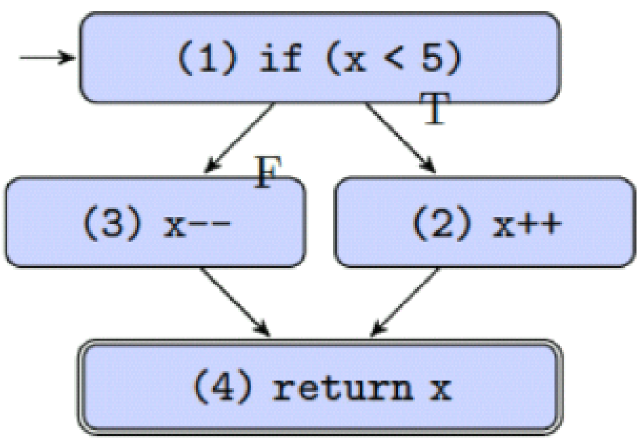
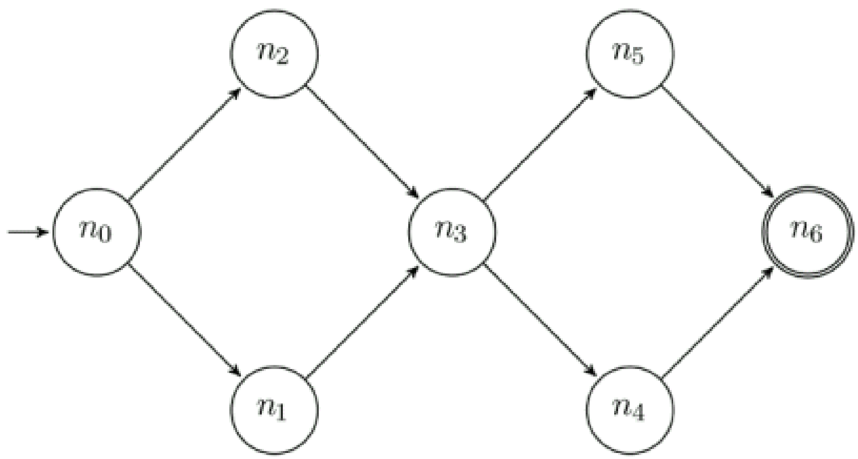
Connecting Test Cases, Test Paths & CFG

```
int foo(int x) {
  if (x < 5) {
    x ++;
  } else {
    x --;
  }
  return x;
}
```



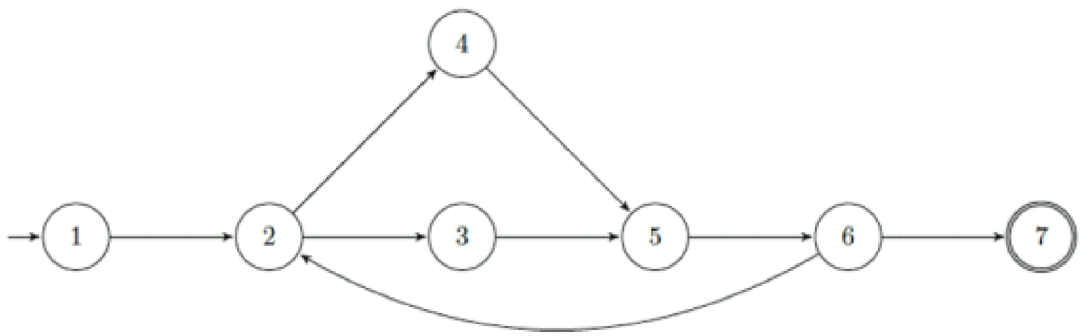
- Test case: $x = 5$; test path: $[(1), (3), (4)]$.
- Test case: $x = 2$; test path: $[(1), (2), (4)]$.

