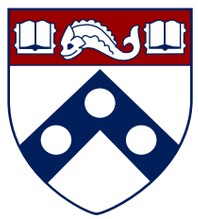
Program Transformation for Reducing Software Complexity

Kihong Heo
(joint work with Woosuk Lee, Pardis Pashakhanloo, Mayur Naik)
University of Pennsylvania



Jul 9 2018 @ Korea University

Modern Software

Modern Software

framework

one-size-fit-all design

add-on

plugin

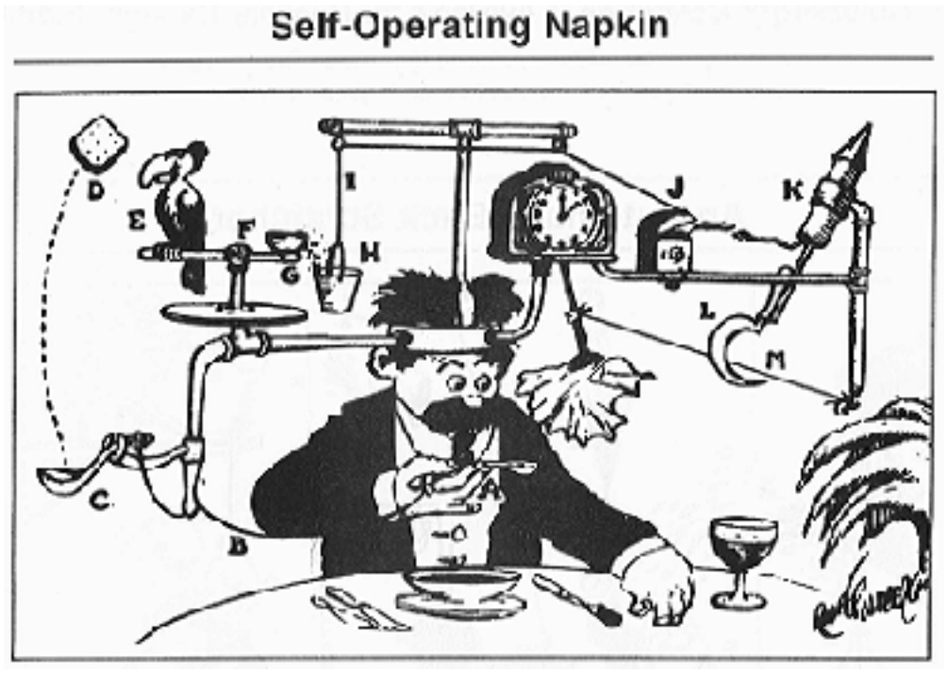
library

virtual machine

OOP

extension

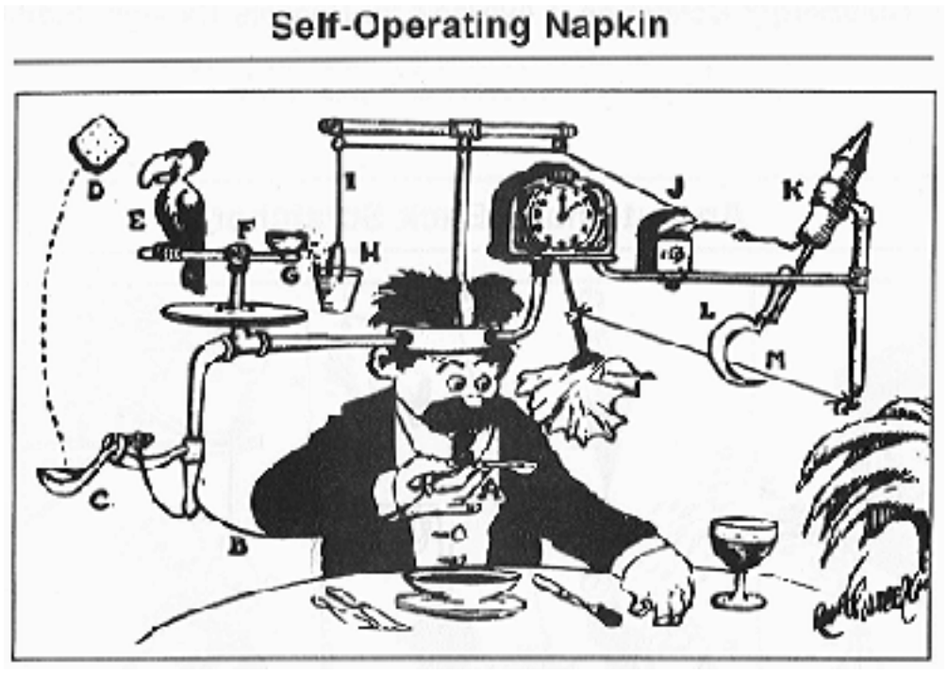
Bloatware



“A stock brokerage benchmark executes 268 method calls and creates 70 new objects just to move a single date field from XML to Java.”

— IBM Research Report, 2005

Bloatware



“A stock brokerage benchmark executes 268 method calls and creates 70 new objects just to move a single date field from XML to Java.”

