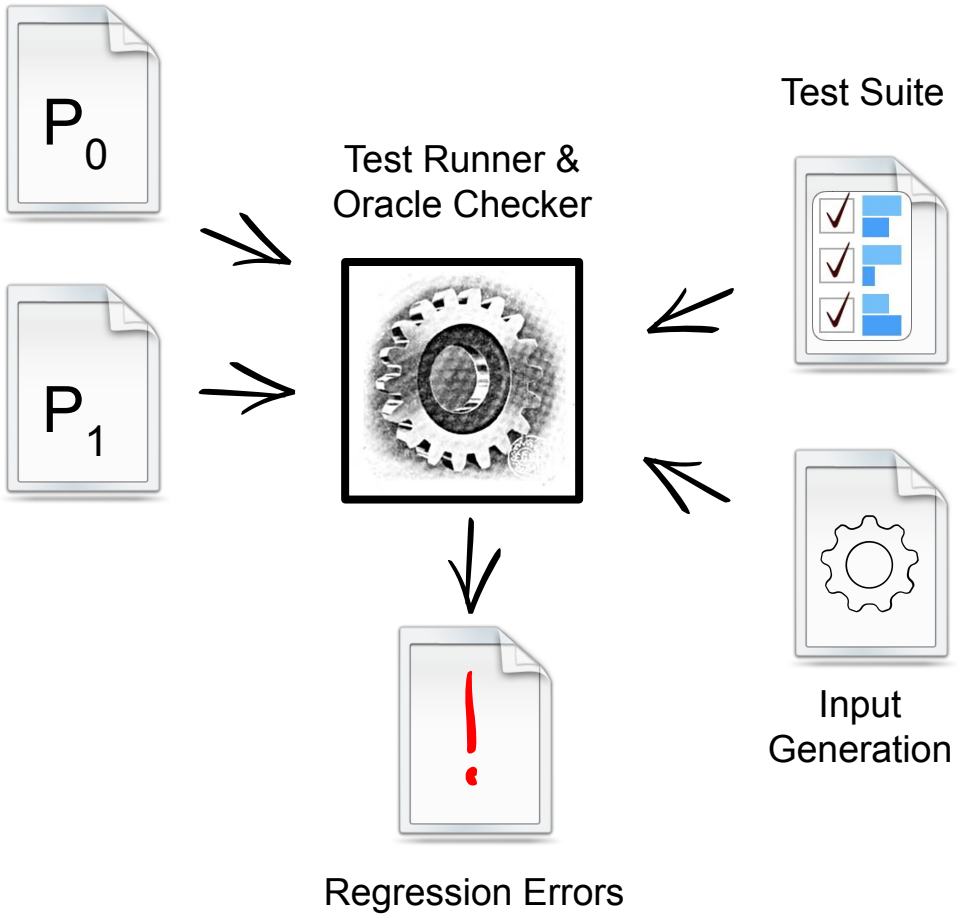
Extending KLEE to Support Behavioral Regression Testing