Node Coverage



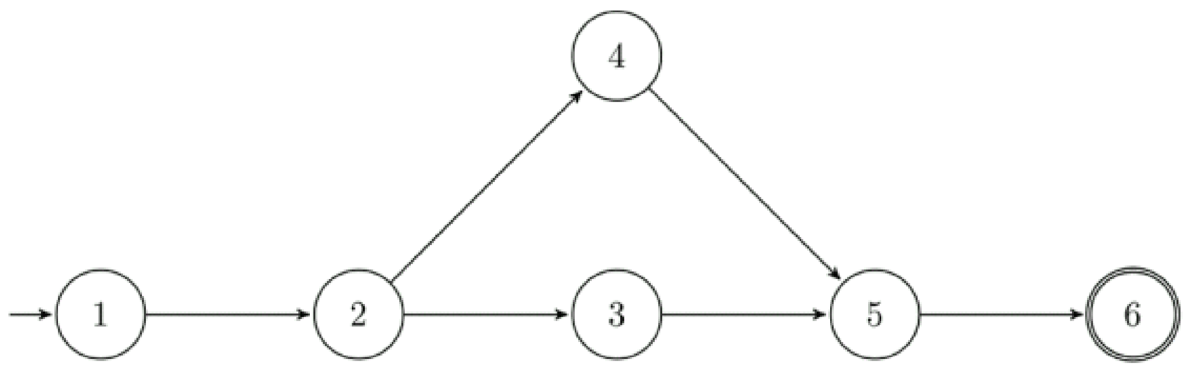
- *Node coverage: For each node $n \in reachG[N_0]$, TR contains a requirement to visit node n.*
- *Node Coverage [NC]: TR contains each **reachable** node in G.*
- TR = {n0,n1,n2,n3,n4,n5,n6}

Edge Coverage



- *Edge Coverage [EC]: TR contains each **reachable** path of length up to 1, inclusive, in G.*
- *TR = {[1,2], [2,4], [2,3], [3,5], [4,5], [5,6] [6,7], [6,2]}*

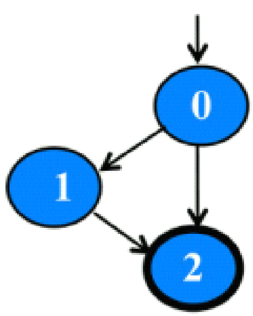
Edge Pair Coverage



- *Edge-Pair Coverage [EPC]: TR contains each* ***reachable*** *path of length up to 2, inclusive, in G.*
- *TR= {[1,2,3], [1,2,4], [2,3,5], [2,4,5], [3,5,6], [4,5,6]}*

Node and Edge Coverage

- **Edge coverage is slightly stronger than node coverage.**
- **NC and EC are only different when there is an edge and another subpath between a pair of nodes [as in an "if-else" statement]**



Node Coverage : TR = { 0, 1, 2 }
Test Path = [0, 1, 2]

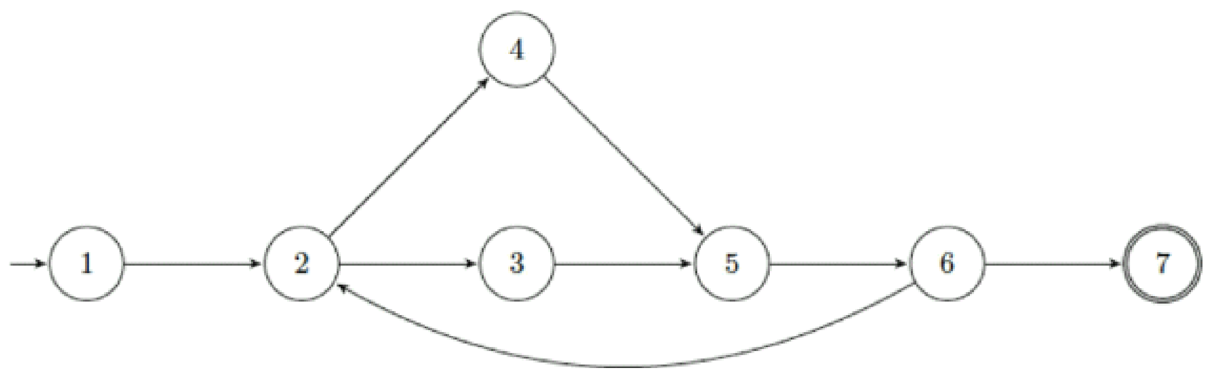
Edge Coverage : TR = { [0,1], [0, 2], [1, 2] }
Test Paths = [0, 1, 2]
[0, 2]

Simple Path

- A path is simple if no node appears more than once in the path, except that the first and last nodes may be the same.