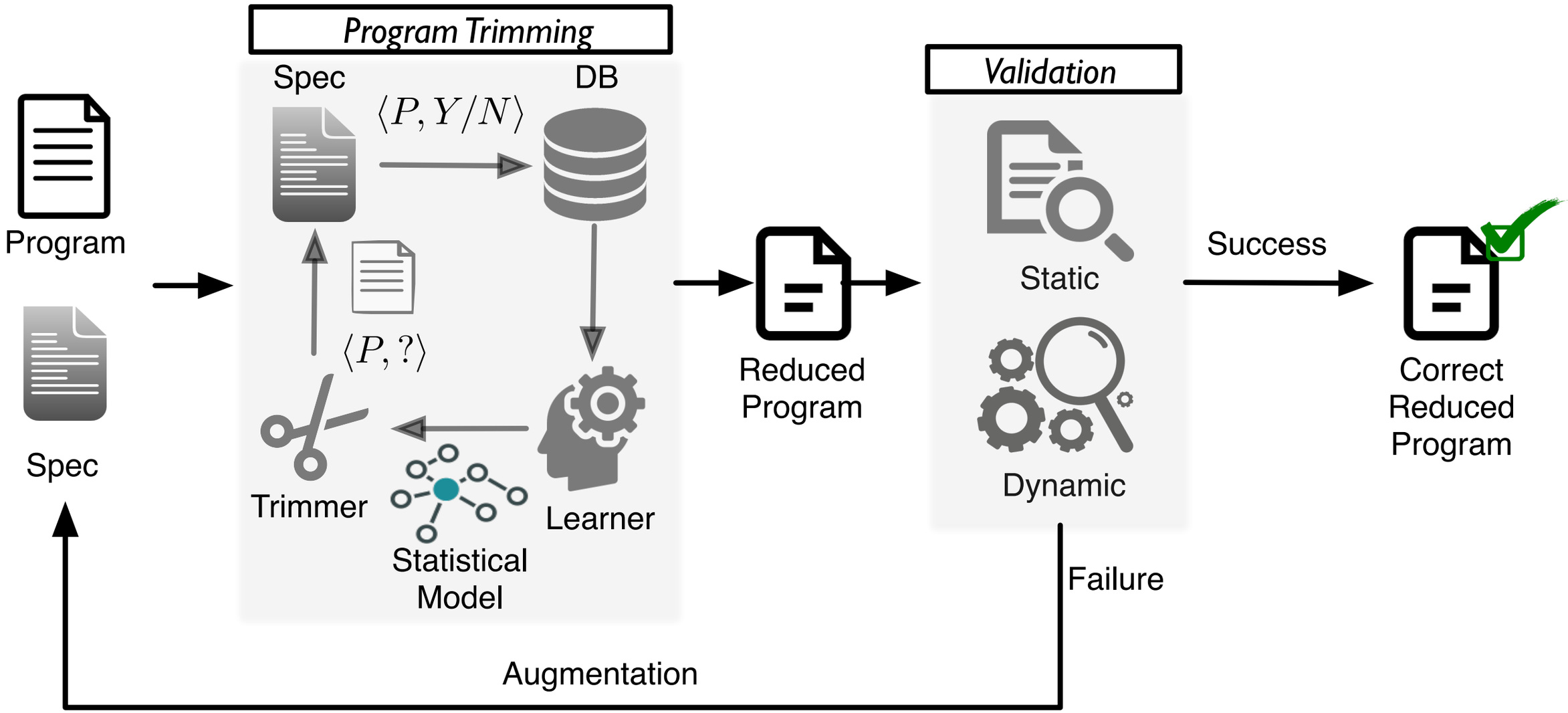
— IBM Research Report, 2005

Degrading performance & Expanding attack surface

Goal

- Automated program debloating system that satisfies
  - **minimality**: trim code as aggressively as possible
  - **efficiency**: scale to large programs
  - **robustness**: avoid introducing new vulnerabilities
  - **naturalness**: produce maintainable code
  - **generality**: handle a variety of programs and specs

CHISEL: Program Reducer Framework



Example Program

- tar-1.14
- 31,605 LOC (11,134 stmts)
- 97 command-line options
- c.f.) 8 options in a lightweight counterpart in BusyBox*

*https://busybox.net

Exploit

