Better Algorithms to Minimize the Cost of Test Paths

Nan Li, Fei Li, and Jeff Offutt

Presented by

Nan Li

George Mason University

04/18/2012

Overview

- Introduction and Motivation
- Prime Path Coverage
- Minimum Cost Test Paths Problem
- The Shortest Superstring Problem
- Algorithms
- Experiment Design
- Experiment Results and Analysis
- Threats to Validity
- Conclusions
- Future Work

Introduction and Motivation

- Model-driven test development
  - Model (graph)
  - Test Criteria
  - Test Requirements (subpaths)
  - Test Paths (paths from an initial node to a final node)

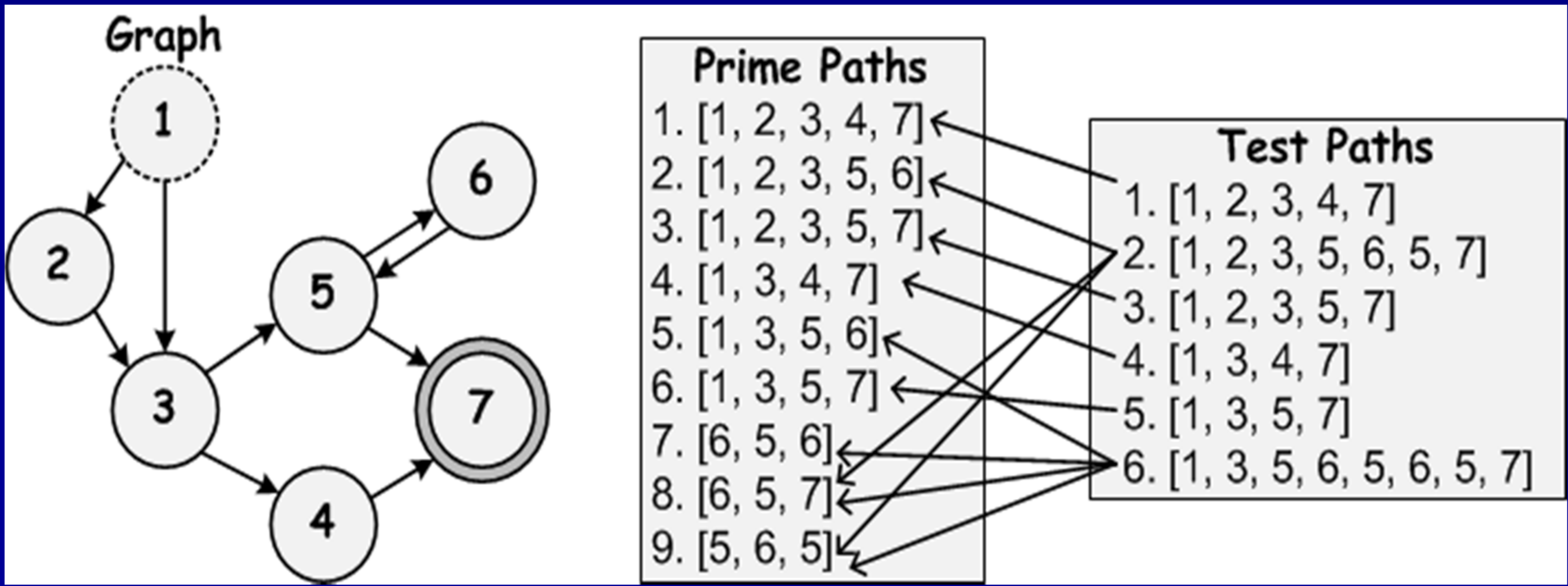
How to generate test paths to cover test requirements?