- Some properties of simple paths:
  - no internal loops;
  - can bound their length;
  - can create any path by composing simple paths; and
  - many simple paths exist [too many!]

Simple Path Examples



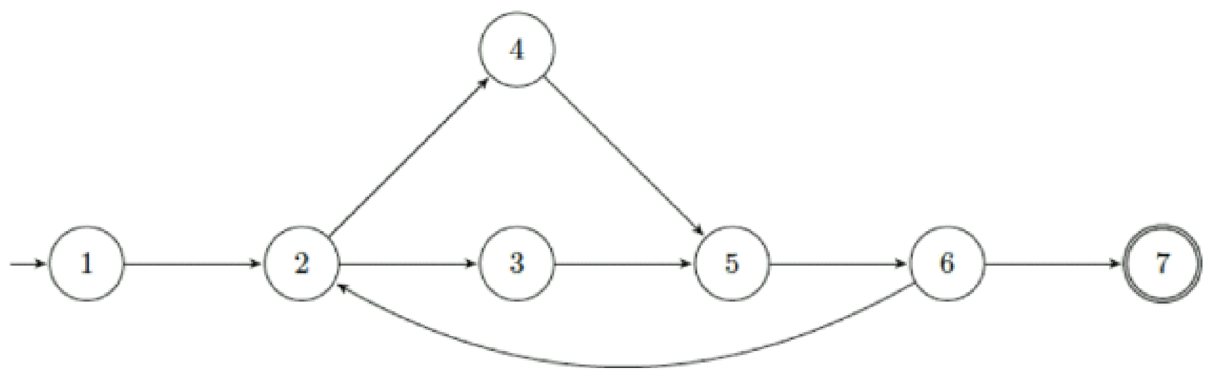
- Simple path examples:
  - [1, 2, 3, 5, 6, 7]
  - [1, 2, 4]
  - [2,3,5,6,2]
- Not simple Path: [1,2,3,5,6,2,4]

Prime Path

- Because there are so many simple paths, let's instead consider prime paths, which are simple paths of maximal length.

- A path is prime if it is simple and does not appear as a proper subpath of any other simple path.

Prime Path Examples



- Prime path examples:
  - [1, 2, 3, 5, 6, 7]
  - [1, 2, 4, 5, 6, 7]
  - [6, 2, 4, 5, 6]
- Not a prime path: [3, 5, 6, 7]

Prime Path Coverage

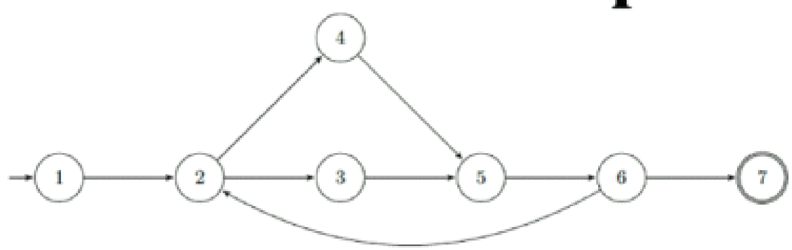
- Prime Path Coverage [PPC]: TR contains each prime path in G.

- There is a problem with using PPC as a coverage criterion: a prime path may be infeasible but contains feasible simple paths.
  - How to address this issue?

More Path Coverage Criterions

- Complete Path Coverage [CPC]: TR contains all paths in G.
- Specified Path Coverage [SPC]: TR contains a specified set S of paths.

Prime Path Example



Simple paths

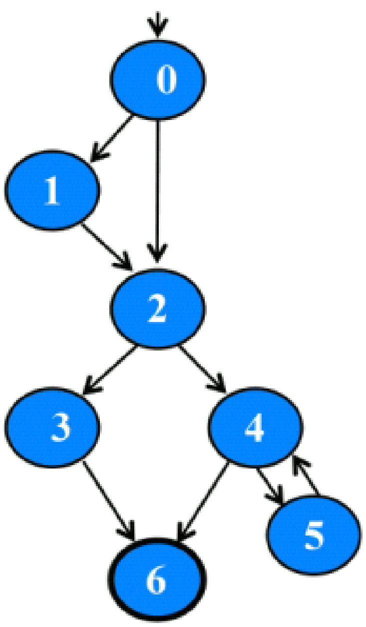
Len 0	Len 1	Len 2	Len 3	Len 4	Len 5
[1]	[1,2]				
[2]	[2,4]				
[3]	[2,3]				
[4]	[3,5]				
[5]	[4,5]				
[6]	[5,6]				
[7]!	[6,7]!				
	[6,2]				