- Security vulnerability (path traversal): CVE-2016-6321

```
root:/$ _
```

Exploit

- Security vulnerability (path traversal): CVE-2016-6321

```
root:/$ cat etc/shadow
root:l1k4qj1xQWErkzQW1:0:99999:7:::
root:/$ _
```

Exploit

- Security vulnerability (path traversal): CVE-2016-6321

```
root:/$ cat etc/shadow
root:l1k4qj1xQWErkzQW1:0:99999:7:::
root:/$ tar xv etc/motd malicious.tar
root:/$ _
```

Exploit

- Security vulnerability (path traversal): CVE-2016-6321

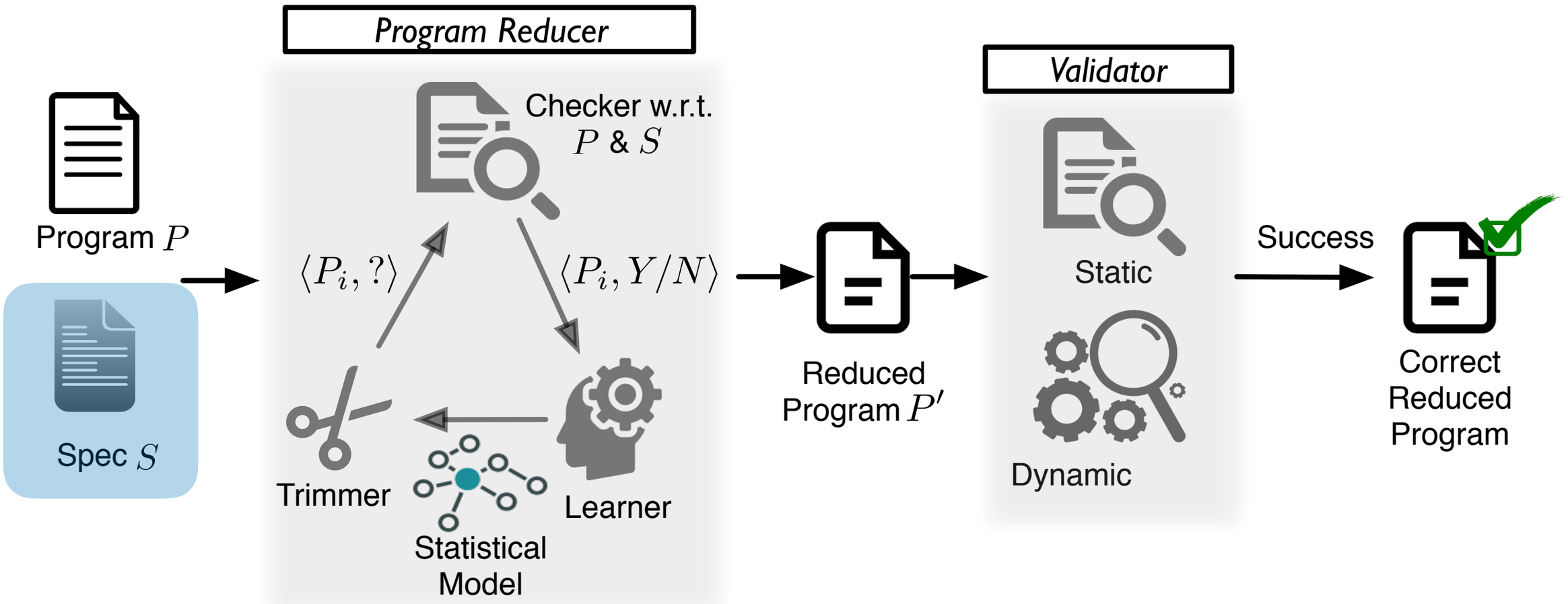
```
root:/$ cat etc/shadow
root:l1k4qj1xQWErkzQW1:0:99999:7:::
root:/$ tar xv etc/motd malicious.tar
root:/$ cat etc/shadow
Your system has been compromised :)
root:/$ _
```