Richard Rutledge, Alessandro Orso

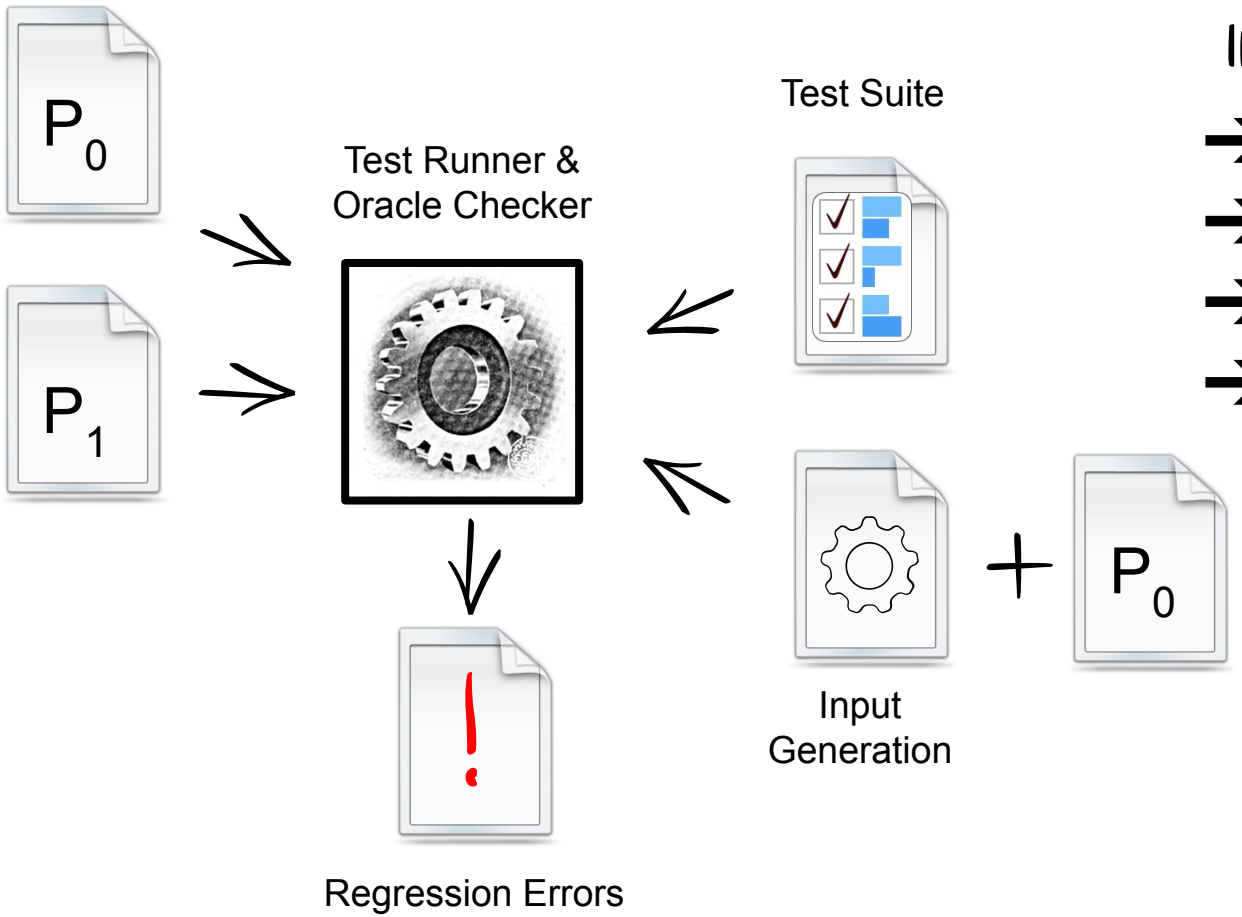
Regression Testing



Issues with regression test suites:

- ➔ Focus on core behavior
- ➔ Provide haphazard coverage
- ➔ Use approximated oracles
- ➔ Sometimes not present at all

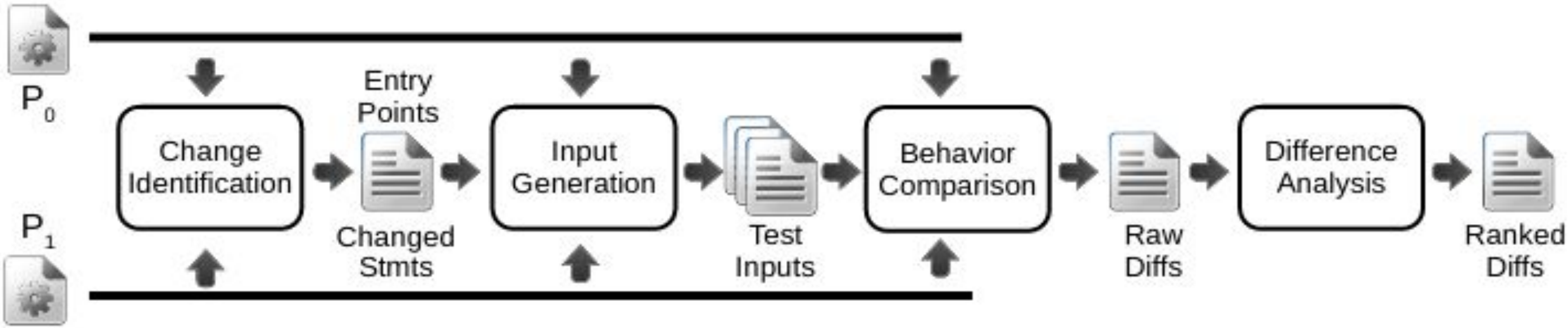
Intuition



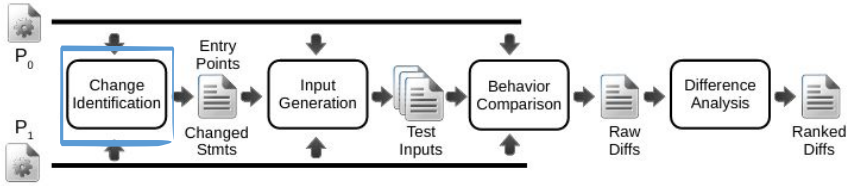
Input Generation + P_0 as oracle:

- ➔ focus on changed code
- ➔ thorough coverage
- ➔ no need for oracles
- ➔ fast enough for CI

