Graph Coverage Web Application

Graph Information

Please enter your **graph edges** in the text box below. Put each edge in one line. Enter edges as pairs of nodes, separated by spaces.(e.g.: 1 3)	Enter **initial nodes** below (can be more than one), separated by spaces. If the text box below is empty, the first node in the left box will be the initial node.	Enter **final nodes** below (can be more than one), separated by spaces.
1 2 1 3 3 4 3 5 5 6	1	2 4 13

Test Requirements: Nodes | Edges | Edge-Pair | Simple Paths | Prime Paths

Test Paths: Algorithm 1: Slower, more test paths, shorter test paths Node Coverage | Edge Coverage | Edge-Pair Coverage | Prime Path Coverage

Algorithm 2: Faster, fewer test paths, longer test paths Edge Coverage | Edge-Pair Coverage | Prime Path Coverage

Algorithm 1 is our original, not particularly clever, algorithm to find test paths from graph coverage test requirements. In our 2012 ICST paper, "*Better Algorithms to Minimize the Cost of Test Paths*," we described an algorithm that combines test requirements to produce fewer, but longer test paths (algorithm 2). Users can evaluate the tradeoffs between more but shorter test paths and fewer but longer test paths and choose the appropriate algorithm.

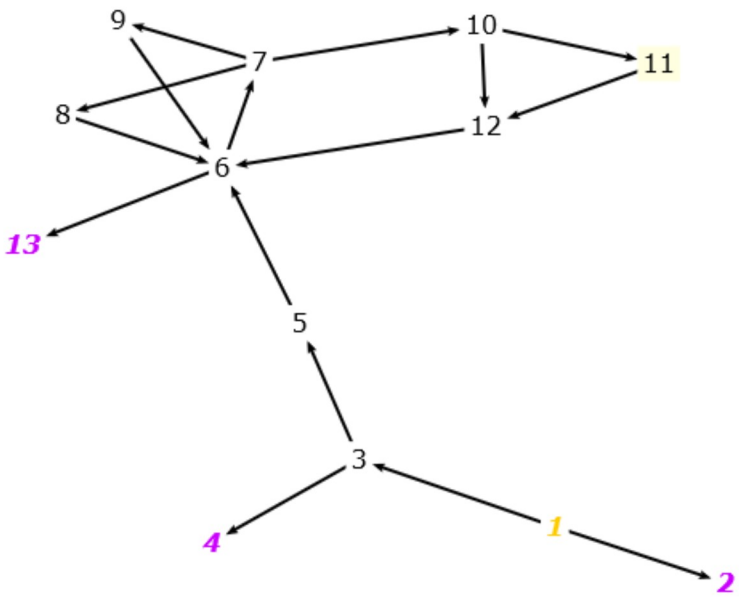
Other Tools: New Graph | Data Flow Coverage | Logic Coverage | Minimal-MUMCUT Coverage

Share Graph: Share

19 test paths are needed for Prime Path Coverage

Test Paths	Test Requirements that are toured by test paths directly
[1,3,5,6,7,10,11,12,6,7,8,6,13]	[1,3,5,6,7,10,11,12], [6,7,10,11,12,6], [7,10,11,12,6,7], [10,11,12,6,7,8], [7,8,6,13], [6,7,8,6]
[1,3,5,6,7,8,6,7,10,11,12,6,13]	[6,7,10,11,12,6], [7,10,11,12,6,13], [1,3,5,6,7,8], [8,6,7,10,11,12], [7,8,6,7], [6,7,8,6]
[1,3,5,6,7,9,6,7,10,11,12,6,13]	[6,7,10,11,12,6], [7,10,11,12,6,13], [1,3,5,6,7,9], [9,6,7,10,11,12], [7,9,6,7], [6,7,9,6]
[1,3,5,6,7,10,12,6,7,10,11,12,6,13]	[1,3,5,6,7,10,12], [6,7,10,11,12,6], [7,10,11,12,6,13], [12,6,7,10,11,12], [6,7,10,12,6], [7,10,12,6,7], [10,12,6,7,10]
[1,3,5,6,7,10,11,12,6,7,10,11,12,6,13]	[1,3,5,6,7,10,11,12], [6,7,10,11,12,6], [7,10,11,12,6,13], [7,10,11,12,6,7], [12,6,7,10,11,12],