Cause

- Malicious input file

- Incorrect and redundant features
  - incorrectly sanitize destination paths including “..”: “etc/motd/../etc/shadow” -> “etc/shadow”
  - overwrite existing files

CHISEL: Program Reducer Framework



Specification

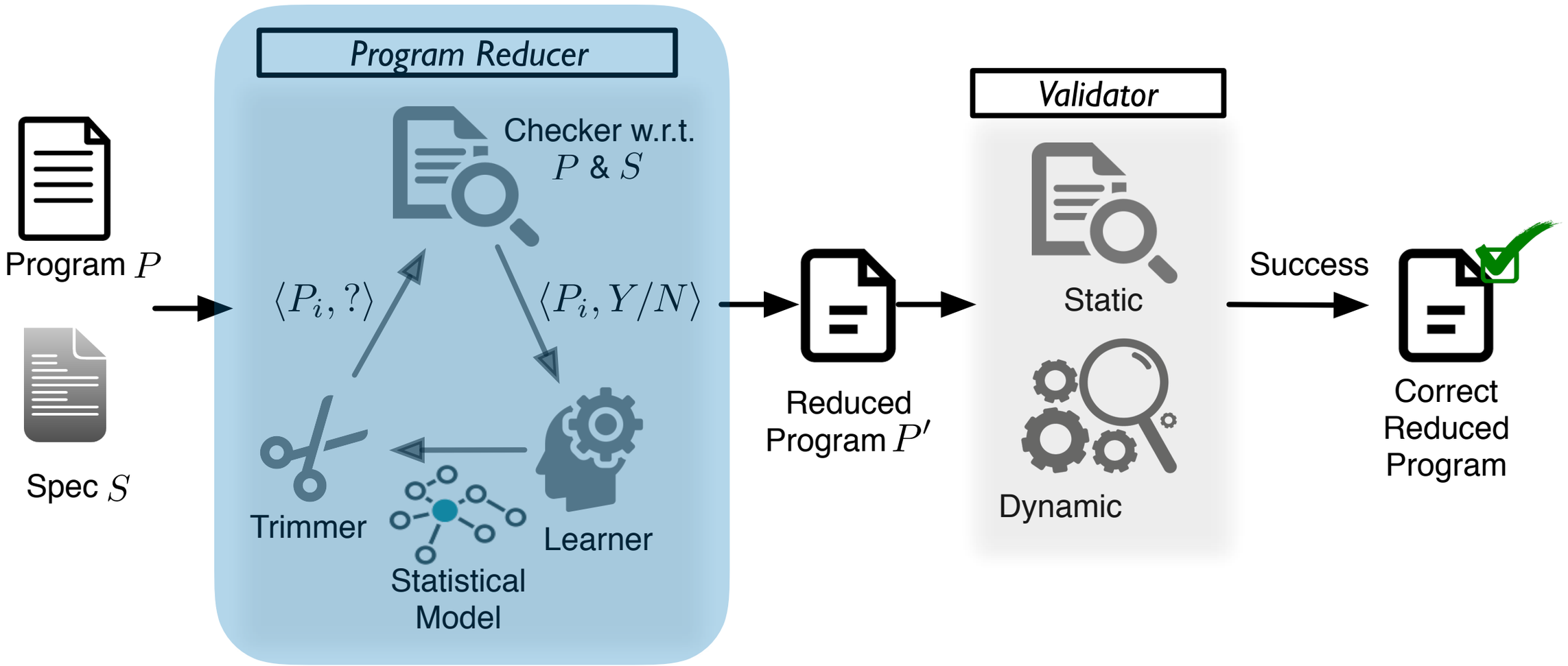
- Given a program *P,* a script that returns true if
  - *P* is compilable
  - *P* preserves “core” functionalities
    - e.g.) 8 options in Busybox
  - *P* does not crash on “non-core” functionalities