- Solution impacts the overall cost of testing

© Nan Li, 2012

Prime Path Coverage

- Simple Path: no node appears more than once, except possibly the first and last nodes are the same
- Prime Path: a simple path that does not appear as a proper subpath of any other simple path
- Prime Path Coverage Criterion: TR contains each prime path in G



© Nan Li, 2012

Minimum Cost Test Paths Problem

- Input: a set of test requirement $TR = \{r_1, r_2, \ldots, r_n\}$
  - Each test requirement is presented as a subpath in a graph $G = (V, E)$
- The problem MCTP is to find a set of test paths $TP = \{t_1, t_2, \ldots, t_k\}$ that cover all test requirements in the graph G such that the cost of using the test paths is minimum
  - Cost can be reduced in several ways
  - First defined in this research

Minimum Cost Test Paths Problem (cont.)

- Fewer test paths
  - Each test path represents a test
- Fewer total nodes
  - Each node represents lines of code
- Fewer test requirements per test path
- Shorter test paths
  - Finding test values for long test paths
- Achieving multiple goals is hard
  - Conflict: smaller TR / TP ratio and fewer test paths
  - Complementary: smaller TR / TP ratio and shorter test paths
  - Always valid: fewer total nodes

© Nan Li, 2012

Minimum Cost Test Paths Problem (cont.)

- Optimization of the goals:
  1. The total number of test paths
  2. The total number of nodes
  3. The maximum ratio of TR to TP
  4. The total number of test paths subject to a bounded ratio of TR to TP
  5. The total number of nodes subject to a bounded ratio of TR to TP

Minimum Cost Test Paths Problem (cont.)

- NP-completeness and reductions

Problem	NP-completeness	Reduction / Solution
Total number of test paths	P	Modified version algorithm used to solve CP_1[1] by Aho and Lee
Total number of nodes	NP-complete	Bin-Packing
Maximum ratio of TR to TP	NP-complete	Bin-Packing
Total number of test paths subject to a bounded ratio of TR to TP	NP-complete	Bin-Packing
Total number of nodes subject to a bounded ratio of TR to TP	NP-complete	Bin-Packing