Node color: Initial Node, Final Node



	[11,12,6,7,10,11], [10,11,12,6,7,10]
[1,3,5,6,7,10,11,12,6,7,9,6,13]	[1,3,5,6,7,10,11,12], [6,7,10,11,12,6], [7,10,11,12,6,7], [10,11,12,6,7,9], [7,9,6,13], [6,7,9,6]
[1,3,5,6,7,10,11,12,6,7,10,12,6,13]	[1,3,5,6,7,10,11,12], [6,7,10,11,12,6], [7,10,11,12,6,7], [10,11,12,6,7,10], [6,7,10,12,6], [7,10,12,6,13], [12,6,7,10,12]
[1,3,5,6,7,10,12,6,7,8,6,13]	[1,3,5,6,7,10,12], [6,7,10,12,6], [7,10,12,6,7], [10,12,6,7,8], [7,8,6,13], [6,7,8,6]
[1,3,5,6,13]	[1,3,5,6,13]
[1,3,5,6,7,8,6,7,10,12,6,13]	[1,3,5,6,7,8], [6,7,10,12,6], [7,10,12,6,13], [8,6,7,10,12], [7,8,6,7], [6,7,8,6]
[1,3,5,6,7,9,6,7,10,12,6,13]	[1,3,5,6,7,9], [6,7,10,12,6], [7,10,12,6,13], [9,6,7,10,12], [7,9,6,7], [6,7,9,6]
[1,3,5,6,7,10,12,6,7,10,12,6,13]	[1,3,5,6,7,10,12], [6,7,10,12,6], [7,10,12,6,13], [7,10,12,6,7], [12,6,7,10,12], [10,12,6,7,10]
[1,3,5,6,7,10,12,6,7,9,6,13]	[1,3,5,6,7,10,12], [6,7,10,12,6], [7,10,12,6,7], [10,12,6,7,9], [7,9,6,13], [6,7,9,6]
[1,3,5,6,7,8,6,7,8,6,13]	[1,3,5,6,7,8], [7,8,6,13], [7,8,6,7], [6,7,8,6], [8,6,7,8]
[1,3,5,6,7,9,6,7,8,6,13]	[1,3,5,6,7,9], [7,8,6,13], [7,9,6,7], [6,7,9,6], [6,7,8,6], [9,6,7,8]
[1,3,5,6,7,9,6,7,9,6,13]	[1,3,5,6,7,9], [7,9,6,7], [7,9,6,13], [6,7,9,6], [9,6,7,9]
[1,3,5,6,7,8,6,7,9,6,13]	[1,3,5,6,7,8], [7,8,6,7], [7,9,6,13], [6,7,9,6], [6,7,8,6], [8,6,7,9]
[1,3,4]	[1,3,4]
[1,2]	[1,2]

Test Paths	Test Requirements that are toured by

	test paths with sidetrips
[1,3,5,6,7,10,11,12,6,7,8,6,13]	[6,7,10,11,12,6], [7,10,11,12,6,13], [1,3,5,6,7,8], [6,7,8,6]
[1,3,5,6,7,8,6,7,10,11,12,6,13]	[1,3,5,6,7,10,11,12], [6,7,10,11,12,6], [7,8,6,13], [6,7,8,6]
[1,3,5,6,7,9,6,7,10,11,12,6,13]	[1,3,5,6,7,10,11,12], [6,7,10,11,12,6], [7,9,6,13], [6,7,9,6]
[1,3,5,6,7,10,12,6,7,10,11,12,6,13]	[1,3,5,6,7,10,11,12], [1,3,5,6,7,10,12], [6,7,10,11,12,6], [7,10,11,12,6,13], [6,7,10,12,6], [7,10,12,6,13]
[1,3,5,6,7,10,11,12,6,7,10,11,12,6,13]	[1,3,5,6,7,10,11,12], [6,7,10,11,12,6], [7,10,11,12,6,13]
[1,3,5,6,7,10,11,12,6,7,9,6,13]	[6,7,10,11,12,6], [7,10,11,12,6,13], [1,3,5,6,7,9], [6,7,9,6]
[1,3,5,6,7,10,11,12,6,7,10,12,6,13]	[1,3,5,6,7,10,11,12], [1,3,5,6,7,10,12], [6,7,10,11,12,6], [7,10,11,12,6,13], [6,7,10,12,6], [7,10,12,6,13]
[1,3,5,6,7,10,12,6,7,8,6,13]	[1,3,5,6,7,8], [6,7,10,12,6], [7,10,12,6,13], [6,7,8,6]
[1,3,5,6,13]	None
[1,3,5,6,7,8,6,7,10,12,6,13]	[1,3,5,6,7,10,12], [6,7,10,12,6], [7,8,6,13], [6,7,8,6]
[1,3,5,6,7,9,6,7,10,12,6,13]	[1,3,5,6,7,10,12], [6,7,10,12,6], [7,9,6,13], [6,7,9,6]
[1,3,5,6,7,10,12,6,7,10,12,6,13]	[1,3,5,6,7,10,12], [6,7,10,12,6], [7,10,12,6,13]
[1,3,5,6,7,10,12,6,7,9,6,13]	[1,3,5,6,7,9], [6,7,10,12,6], [7,10,12,6,13], [6,7,9,6]
[1,3,5,6,7,8,6,7,8,6,13]	[1,3,5,6,7,8], [7,8,6,13], [6,7,8,6]
[1,3,5,6,7,9,6,7,8,6,13]	[1,3,5,6,7,8], [7,9,6,13], [6,7,9,6], [6,7,8,6]
[1,3,5,6,7,9,6,7,9,6,13]	[1,3,5,6,7,9], [7,9,6,13], [6,7,9,6]
[1,3,5,6,7,8,6,7,9,6,13]	[1,3,5,6,7,9], [7,8,6,13], [6,7,9,6],

	[6,7,8,6]
[1,3,4]	None
[1,2]	None

Infeasible prime paths are:
None

List any infeasible sub paths in the box below. Enter sub paths
as strings of nodes, separated by commas.
Sub paths you mark as infeasible will **not** be used
in any test paths.
Example: 3,4,7,1,2,3,4,7,1