Specification

```
#!/bin/bash

function compile {
  clang -o tar.debloat tar-1.14.c
  return $?
}

# tests for the core functionalities
function core {
  # 1. archiving multiple files
  touch foo bar
  ./tar.debloat cf foo.tar foo bar
  rm foo bar
  ./tar.debloat xf foo.tar
  test -f foo -a -f bar || exit 1

  # 2. extracting from stdin
  touch foo
  ./tar.debloat cf foo.tar foo
  rm foo
  cat foo.tar | ./tar.debloat x
  test -f foo || exit 1

  # other tests
  …
  return 0
}
```

```
# tests for the non-core functionalities
function non_core {
  for test_script in `ls other_tests/*.sh`
  do
    { sh -x -e $test_script; } >& log
    grep 'Segmentation fault' log && exit 1
  done
  return 0
}

compile || exit 1
core || exit 1
non_core || exit 1
```

CHISEL: Program Reducer Framework



Reducer

- A variant of delta debugging

Original Ver.
31,605 LOC
11,134 Stmts

Debloated Ver.
3,631 LOC
870 Stmts

```
root:/$ _
```

Reducer

- A variant of delta debugging

Original Ver.
31,605 LOC
11,134 Stmts

Debloated Ver.
3,631 LOC
870 Stmts

```
root:/$ cat etc/shadow
root:l1k4qj1xQWErkzQW1:0:99999:7:::
root:/$ tar xv etc/motd malicious.tar
root:/$ cat etc/shadow
root:l1k4qj1xQWErkzQW1:0:99999:7:::
root:/$ _
```

Reduced Program

```c
int absolute_names;
int ignore_zeros_option;
struct tar_stat_info stat_info;

char *safer_name_suffix (char *file_name, int link_target) {
    int prefix_len;
    char *p;

    if (absolute_names) {
        p = file_name;
    } else {
        prefix_len = 0;
        for (p = file_name + prefix_len; *p; ) {
            /* CVE-2016-6321 if file_name contains ".." */
            ...
            /* Update prefix_len */
        }
    }
    ...
    return p;
}

void extract_archive() {
    char *file_name = safer_name_suffix(stat_info.file_name, 0);
    maybe_recoverable(file_name); /* remove existing files */
    ...
}

void list_archive() { ... }
```