- Use dynamic programming to solve other variants

© Nan Li, 2012

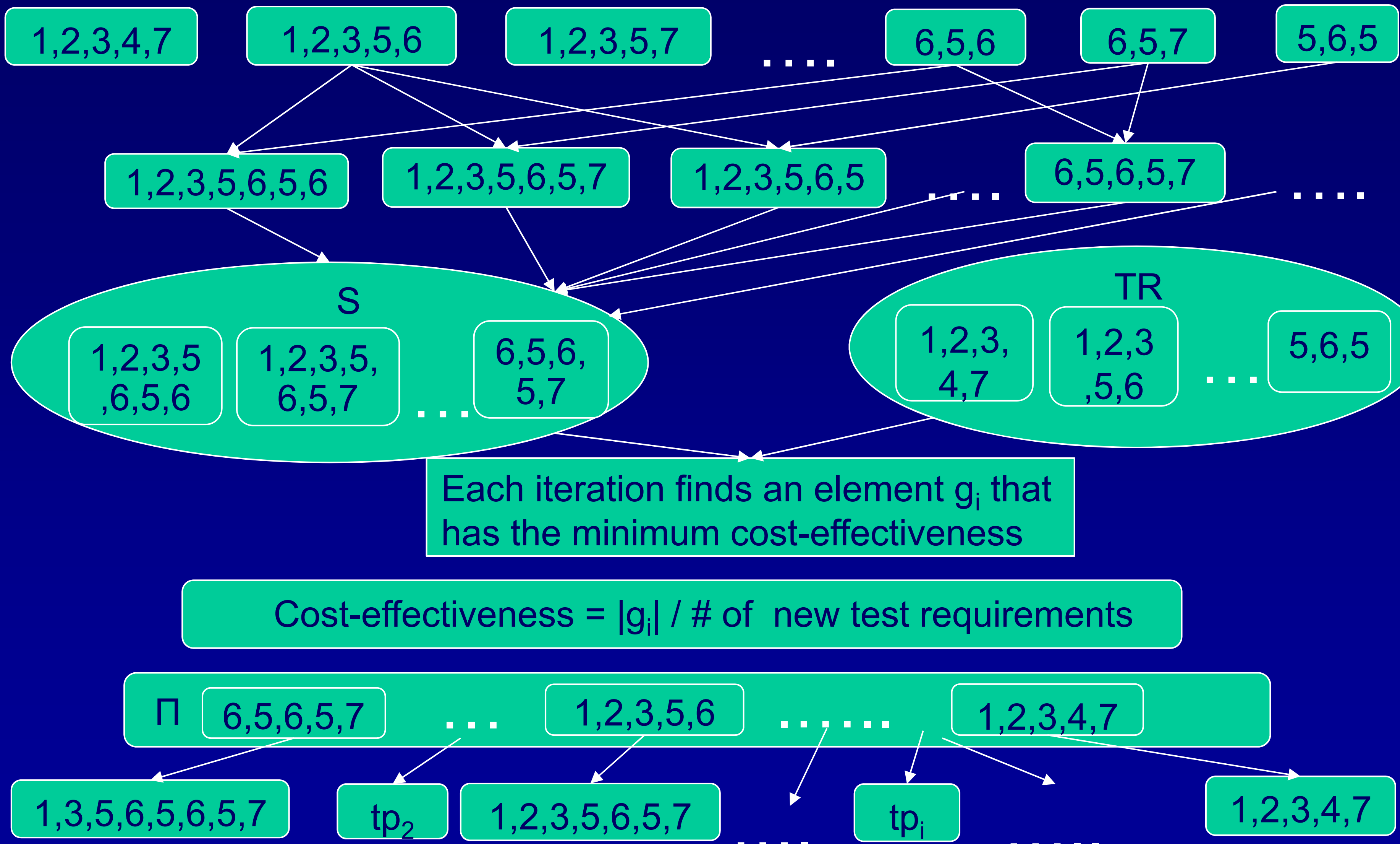
The Shortest Superstring Problem

- Input: a set of n strings, $S = \{s_1, \ldots, s_n\}$
- The shortest superstring problem is to find a shortest string s that contains each s_i as substring
  - NP-complete
  - The best approximation ratio is 2.0
  - If a string s and another string t have overlap x, s = mx and t = xn. |over (s, t)| = x; |prefix (s, t)| = m
  - In software testing, a string is a test requirement
  - Example: prime paths [1,2,3,1] and [2,3,1,2]; super-prime paths: [1,2,3,1,2] and [2,3,1,2,3,1]
  - Set-covering algorithm and matching-based prefix graph algorithm

Old Solution vs. New Solutions

- Current Solution
  - Used in the graph coverage web application
  - Straightforward (Breadth-first search) algorithm
  - Test minimization algorithm
- Set-covering based solution
  - Set-covering algorithm
  - Splitting algorithm
  - Test minimization algorithm
- Matching-based prefix graph solution
  - Prefix-graph based algorithm
  - Splitting algorithm
  - Test minimization algorithm