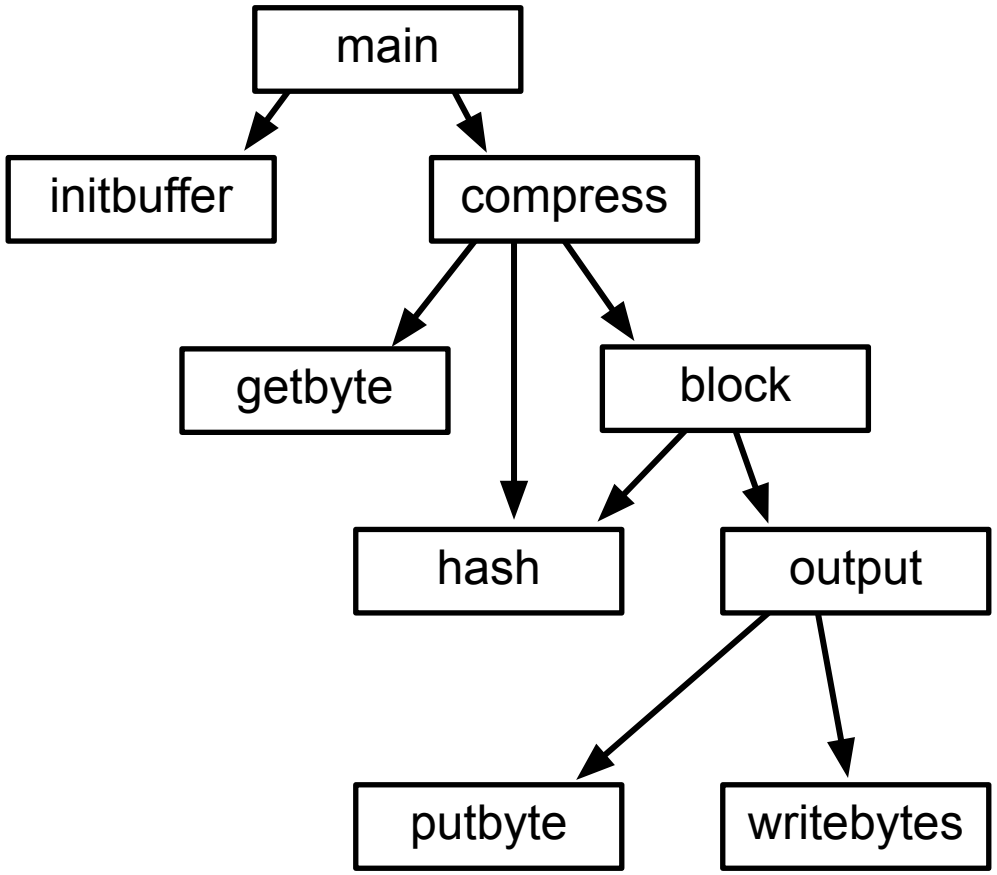
BRT-KLEE Overview



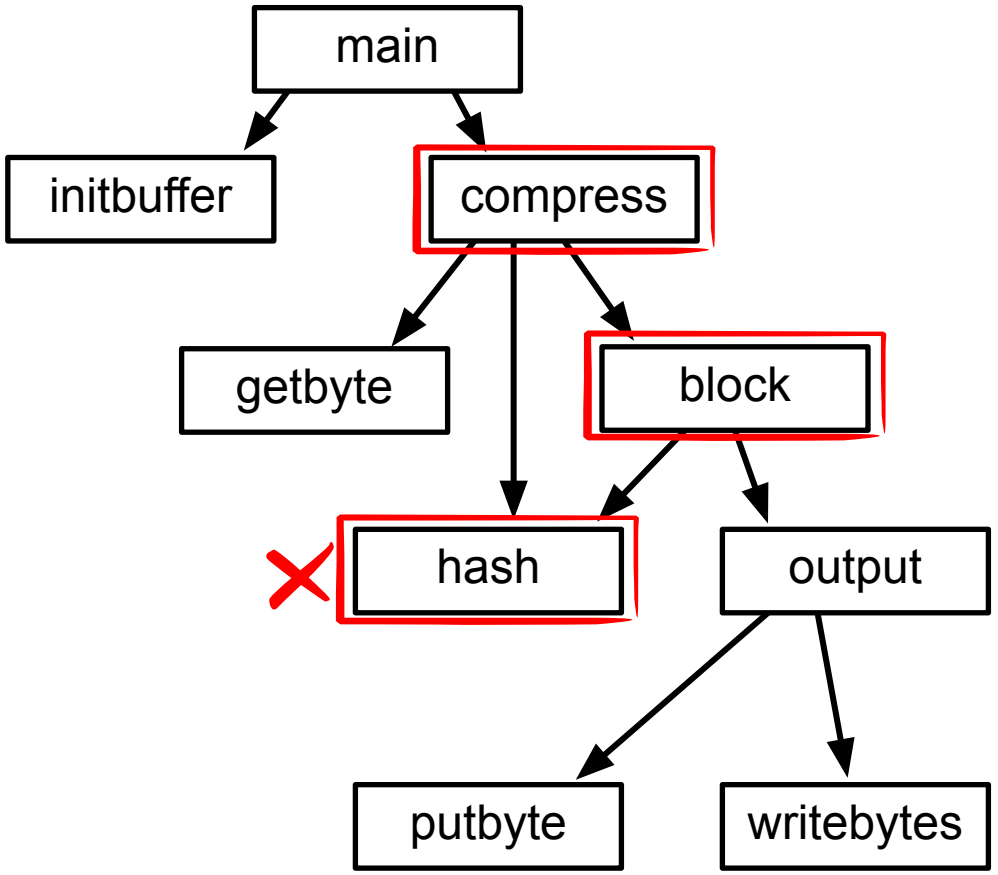
Change Identification



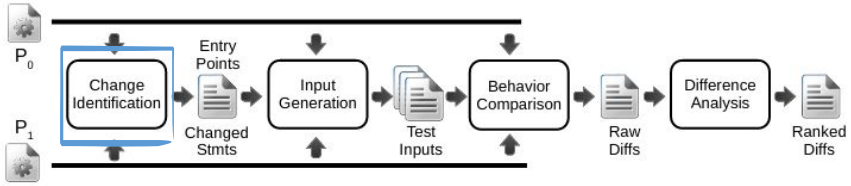
P_0

