Data Analysis and Reporting

Jeff Perkins and Michael Ernst

MIT CSAIL

Data Analysis and Reporting

- Processing time
  - online (as data is encountered)
  - offline (write data to file)
- Report information at machine or source level
  - May require some online processing
- Speed

Online Processing

- Can handle unbounded amounts of data
- Algorithm must be incremental
  - ○ Sometimes this is quite natural
  - ○ Other times it is quite complex
- Processing may affect target program

Offline Processing

- Output files can be very large (many gigabytes)
- Output file can be processed multiple times
- Development can be easier
  - Don’t need to rerun target program

Simple Example

- Basic block coverage
- Offline
  - When a basic block is executed, write its PC to an output file
  - Later, determine from the output file what blocks were covered
- Online
  - Keep a boolean for each basic block
  - Set the boolean when its basic block is executed
  - At the end of the run, dump the state of each boolean

Daikon Example

- Daikon infers invariants from a program trace
- Looks for invariants between each combination of variables
- Polynomial in the number of variables
- One optimization is equality:

  $x = y \wedge f(x) \Rightarrow f(y)$
- Easy to implement offline, first pass finds equal variables
- More complex to implement incrementally.

Path Profiling

- Initially looks complex -- must capture each branch choice to recreate the path.
- Fast incremental algorithm
  - Assign each path a unique id
  - Initialize the path id to 0 at entry
  - At each branch
    - Left branch does nothing
    - Right branch increments the id by the number of possible branches on the left branch
  - Result is a unique identifier for each possible path
- Path profiling can be faster than statement profiling
  - Only one id per method

Path Profiling example

- Source

```
id = 0    if (a)      stmt       if (b)         stmt       else         stmt         id += 1       endif    else       stmt          id += 2       if (c)         stmt       else         stmt ...
```

- Results: ab = 0; a!b = 1; !ac = 2; !a!c = 3;

Speed

- Instrumentation decisions are cheaper than runtime decisions
- Online solutions are often possible
- I/O is expensive