The Greedy Set-covering Solution



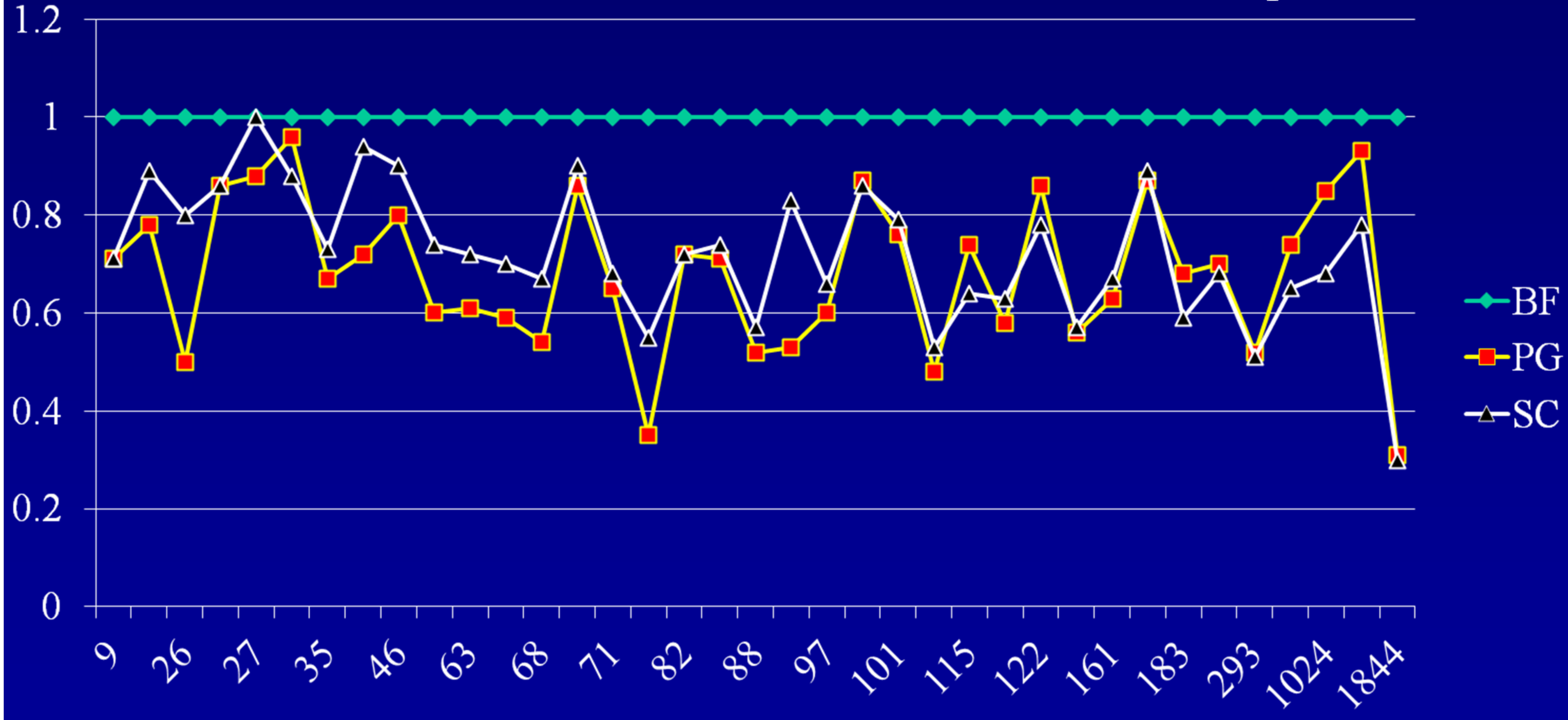
Experiment

- Subject
  - Methods from Java programs (four open source and one GMU project)
  - These methods have complex structures (nested loops)
  - 37 methods
  - Construct control-flow graphs from the methods
  - Test requirements: prime paths
  - Each method was measured with respect to number of prime paths: 9 to 1844
- Procedure
  - Run the graph coverage web application on one computer
  - Record the number of test paths, the total number of nodes, the maximum ratio of TR over TP and the execution time (mean time)
  - No interruption from Internet or other programs

Test Paths Ratios

Set-covering and prefix graph-based solutions generate fewer test paths and nodes than the current solution