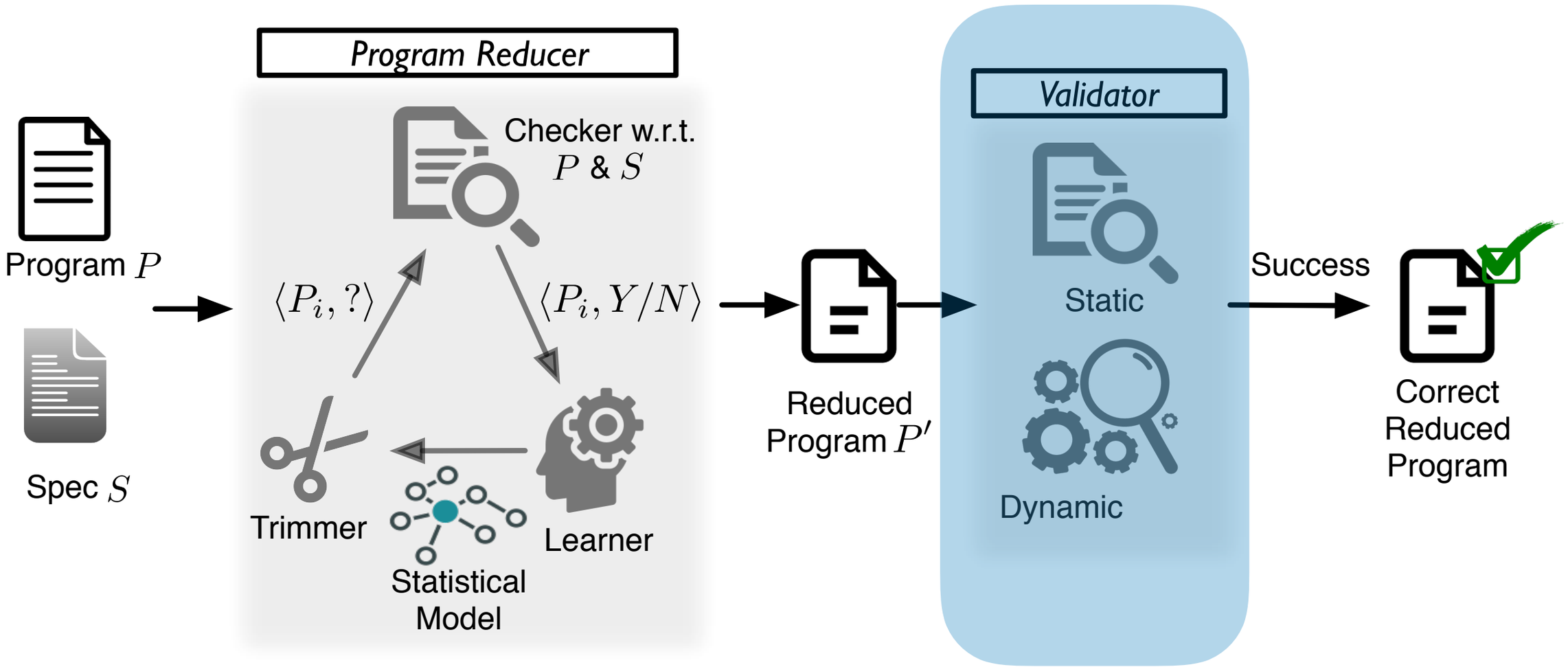
```c
void read_and(void *(do_something)(void)) {
    enum read_header status;
    while (...) {
        status = read_header();
        switch (status) {
        case HEADER_SUCCESS: (*do_something)(); continue;
        case HEADER_ZERO_BLOCK:
          if (ignore_zeros_option) continue;
          else break;
        default: break;
        }
    }
    ...
}

/* Supports all options: -x, -t, -P, -i, ... */
int main(int argc, char **argv) {
    int optchar;
    while (optchar = getopt_long(argc, argv) != -1) {
        switch(optchar) {
        case 'x': read_and(&extract_archive); break;
        case 't': read_and(&list_archive); break;
        case 'P': absolute_names = 1; break;
        case 'i': ignore_zeros_option = 1; break;
        ...
        }
    }
    ...
}
```

Reduced Program

```
/* Chisel: global variable declarations removed */

struct tar_stat_info stat_info;

char *safer_name_suffix (char *file_name, int link_target) {
    /* Chisel: code containing CVE removed */


        p = file_name;


    return p;
}

void extract_archive() {
    char *file_name = safer_name_suffix(stat_info.file_name, 0);
    /* Chisel: code containing CVE removed */
    ...
}

void list_archive() { ... }
```

```
void read_and(void *(do_something)(void)) {
    enum read_header status;
    while (...) {
        status = read_header();
        switch (status) {
        case HEADER_SUCCESS: (*do_something)(); continue;
        /* Chisel: unnecessary functionalities removed */


        default: break;
        }
    }

}

/* Supports only 8 opts: -c, -f, -x, -v, -t, -O, -o, -k */
int main(int argc, char **argv) {
    int optchar;
    while (optchar = getopt_long(argc, argv) != -1) {
        switch(optchar) {
        case 'x': read_and(&extract_archive); break;
        case 't': read_and(&list_archive); break;
        /* Chisel: unsupported options removed */


        }
    }
    ...
}
```