53 Simple Paths
12 Prime Paths

! means path terminates

Prime Path Example (2)

- This graph has 38 simple paths
- Only 9 *prime paths*



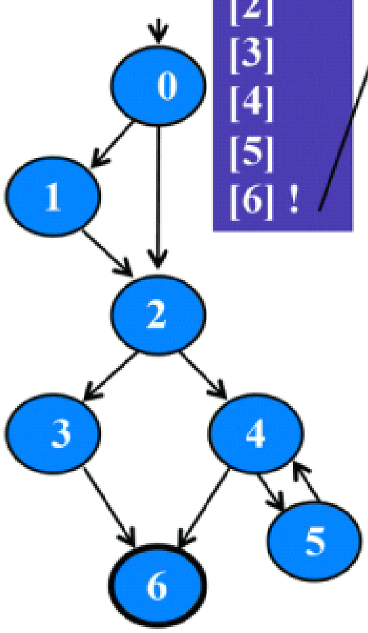
Prime Paths

[0, 1, 2, 3, 6]
[0, 1, 2, 4, 5]
[0, 1, 2, 4, 6]
[0, 2, 3, 6]
[0, 2, 4, 5]
[0, 2, 4, 6]
[5, 4, 6]
[4, 5, 4]
[5, 4, 5]

Prime Path Example (2)

! means path terminates

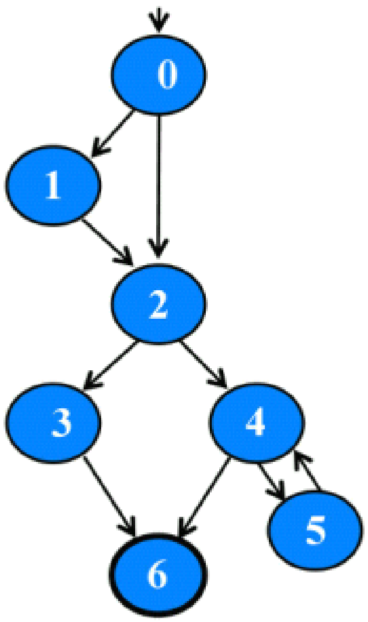
Simple paths



Len 0	Len 1	Len 2	Len 3	Len 4
[0]	[0, 1]	[0, 1, 2]	[0, 1, 2, 3]	[0, 1, 2, 3, 6] !
[1]	[0, 2]	[0, 2, 3]	[0, 1, 2, 4]	[0, 1, 2, 4, 6] !
[2]	[1, 2]	[0, 2, 4]	[0, 2, 3, 6] !	[0, 1, 2, 4, 5] !
[3]	[2, 3]	[1, 2, 3]	[0, 2, 4, 6] !	
[4]	[2, 4]	[1, 2, 4]	[0, 2, 4, 5] !	
[5]	[3, 6] !	[2, 3, 6] !	[1, 2, 3, 6] !	
[6] !	[4, 6] !	[2, 4, 6] !	[1, 2, 4, 5] !	
	[4, 5]	[2, 4, 5] !	[1, 2, 4, 6] !	
	[5, 4]	[4, 5, 4] *		
		[5, 4, 6] !		
		[5, 4, 5] *		

* denotes path cycles

9 Prime Paths



Node Coverage

TR = { 0, 1, 2, 3, 4, 5, 6 }

Test Paths: [0, 1, 2, 3, 6] [0, 1, 2, 4, 5, 4, 6]

Edge Coverage

TR = { [0,1], [0,2], [1,2], [2,3], [2,4], [3,6], [4,5], [4,6], [5,4] }

Test Paths: [0, 1, 2, 3, 6] [0, 2, 4, 5, 4, 6]

Edge-Pair Coverage

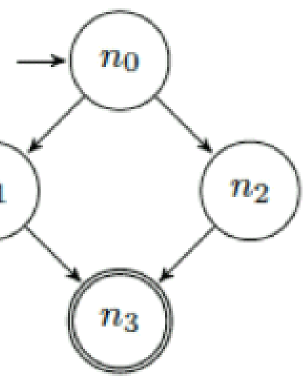
TR = { [0,1,2], [0,2,3], [0,2,4], [1,2,3], [1,2,4], [2,3,6], [2,4,5], [2,4,6], [4,5,4], [5,4,5], [5,4,6] }