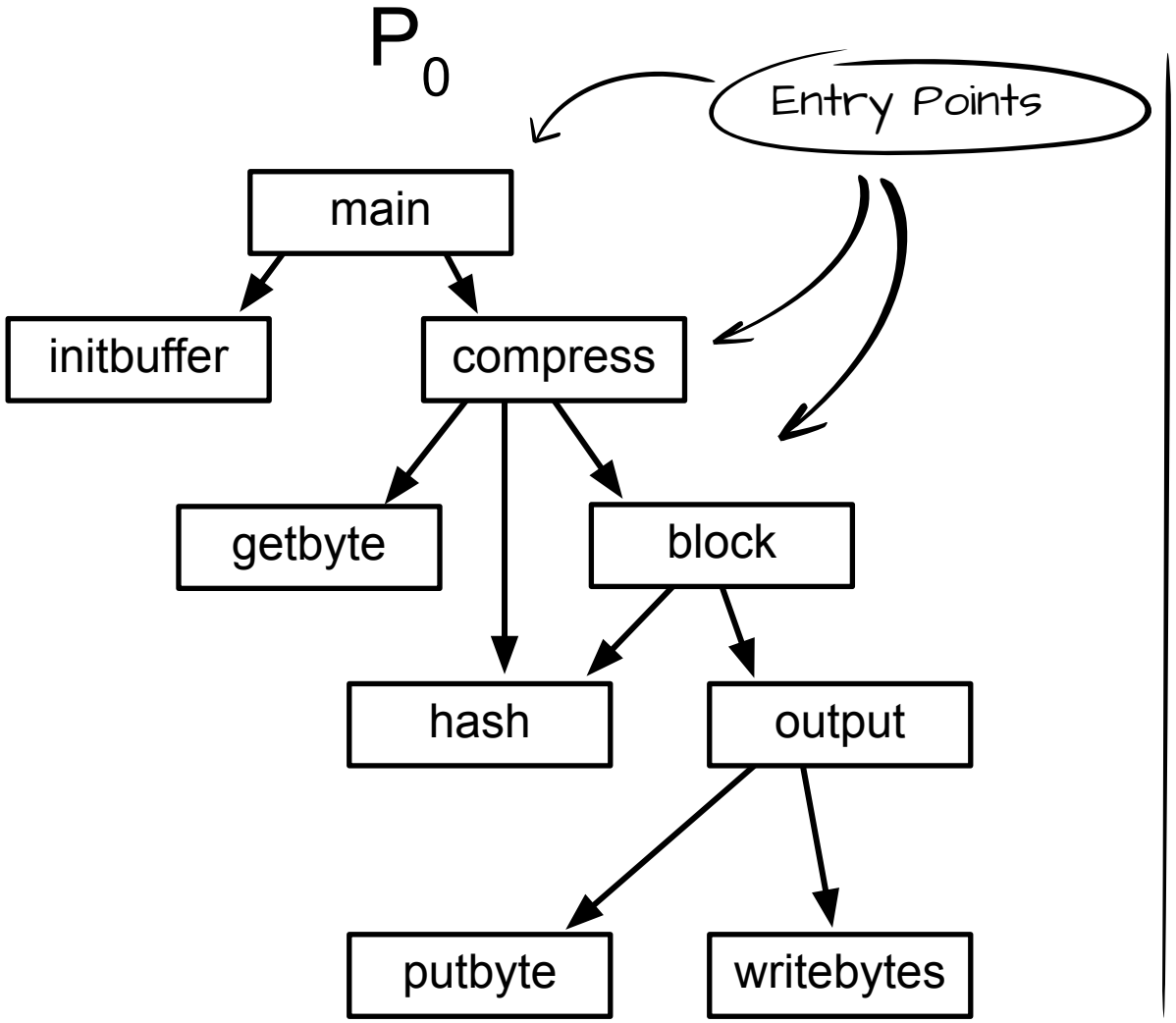


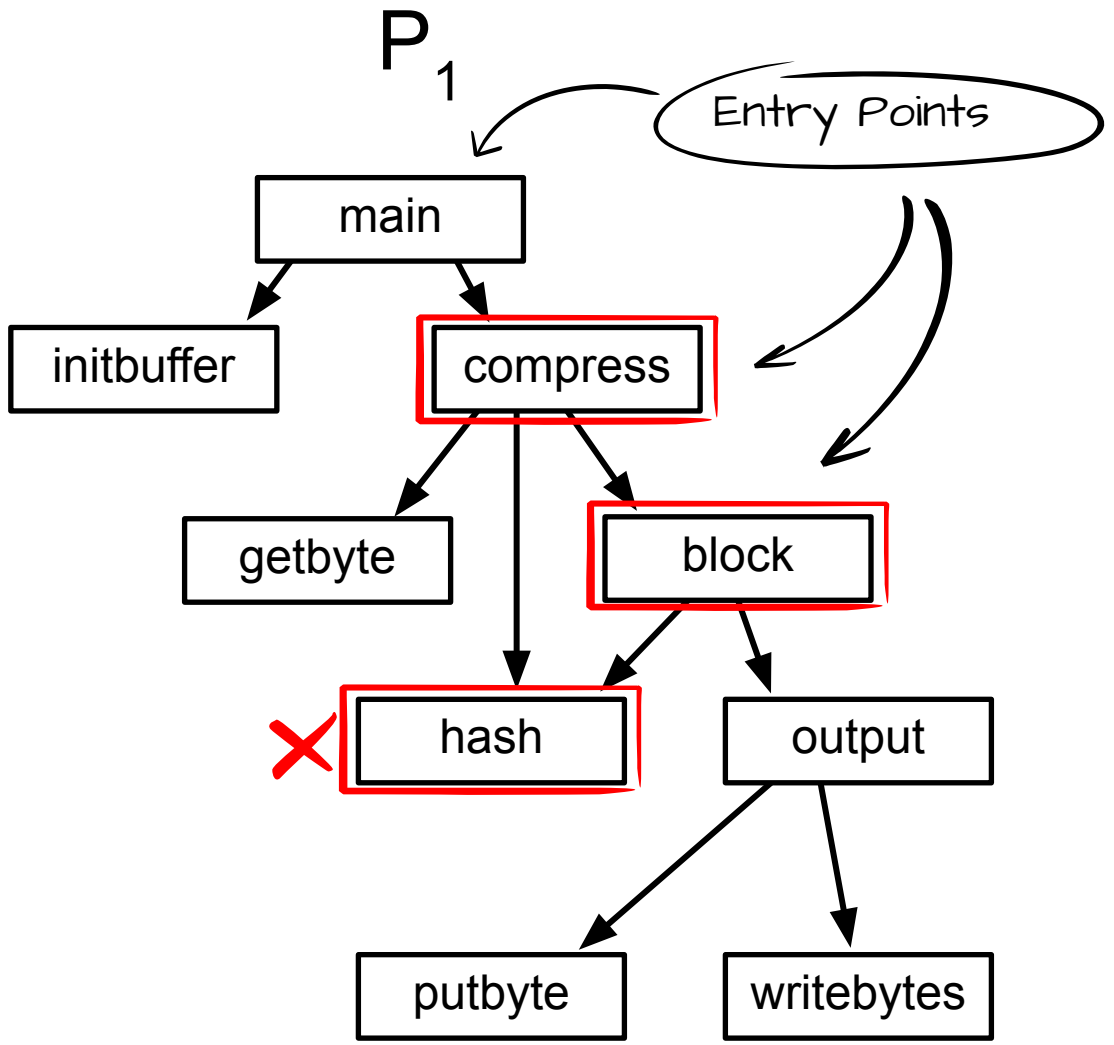
P_1



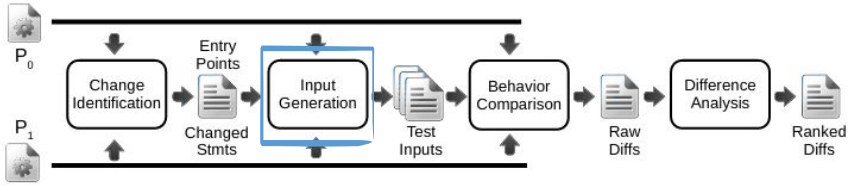
Change Identification

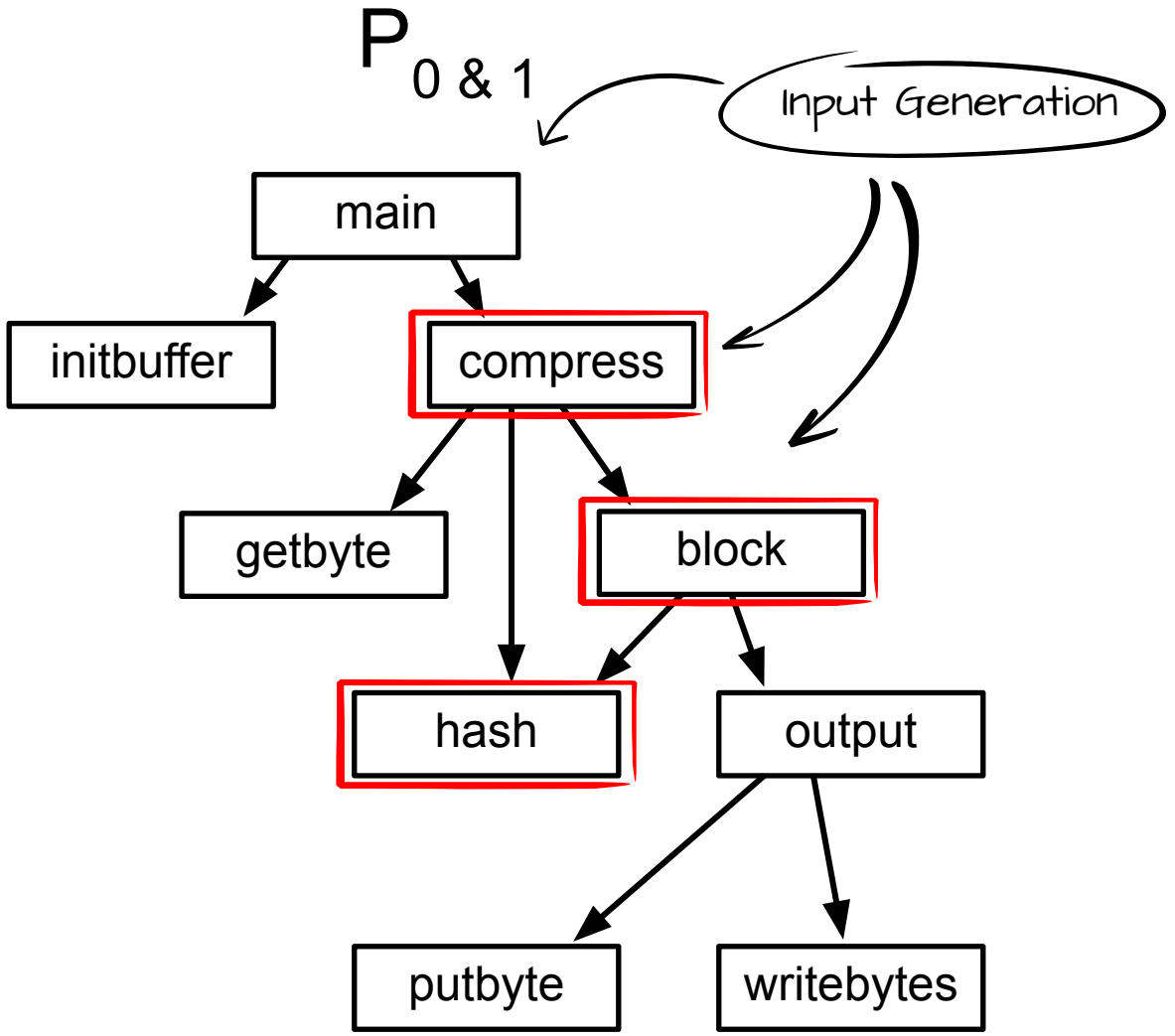




