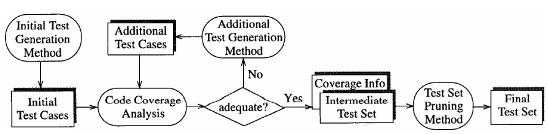
Data-flow and Control-flow Criteria Compared

- Coverage criteria monitor the thoroughness of software tests
  - Control-flow based
  - Data-flow based
- Are they effective?
- Which ones are more effective?

Experiments

- Goals
  - Comparing effectiveness of data-flow coverage and control-flow coverage for fault-detection
  - Is it necessary to achieve 100% coverage to benefit from a criterion?
- Criteria
  - Data-flow
  - Edge coverage
    - Extends branch coverage by considering both explicit and implicit control-flow in Boolean expressions
    - IF (a && b && c) THEN x=5 ELSE x=10; has 6 edges, not 2

Model of Coverage-based Testing



Base Programs

- 7 moderate sized programs

Program	LOC	Executable		Description
		Edges	DUs	
replace	512	191	664	pattern replace
tcas	141	46	57	altitude separation
usl.123	472	97	268	lexical analyzer
usl.128	399	159	240	lexical analyzer
schedule1	292	62	294	priority scheduler
schedule2	301	80	217	priority scheduler
tot_info	440	83	292	information measure

How Do We Proceed?

- Generate test cases according to criteria
  - How many test cases?
    - Say we decide on a number N
  - What coverage?
    - Say 100%
- Execute them on the programs
  - How to detect faults?
  - What if no faults are found?
- Discussion

Fault Space

- Seed faults in the programs
- Ideal world
  - Real faults that have been recorded in the course of development of production software
- Real world
  - Seeded "realistic" faults
    - Mostly changes to single line of code
      - Simple mutations or missing code
    - Sometimes multiple changes
  - Requirements on seeded faults
    - Neither too easy nor too difficult to detect

Fault Space

- Why?
  - If too easy then all tests would detect them, irrespective of the coverage
  - If too difficult, then none would detect – no difference in techniques
- Objective measure of "reasonable" fault
  - Too difficult if less than LB test cases detect it
  - Too easy if more than UB test cases detect
- 10 people seeded faults
  - LB = 3; UB = 350
  - 55 were too difficult, 113 were too easy
  - 130 were reasonable; were included in study

Test Oracle

- The original program was assumed to be "correct" and used as an Oracle

Now How Do We Proceed?

- *Generate test cases*
- Execute them on the programs/mutants
- Record the faults detected

- *Any problems with test case generation?*
  - Do two test suites that satisfy a coverage criterion have the same fault detection ability?
- Discussion

Test Pool

- Use 2-3 testers to create a test pool
- Randomly select test cases from this test pool

Creation of Test Pool

- Realistic process
- Create *initial test pool* (ITP)
  - Category-partition method
- Examine coverage; identify missing areas
- Create *additional test pool* (ATP)
- Goal
  - Each exercisable coverage unit is covered by at least 30 test cases
- Run each test case in the pool and record the outcome (fault detected vs. undetected) and the list of edges and DUs exercised

Test Pool Data

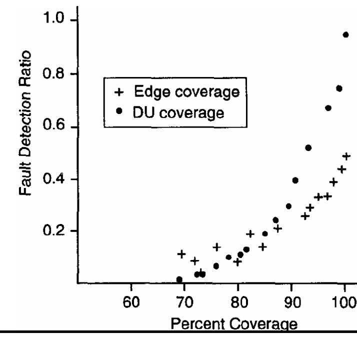
Base Program	Number of faulty versions	Test Pool (TP)			Range of failure ratios in the test pool
		Initial Tests (ITP)	Additional Tests (ATP)	Final Size (ITP + ATP)	
replace	32	79%	21%	5548	.0005-.056
tcas	39	65%	35%	1562	.0006-.084
usl.123	7	99%	1%	4092	.0007-.056
usl.128	10	99%	1%	4076	.0079-.086
schedule1	9	90%	10%	2637	.0027-.100
schedule2	10	77%	23%	2666	.0008-.024
tot_info	23	64%	36%	1067	.0019-.159

- usl.128
  - Test pool size = 4076 cases
  - Hardest fault detected by 32 cases
  - Easiest detected by 350 cases