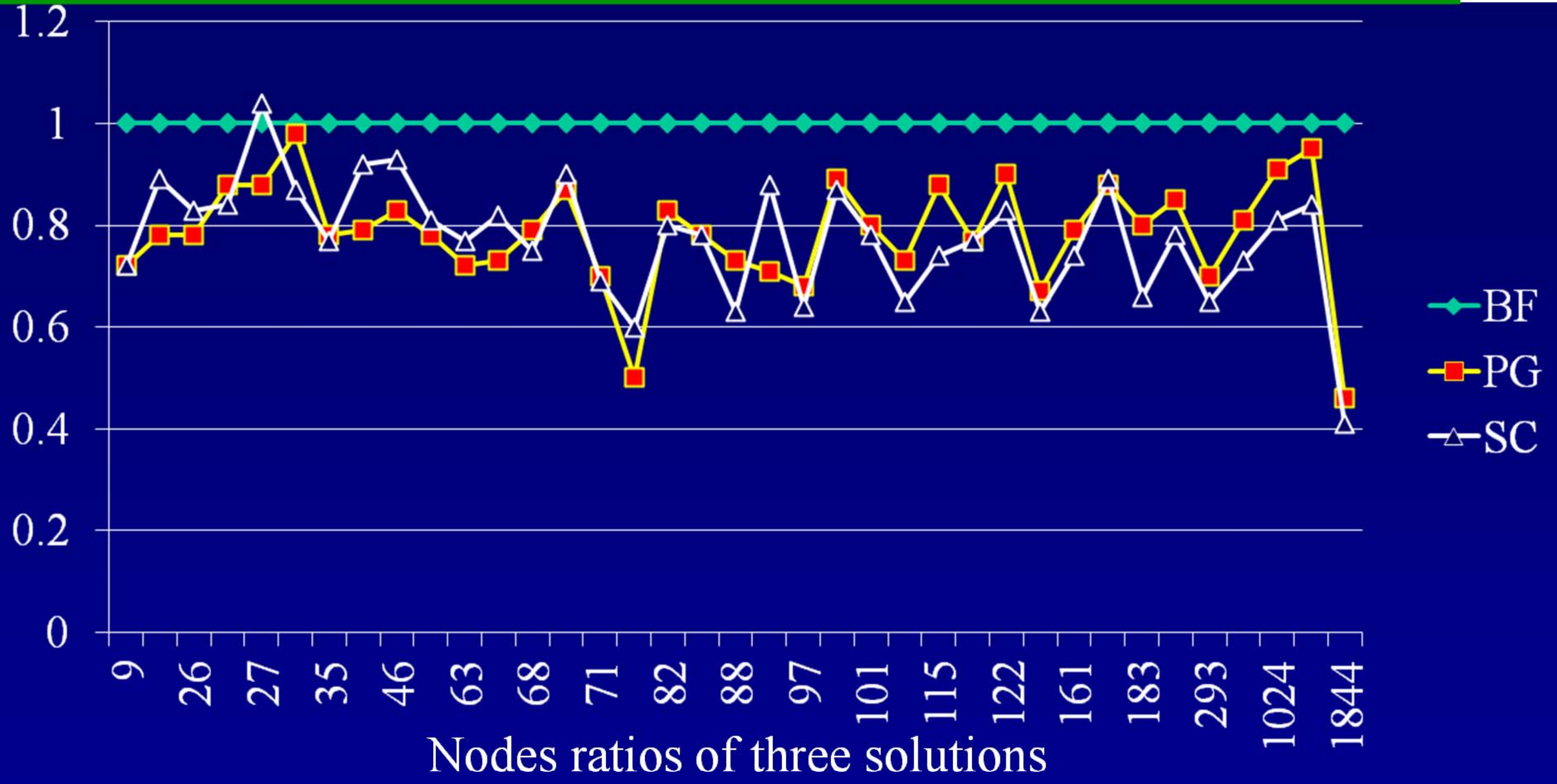
– Save 20 – 30% of the nodes and 30 -40 % of the test paths