Test Paths: [0, 1, 2, 3, 6] [0, 2, 3, 6] [0, 2, 4, 5, 4, 5, 4, 6] [0, 1, 2, 4, 6]

Complete Path Coverage

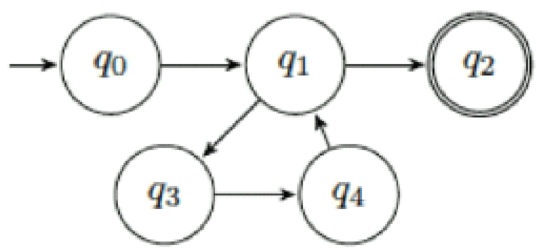
Test Paths: [0, 1, 2, 3, 6] [0, 1, 2, 4, 6] [0, 1, 2, 4, 5, 4, 6] [0, 1, 2, 4, 5, 4, 5, 4, 6] [0, 1, 2, 4, 5, 4, 5, 4, 5, 4, 6] …

Prime Path Coverage vs. Complete Path Coverage



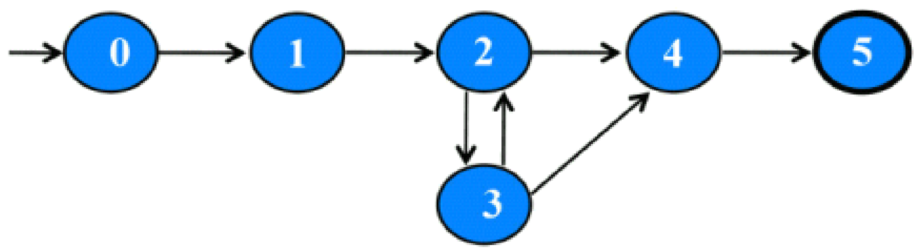
- Prime paths:
- $\text{path}(t_1) =$
- $\text{path}(t_2) =$
- $T_1 = \{t_1, t_2\}$ satisfies both PPC and CPC.

Prime Path Coverage vs. Complete Path Coverage (2)



- Prime paths:
- $\text{path}(t_3) =$
- $\text{path}(t_4) =$
- $T_1 = \{t_3, t_4\}$ satisfies PPC but not CPC.

Sidetrips and Detours



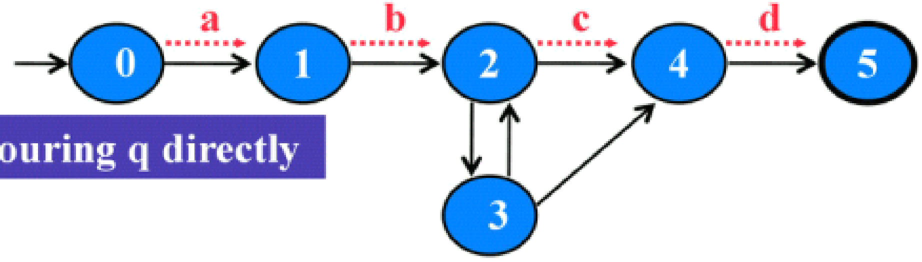
- It may not be possible for any test path to tour path q = [1,2,4].
  - Why?

- Any path that visits 3 does not tour path q.
  - p1 =[0, 1, 2, 3, 2, 4, 5]
  - p2 =[0, 1, 2, 3, 4, 5]

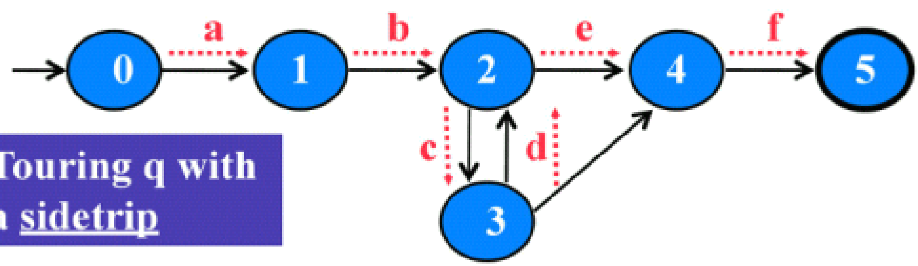
- But p1 and p2 tour q in a more general sense.