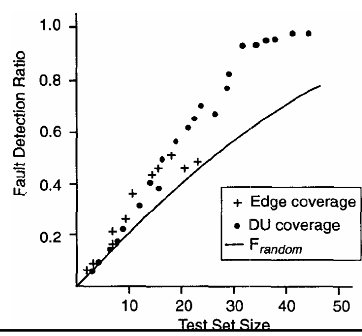
Generating Test Sets

- Goal
  - 5000 test sets for each faulty program
- For each test set of size N
  - Randomly take a test case from pool
  - If it increases coverage, add it
  - Until N tests or 100% coverage
- Sizes
  - Chosen randomly from 1, 2, …,R, where R was determined for each program by trial-and-error as the number slightly larger than the size of the largest test set reaching 100% coverage
- At least 30 tests for each 2% coverage interval

Coverage Graph



Size Graph



Observations

- In general, the performance of both coverage varied widely
- Program classification
  - According to the method that seemed most effective in detecting its faults
  - Define relations
    - DU > Edge
    - Edge > DU
    - DU > Random
    - Edge > Random
    - Random > DU
    - Random > Edge

Better Analysis

- For each faulty program
  - Fit second order, least squares curves
    - Coverage (FC_{DU}, FC_{Edge})
    - and size plots (FS_{DU}, FS_{Edge})
- Definition
  - DU > Edge if
    - $FC_{DU}(100\%) > FC_{Edge}(100\%)$
    - And ($FC_{DU}(100\%) - FC_{Edge}(100\%)$) > (standard deviation of the difference between the measured fault detection ratio and their least squares approximation)

Better Analysis

- $F_{random}(s)$
  - Given a test set size s
  - Probability that a randomly chosen set of s test cases from the test pool contains at least one fault-detecting test case
  - Expected fault-detection ratio of random test sets of size s
- Always computed from TP or ITP
  - Avoids bias in favor of coverage

Better Analysis

- For DU coverage
  - Largest test set generated = d
  - Maximum value of $FS_{DU}(s)$ for s = 1…d = Max_{DU}
- Similarly, For edge coverage
  - Largest test set generated = e
  - Maximum value of $FS_{Edge}(s)$ for s = 1…e = Max_{Edge}
- Definitions
  - DU > Random if $Max_{DU} > F_{random}(d)$
  - Edge > Random if $Max_{Edge} > F_{random}(e)$
    - And differences satisfy a similar property for DU > Edge and Edge > DU
- Similarly, DU < Random if $Max_{DU} < F_{random}(d)$ and Edge < Random if $Max_{Edge} < F_{random}(d)$

Classification of Faults

Class	Characteristics	Number of faults	Fault Detection Ratio at 100% coverage min, avg, max
DU	DU > Edge and DU > Random	31	.19, .67, 1.0
Edge	Edge > DU and Edge > Random	25	.17, .57, .99
DU-&-Edge	DU > Random and Edge > Random and not (DU > Edge or Edge > DU)	32	.14, .59, 1.0
Coverage Total	DU > Random or Edge > Random	88	-
Non-Coverage	DU < Random and Edge < Random	9	-
Other	cannot classify	9	-
	Detection ratios were very low	24	
		130	

DU coverage vs. Random

% DU Coverage	91-93%	93-95%	95-97%	97-99%	99-100%
average size of DU coverage test sets	7.9	9.1	11.3	14.2	17.4
average fault detection ratio of DU coverage test sets	.20	.25	.33	.42	.51
average % superiority in fault detection of DU coverage test sets over same size random test sets	1%	14%	33%	52%	68%
average % increase in the size of random test sets required to yield the same fault detection as the DU coverage test sets	*	21%	46%	79%	160%

* The observed difference is not statistically significant (less than 95% confidence).

Edge Coverage vs. Random

% Edge Coverage	91-93%	93-95%	95-97%	97-99%	99-100%
average size of Edge coverage test sets	7.6	8.5	9.7	11.2	12.6
average fault detection ratio of Edge coverage test sets	.28	.31	.35	.41	.46
average % superiority in fault detection of Edge coverage test sets over same size random test sets	40%	48%	50%	68%	75%
average % increase in the size of random test sets required to yield the same fault detection as the Edge coverage test sets	51%	64%	77%	112%	163%

DU Coverage vs. Edge Coverage

% Coverage	95-97%	97-99%	99-100%
average % difference in size of DU coverage test sets over Edge coverage test sets	1%	9%	21%
average % difference in fault detection of DU coverage test sets over Edge coverage test sets	*	*	38%

*The observed difference is not statistically significant (less than 95% confidence).