Program Analysis
Path Profiling (Part 3)

Prof. Dr. Michael Pradel
Software Lab, University of Stuttgart
Winter 2020/2021
Outline

1. Motivation and Challenges
2. Ball-Larus algorithm for DAGs
3. Generalization and Applications

Mostly based on this paper:
- Efficient path profiling, Ball and Larus, MICRO 1996

Other reading material:
- Whole program paths, Larus, PLDI 1999
- HOLMES: Effective statistical debugging via efficient path profiling, Chilimbi et al., ICSE 2009
Generalizing to Cyclic CFGs

- For each backedge \( n \rightarrow m \), add dummy edges
  - \( \text{Entry} \rightarrow m \)
  - \( n \rightarrow \text{Exit} \)

- Remove backedges and add DAG-based increments

- In addition, add instrumentation to each backedge
  - \( \text{count}[r]++; \ r=0 \)
Generalizing to Cyclic CFGs (2)

- **Leads to four kinds of paths**
  - From entry to exit
  - From entry to backedge
  - From end of backedge to beginning of (possibly another) backedge
  - From end of backedge to exit

- **Full path information can be constructed from these four kinds**
Example: Generalizing

Graph:
- Nodes: A, B, C, D, E, F
- Edges: A to B (6), B to C (8), B to E, C to D, D to B (back edge), E to F
- Dummy edge: B to D

Encoding table:

<table>
<thead>
<tr>
<th>Path</th>
<th>Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>0</td>
</tr>
<tr>
<td>ABC EF</td>
<td>1</td>
</tr>
<tr>
<td>AB CE</td>
<td>2</td>
</tr>
<tr>
<td>AB DE</td>
<td>3</td>
</tr>
<tr>
<td>AB DEF</td>
<td>4</td>
</tr>
<tr>
<td>BCEF</td>
<td>5</td>
</tr>
<tr>
<td>BCE</td>
<td>6</td>
</tr>
<tr>
<td>BDEF</td>
<td>7</td>
</tr>
<tr>
<td>BDE</td>
<td>8</td>
</tr>
</tbody>
</table>
Applications

- **Performance optimization**
  - Frequent path should get most attention by optimizer

- **Statistical debugging**
  - Paths correlated with failure are more likely to contain the bug

- **Energy analysis**
  - Warn developers about paths and statements associated with high power consumption