Test paths ratios of three solutions

© Nan Li, 2012

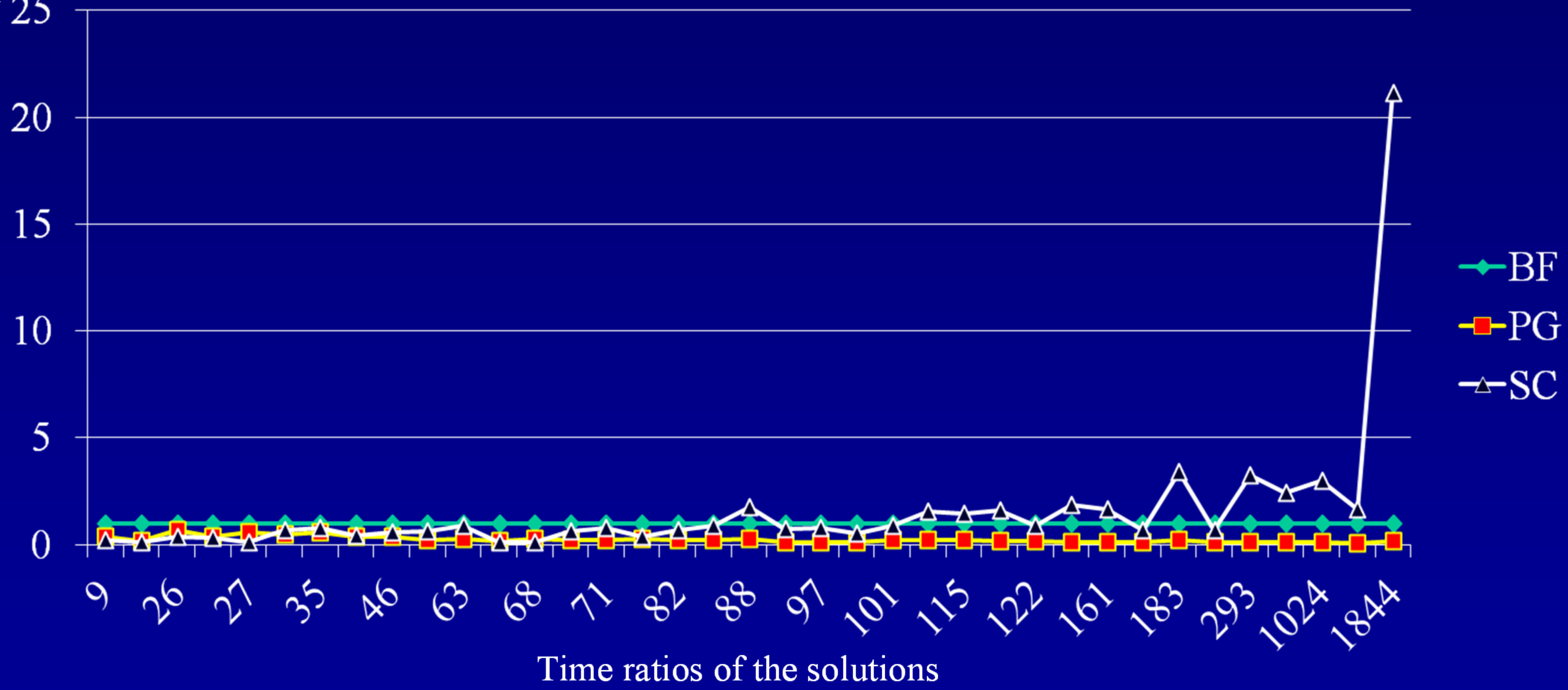
Nodes Ratios



- More savings on methods that have complex nested loops
  - Not able to quantify the complexity of methods
- Maximum ratio of TR / TP is higher for the set-covering and prefix graph-based solutions than the current algorithm

Time Ratios

The set-covering solution runs faster than the other two solutions when graphs have few prime paths and slower when graphs have more prime paths



We recommend the prefix graph-based algorithm

© Nan Li, 2012

Threats to Validity

- The subjects might not be representative
  – The results may not hold on other programs
- Implementation of these algorithms
- A different splitting algorithm or test minimization algorithm may have different results

Conclusions

- 37 methods were used
- Three solutions: the current brute force, set-covering, and matching-based prefix solutions
- Generate test paths to cover prime paths
- The prefix-graph based and set-covering based solutions generated fewer test paths and nodes than the current solution
- The set-covering based solution took much longer time on graphs that had more prime paths
- Prefix-graph based solution is preferable

© Nan Li, 2012

Future Work

- Try other shortest superstring algorithms
- Quantify properties of methods
  - Number of prime paths
  - Overlaps among the prime paths
  - Other factors
- Different splitting and test minimization algorithms
- Apply additional algorithms such as dynamic programming to our set-covering and prefix-graph based solutions to solve the variants of MCTP
- Integrate new algorithms into test generation tools

Contact

Nan Li

Email: nli1@gmu.edu

Homepage: *http://cs.gmu.edu/~nli1*

Graph web application:

http://cs.gmu.edu:8080/offutt/coverage/GraphCoverage

© Nan Li, 2012