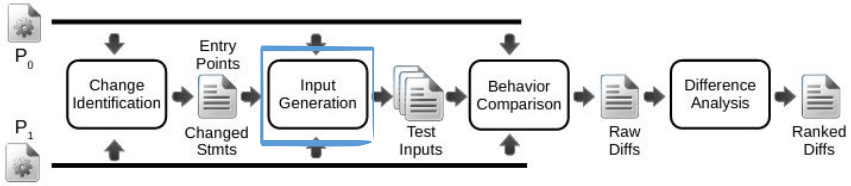


Input Generation

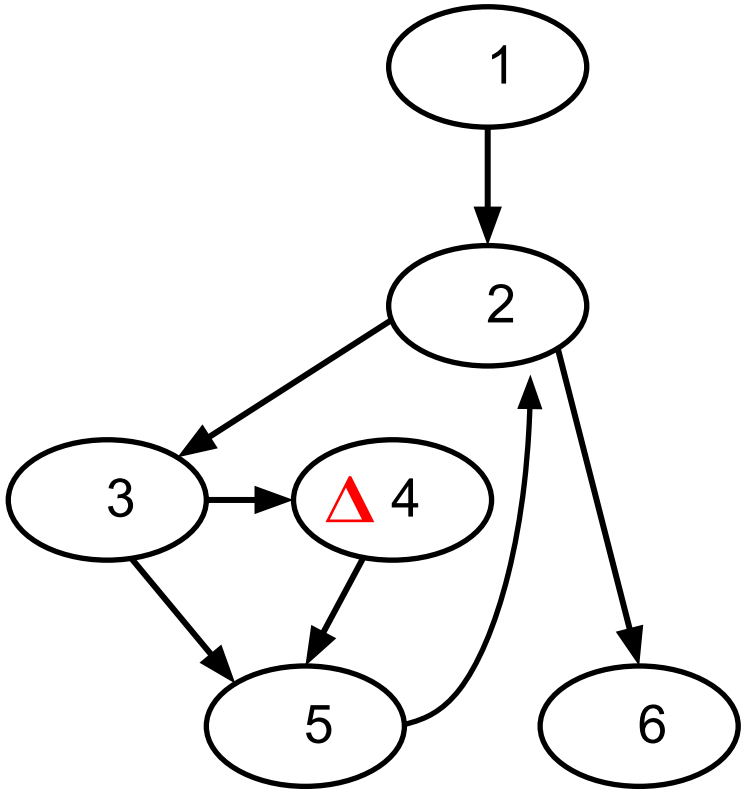




Input Generation



`block` Control Flow Graph (CFG)