Sidetrips and Detours Example

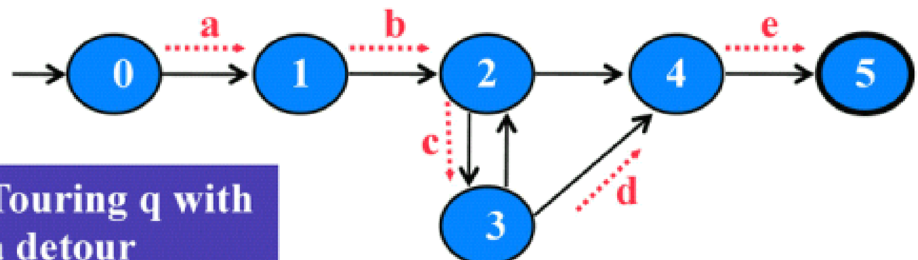
q = [1,2, 4]



Touring q directly



Touring q with a <u>sidetrip</u>

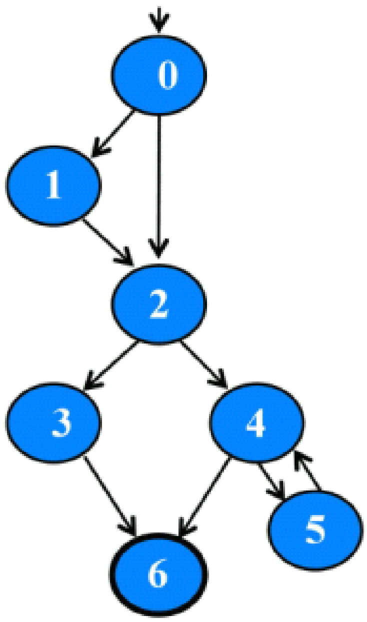


Touring q with a <u>detour</u>

Tour with Sidetrips and Detours

- For the following definitions, assume that q is simple.
- Tour (directly): Test path p tours subpath q iff q is a **subpath** of p.
- Tour with sidetrips: Test path p tours subpath q with sidetrips iff every **edge** in q is also in p, in the same order.
- Tour with detours: Test path p tours subpath q with detours iff every **node** in q is also in p, in the same order.

Sidetrips and Detours Example (2)



- p = [0, 1, 2, 4, 5, 4, 6]
- p1 = [0, 1, 2, 4, 6]
- p2 = [0, 2, 4, 5,4, 6]

- p tours p1 with a sidetrip.
- p tours p2 with a detour.

Refining Coverage Criteria

- We could define each graph coverage criterion and explicitly include the kinds of tours allowed, e.g.
  - prime paths, with direct tours;
  - prime paths, sidetrips allowed;
  - prime paths, detours allowed.

- Detours seem less practical, so we do not include detours further.
- We do need sidetrips sometimes, or too many TRs are infeasible.

