5(x) \)
- c: \( d_3(x) \) to \( u_5(x) \)
- d: \( d_3(x) \) to \( u_2(x) \)

Satisfactory Paths:
- 0,1,2,4,5,6 (satisfies a,b)
- 0,1,3,4,5,6 (satisfies c)
- 0,1,3,4,1,2,4,6 (satisfies d)

Ntafos' Data Flow Criteria

• Foundation:
  - Chains of alternating definitions and uses linked by definition-clear subpaths (k-dr interactions)
  - \( i^{th} \) definition reaches \( i^{th} \) use,
  - which defines \( i^{th}+1 \) definition
  - \( K \) is number of branches
**k-dr interactions**

1-dr

```plaintext
x := def-clear : x
```

2-dr

```plaintext
x := def-clear : y := ..x.. def-clear := ..y..
```

**Ntafos’ Data Flow Criteria**

- **Required K-tuples**
  
  Some subpath propagating each k-dr interaction
  
  + if last use is a predicate, both branches
  
  + if first definition or last use is in a loop, minimal and some larger number of loop iterations
1-DR interaction

From 1-DR to 2-DR

PATHS
0, 1, 2, 4, 5, 6 (satisfies a-d, j)
0, 1, 2, 3, 5, 6 (satisfies e-h)
0, 1, 2, 3, 5, 2, 4, 5, 2, 3, 5, 6
(satisfies i, k, l)
2-DR interactions

\begin{itemize}
\item $a_j$: $d_1(x), u_4(x), d_4(y), u_6(y)$
\item $a_k$: $d_1(x), u_4(x), d_4(y), u_2(y)$
\item $a_l$: $d_1(x), u_4(x), d_4(y), u_3(y)$
\item $e_g$: $d_0(y), u_3(y), d_3(x), u_5(x)$
\item $e_h$: $d_0(y), u_3(y), d_3(x), u_6(x)$
\item $e_i$: $d_0(y), u_3(y), d_3(x), u_4(x)$
\item $i_j$: $d_3(x), u_4(x), d_4(y), u_6(y)$
\item $i_k$: $d_3(x), u_4(x), d_4(y), u_2(y)$
\item $i_l$: $d_3(x), u_4(x), d_4(y), u_3(y)$
\item $l_g$: $d_4(y), u_3(y), d_3(x), u_5(x)$
\item $l_h$: $d_4(y), u_3(y), d_3(x), u_6(x)$
\item $l_i$: $d_4(y), u_3(y), d_3(x), u_4(x)$
\end{itemize}

Paths:

0, 1, 2, 4, 5, 6 (satisfies $a_j$)
0, 1, 2, 3, 5, 6 (satisfies $e_g, e_h$)
0, 1, 2, 3, 5, 2, 4, 5, 2, 3, 5, 6
\hspace{10pt} (satisfies $e_i, i_j, i_k, i_l, l_h$)
0, 1, 2, 4, 5, 2, 3, 5, 6 (satisfies $a_k, a_l, l_g$)
\hspace{10pt} (but not $l_i$)

---

Laski's and Korel's Criteria

- **Foundation:**
  - Combinations of definitions that reach uses at some node via a subpath
  - A single node can reference more than one variable
    - Need to consider all the combinations of definitions that can reach a node

- **Reach Coverage**
  - Some definition-clear subpath from each definition to all uses reached by that definition
    - basically the same as all-uses
Laski's and Korel's Criteria

• **Context Coverage**
  
  Some subpath along which each set of definitions reach uses at each node

  \[ :=x..y..z \]

![Diagram showing context coverage](image)

Laski's and Korel's Criteria

• **Ordered Context Coverage**
  
  Some subpath along which each sequence of definitions reach uses at each node

  \[ :=x..y..z \]

![Diagram showing ordered context coverage](image)
Context Coverage

\[ DC(n_6) = \{d_1(x), d_4(y)\} \quad a \]
\[ \quad \{d_3(x), d_0(y)\} \quad b \]
\[ \quad \{d_3(x), d_4(y)\} \quad c \]

Paths
1, 2, 4, 5, 6 (satisfies a)
1, 2, 3, 5, 6 (satisfies b)
1, 2, 3, 5, 2, 4, 5, 6 (satisfies c)

Note: must compute the sets for each node

Ordered Context Coverage

\[ ODC(n_6) = [d_1(x), d_4(y)] \quad a \]
\[ \quad [d_0(y), d_3(x)] \quad b \]
\[ \quad [d_3(x), d_4(y)] \quad c \]
\[ \quad [d_4(y), d_3(x)] \quad d \]

Paths
1, 2, 4, 5, 6 (satisfies a)
1, 2, 3, 5, 6 (satisfies b)
1, 2, 3, 5, 2, 4, 5, 6 (satisfies c)
1, 2, 4, 5, 2, 3, 5, 6 (satisfies a,c)

Note: must compute the sequences for each node
How can we compare these criteria?

• all select a set of paths, so compare the paths that they select
  set of paths that satisfy a criterion are not necessarily unique
  e.g., s1 or s2 satisfies criterion A
  s1, s2, or s3 satisfy criterion B

• define a subsumption relationship
  criterion A subsumes criterion B iff for any set of paths P in a flow graph
  P satisfies A => P satisfies B
  criterion A is equivalent to criterion B
  iff A subsumes B and B subsumes A
Relationships among these criteria

ORDERED CONTEXT COVERAGE \(\rightarrow\) All-Paths

CONTEXT COVERAGE \(\rightarrow\) All-Uses

REACH COVERAGE \(\rightarrow\) All-Defs

All-C-Uses/Some-P-Uses \(\rightarrow\) All-P-Uses/Some-C-Uses

Should we define yet another criteria?

- could subsume all the others (except all paths)?

"the NEW Winner"
Problems with data flow coverage criteria

• infeasible paths
  • Don't usually get 100% coverage

• Need to understand fault detection ability

• Artificially combines control with data flow
  • Considering p-uses or all predicate alternatives, tacked on to incorporate control flow

Conclusions

• An improvement over control flow techniques
• Provides a rational for how many times to iterate a loop or which combinations of subpaths to consider
• Most commonly used criterion is all-uses (with branch coverage)
• Need more empirical evidence to evaluate effectiveness
• Typically used in a tool that monitors coverage