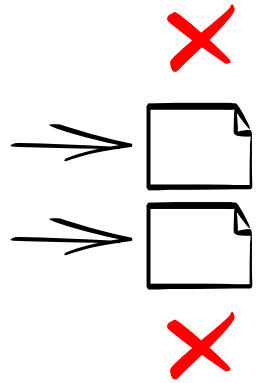
`Input paths:`

1, 2, 3, 5, 2, 6

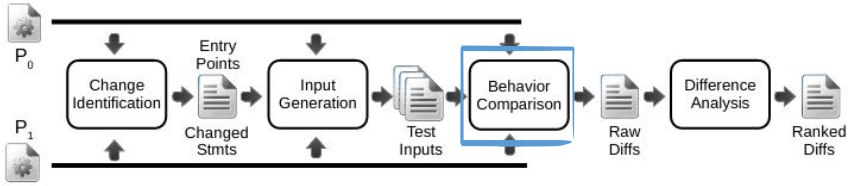
1, 2, 3, 4, 5, 2, 6

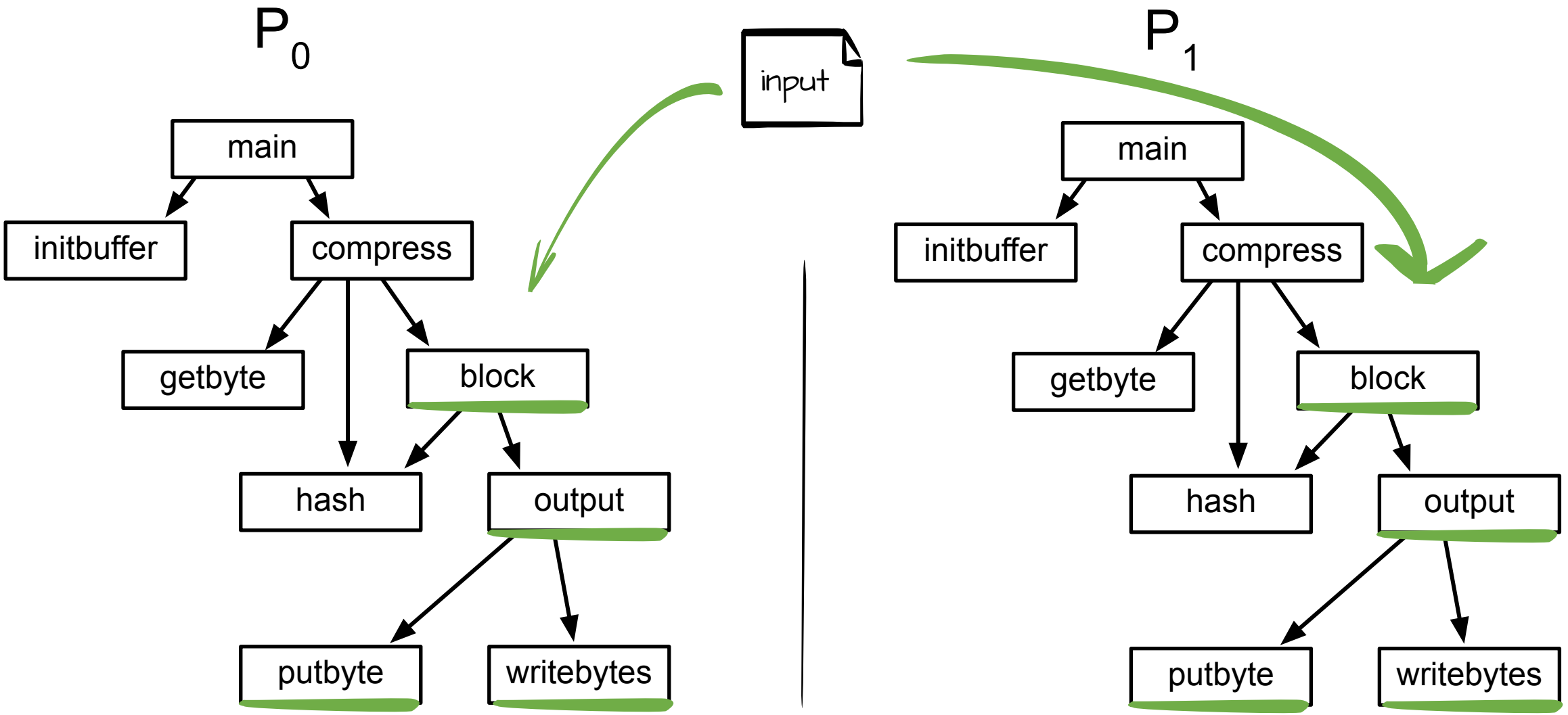
1, 2, 3, 5, 2, 3, 4, 5, 2, 6

1, 2, 6

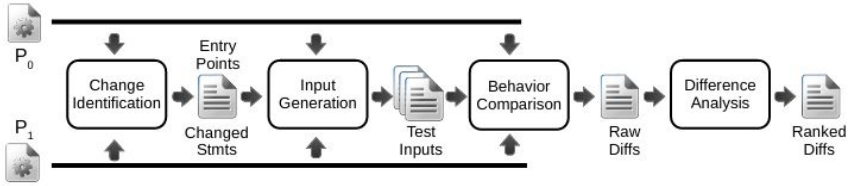


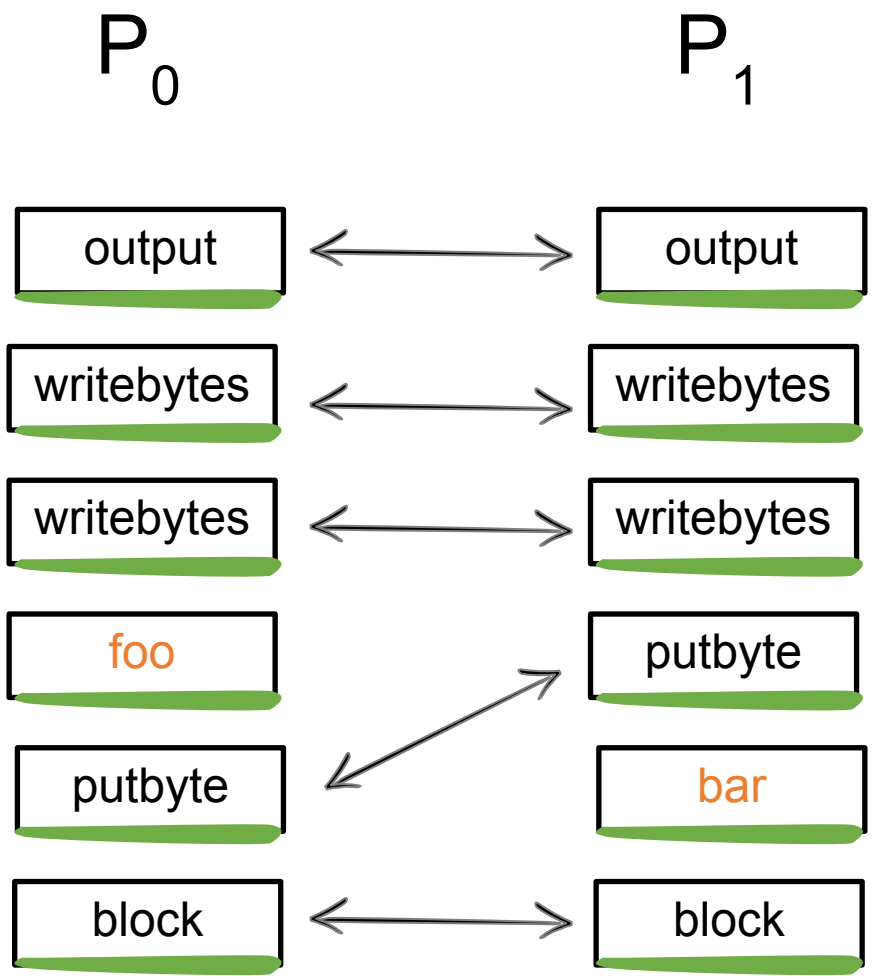
Behavior Comparison





BRT-KLEE Walk-through

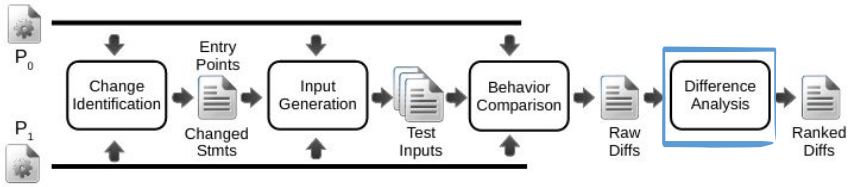




Address space elements compared:

- abnormal termination
- returning function value
- global variables
- output streams

Difference Analysis



- Group differences by program element
- Order dependent differences based on co-occurrence
- Rank differences by distance from changed code on the call graph

Evaluation: Implementation and Research Questions

Implementation:

- ➔ Program analysis and differencing: clang & llvm
- ➔ Symbolic execution engine: forked from KLEE 1.3

Research Questions:

- ➔ RQ1: Can BRT-KLEE detect and effectively rank regressions?
- ➔ RQ2: How do BRT-KLEE's overapproximating results compare to a similar tool's (Shadow's) underapproximating results?
- ➔ RQ3:How does BRT-KLEE perform on mostly refactored code?

Evaluation: Setup and Benchmarks

RQ1:

➔ CoREBench: coreutils, find, and grep
➔ Evaluated with bug oracles

RQ2:

➔ CoREBench: coreutils
(Shadow published results)

program	regressions identifiers	LOC
rm	1	1044
cut	3, 6, 12, 17, 21	519
tail	4, 5, 16	1039
seq	7, 8, 9, 18, 19, 20	254
cp	10	2498
ls	13, 14	3106
du	15	624
expr	22	583
find	23 - 37	8,738
grep	38 - 52	6,153
make	53 - 70	23,805
redis	N/A	121,989