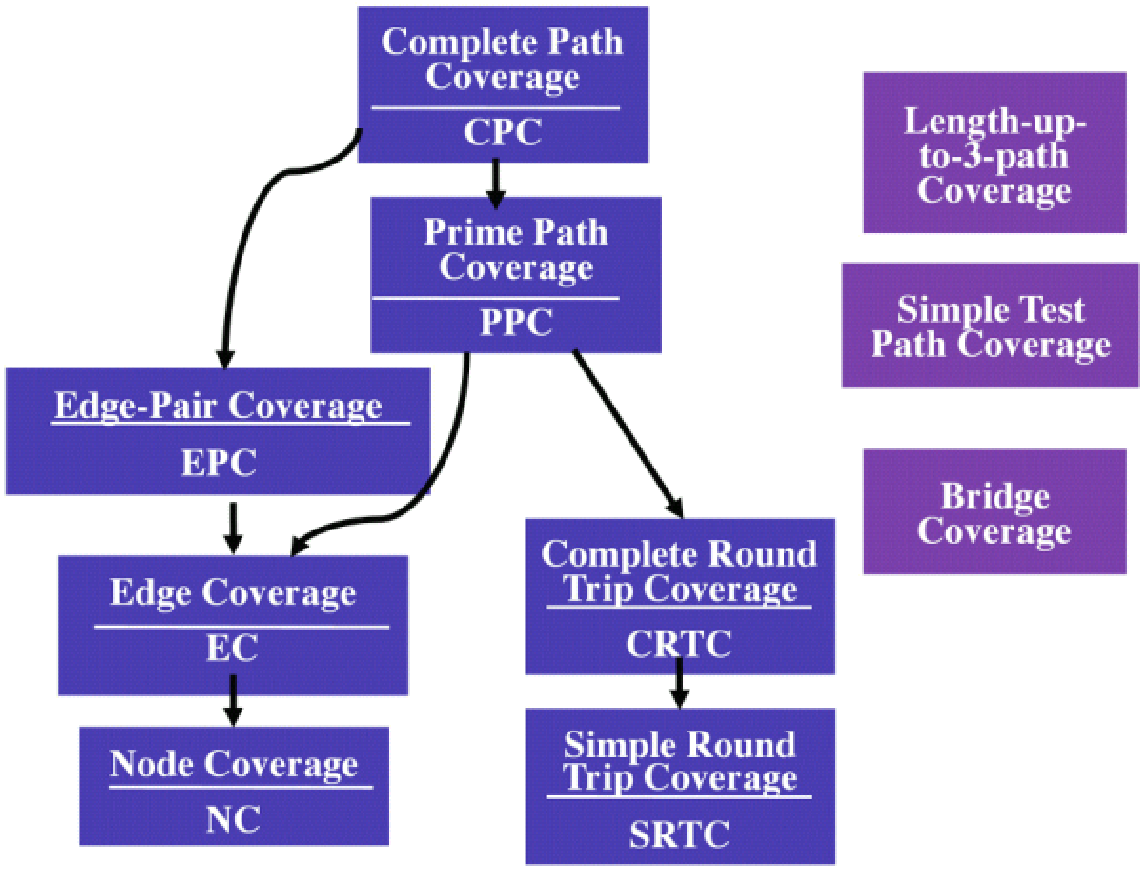
Best Effort Touring

- We'd rather not use sidetrips when we don't have to.
- TR_{tour}: the subset of test requirements that can be toured [directly]
- $TR_{sidetrip}$: the subset of test requirements that can be toured with sidetrips.
- Best Effort Touring: A set T of test paths achieves best effort touring if for every path p in TR_{tour}, some path in T tours p [directly], and for every path p in $TR_{sidetrip}$, some path in T tours p either directly or with a sidetrip.
- Best-effort touring meets as many TRs as possible, each in the strictest possible way. Best-effort touring has desirable theoretical properties with respect to subsumption.

Round trip path

- A round trip path is a prime path of nonzero length that starts and ends at the same node.
- Simple Round Trip Coverage [SRTC]: TR contains at least one round-trip path for each reachable node in G that begins and ends a round-trip path.
- Complete Round Trip Coverage [CRTC]: TR contains all round-trip paths for each reachable node in G.

Graph Coverage Criteria Subsumption



Exercise

- Answer questions [a]-[g] for the graph defined by the following sets:
  - N ={1, 2, 3, 4, 5, 6, 7}
  - N0 = {1}
  - Nf = {7}
  - E = {[1, 2], [1, 7], [2, 3], [2, 4], [3, 2], [4, 5], [4, 6], [5, 6], [6, 1]}
- Also consider the following test paths:
  - t0 = [1, 2, 4, 5, 6, 1, 7]
  - t1 = [1, 2, 3, 2, 4, 6, 1, 7]

Exercise - cont.

- [a] Draw the graph.
- [b] List the test requirements for EPC. [Hint: You should get 12 requirements of length 2].
- [c] Does the given set of test paths satisfy EPC? If not, identify what is missing.
- [d] Consider the simple path [3,2,4,5,6] and test path [1,2,3,2,4,6,1,2,4,5,6,1,7]. Does the test path tour the simple path directly? With a sidetrip? Is so, identify the sidetrip.
- [e] List the test requirements for NC, EC and PPC on the graph.
- [f] List a test path that achieve NC but not EC on the graph.
- [g] List a test path that achieve EC but not PPC on the graph.