Results

bench mark	regression detection inputs	diffs	+o	PPV	-o	FDR	rank	cmp bklee	cmp shdw
01-rm	671	0	0	-	0	-	N/A	✗	✗
03-cut	30641	5	0	0.0%	5	100.0%	N/A	✗	✗
04-tail	11407	1	1	100.0%	0	0.0%	1	✓	✗
05-tail	83								
06-cut	31								
07-seq	134								
08-seq	140								
09-seq	152								
10-cp	4239	0	0	-	0	-	N/A	✗	✓
12-cut	28606	7	3	42.9%	4	57.1%	3	✓	✓[2]
13-ls	13062	3	2	66.7%	1	33.3%	1	✓	✓
14-ls	10186	10	10	100.0%	0	0.0%	1	✓	✗
15-du	1402	10	8	80.0%	2	20.0%	1	✓	✗
16-tail	8296	1	0	0.0%	1	100.0%	N/A	✗	✓[1]
17-cut	28573	7	3	42.9%	4	57.1%	3	✓	✓[2]
18-seq	15248	2	2	100.0%	0	0.0%	1	✓	✗
19-seq	8533	3	1	33.3%	2	66.7%	3	✓	✗
20-seq	15250	2	2	100.0%	0	0.0%	1	✓	✗
21-cut	18841	11	11	100.0%	0	0.0%	1	✓	✓
22-expr	2644	1	1	100.0%	0	0.0%	1	✓	✗

- Automatically identified >50% known regressions
- Reported higher ranked FP in only 18/43 cases

bench mark	regression detection inputs	diffs	+o	PPV	-o	FDR	rank	bklee
23-find	2552	67	1	1.5%	66	98.5%	67	✓
24-find	22994	3	0	0.0%	3	100.0%	N/A	✗
26-find	180975	65	10	15.4%	55	84.6%	1	✓
27-find	7771	1	0	0.0%	1	100.0%	N/A	✗
28-find	89420	4	0	0.0%	4	100.0%	N/A	✗
30-find	35945	7	7	100.0%	0	0.0%	1	✓
						84.4%	1	✓
						77.8%	1	✓
						77.8%	1	✓
						00.0%	N/A	✗
						100.0%	N/A	✗
37-find	89012	2	2	100.0%	0	0.0%	1	✓
38-grep	4583	8	5	62.5%	3	37.5%	2	✓
41-grep	27704	51	0	0.0%	51	100.0%	N/A	✗
42-grep	2965	15	13	86.7%	2	13.3%	1	✓
44-grep	586	0	0	-	0	-	N/A	✗
45-grep	3142	1	0	0.0%	1	100.0%	N/A	✗
46-grep	9069	5	3	60.0%	2	40.0%	2	✓
47-grep	25758	22	0	0.0%	22	100.0%	N/A	✗
48-grep	25918	16	0	0.0%	16	100.0%	N/A	✗
49-grep	58	0	0	-	0	-	N/A	✗
51-grep	168	13	0	0.0%	13	100.0%	N/A	✗
52-grep	2012	3	3	100.0%	0	0.0%	1	✓

Results

bench mark	regression detection inputs	diffs	+o	PPV	-o	FDR	rank	cmp bklee	cmp shdw
01-rm	671	0	0	-	0	-	N/A	✗	✗
03-cut	30641	5	0	0.0%	5	100.0%	N/A	✗	✗
04-tail	11407	1	1	100.0%					
05-tail	8311	1	0	0.0%					
06-cut	3198	5	2	40.0%					
07-seq	13427	2	1	50.0%					
08-seq	14088	3	1	33.3%					
09-seq	15248	2	2	100.0%	0	0.0%	1	✗	✗
10-cp	4239	0	0	-	0	-	N/A	✗	✓
12-cut	28606	7	3	42.9%	4	57.1%	3	✓	✓[2]
13-ls	13062	3	2	66.7%	1	33.3%	1	✓	✓
14-ls	10186	10	10	100.0%	0	0.0%	1	✓	✗
15-du	1402	10	8	80.0%	2	20.0%	1	✓	✗
16-tail	8296	1	0	0.0%	1	100.0%	N/A	✗	✓[1]
17-cut	28573	7	3	42.9%	4	57.1%	3	✓	✓[2]
18-seq	15248	2	2	100.0%	0	0.0%	1	✓	✗
19-seq	8533	3	1	33.3%	2	66.7%	3	✓	✗
20-seq	15250	2	2	100.0%	0	0.0%	1	✓	✗
21-cut	18841	11	11	100.0%	0	0.0%	1	✓	✓
22-expr	2644	1	1	100.0%	0	0.0%	1	✓	✗

Outperformed state-of-the-art technique used as a baseline.

bench mark	regression detection inputs	diffs	+o	PPV	-o	FDR	rank	bklee
23-find	2552	67	1	1.5%	66	98.5%	67	✓
24-find	22994	3	0	0.0%	3	100.0%	N/A	✗
26-find	180975	65	10	15.4%	55	84.6%	1	✓
27-find	7771	1	0	0.0%	1	100.0%	N/A	✗
28-find	89420	4	0	0.0%	4	100.0%	N/A	✗
				100.0%	0	0.0%	1	✓
				15.6%	54	84.4%	1	✓
				22.2%	7	77.8%	1	✓
				22.2%	7	77.8%	1	✓
				0.0%	7	100.0%	N/A	✗
36-find	68074	4	0	0.0%	4	100.0%	N/A	✗
37-find	89012	2	2	100.0%	0	0.0%	1	✓
38-grep	4583	8	5	62.5%	3	37.5%	2	✓
41-grep	27704	51	0	0.0%	51	100.0%	N/A	✗
42-grep	2965	15	13	86.7%	2	13.3%	1	✓
44-grep	586	0	0	-	0	-	N/A	✗
45-grep	3142	1	0	0.0%	1	100.0%	N/A	✗
46-grep	9069	5	3	60.0%	2	40.0%	2	✓
47-grep	25758	22	0	0.0%	22	100.0%	N/A	✗
48-grep	25918	16	0	0.0%	16	100.0%	N/A	✗
49-grep	58	0	0	-	0	-	N/A	✗
51-grep	168	13	0	0.0%	13	100.0%	N/A	✗
52-grep	2012	3	3	100.0%	0	0.0%	1	✓


Tool Demonstration