CHISEL: Program Reducer Framework



Validation

- Static validation

  - #Alarms: **1,295** (original) → **23** (debloated) alarms
  - #ROP gadget: **1,340** → **528**
- Dynamic validation
  - no crashing input found in **3 days** using the AFL fuzzer

CHISEL: Reinforcement-learning-guided Program Debloating System

Program Debloating

- Property test function $\mathcal{O}$: takes a program & returns Y / N

- Given a pgm ***P*** s.t. $\mathcal{O}$(***P***), find a **minimal *P'*** s.t. $\mathcal{O}$(***P'***)

- ***P**** is called **1-minimal**:
  any variant of ***P**** by removing a single element (e.g., stmt)
  does not pass the test

Example

- Property of interest: termination with zero

Original

```
int f1() { return 0; }
int f2() { return 1; }
int f3() { return 1; }
int f4() { return 1; }
int f5() { return 1; }
int f6() { return 1; }
int f7() { return 1; }
int main() { return f1(); }
```

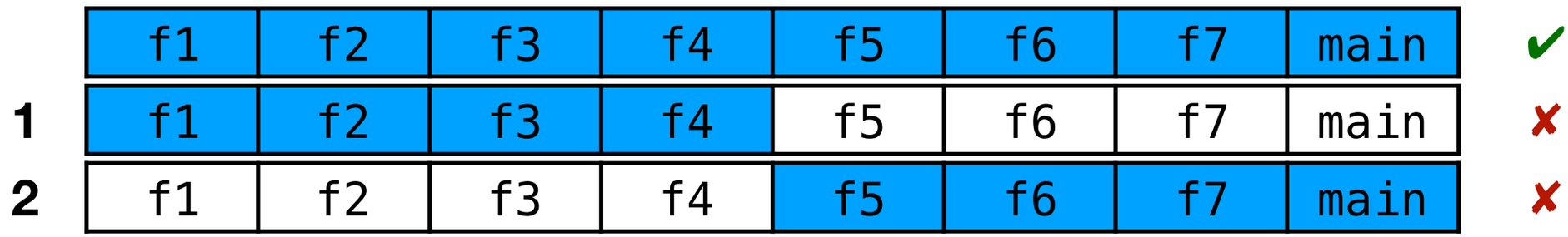
Minimal version

```
int f1() { return 0; }
int main() { return f1(); }
```

Delta Debugging*

- Guarantees 1-minimality
- Proceeds by binary search

(included)



*Andreas Zeller and Ralf Hildebrandt, Simplifying and Isolating Failure-Inducing Input, TSE, 2002

Delta Debugging*

- Guarantees 1-minimality
- Proceeds by binary search

(included)

	f1	f2	f3	f4	f5	f6	f7	main	✔
1	**f1**	**f2**	**f3**	**f4**	f5	f6	f7	main	✘
2	f1	f2	f3	f4	**f5**	**f6**	**f7**	**main**	✘
3	**f1**	**f2**	f3	f4	f5	f6	f7	main	✘
4	f1	f2	**f3**	**f4**	f5	f6	f7	main	✘
5	f1	f2	f3	f4	**f5**	**f6**	f7	main	✘
6	f1	f2	f3	f4	f5	f6	**f7**	**main**	✘

*Andreas Zeller and Ralf Hildebrandt, Simplifying and Isolating Failure-Inducing Input, TSE, 2002

Delta Debugging*

- Guarantees 1-minimality
- Proceeds by binary search

(included)

	f1	f2	f3	f4	f5	f6	f7	main	✔
1	f1	f2	f3	f4				main	✘
2	f1	f2	f3	f4	f5	f6	f7	main	✘
3	f1	f2						main	✘
4	f1	f2	f3	f4				main	✘
5	f1	f2	f3	f4	f5	f6		main	✘
6	f1	f2	f3	f4			f7	main	✘
7	f1	f2	f3	f4	f5	f6	f7	main	✘
8	f1	f2						main	✔

*Andreas Zeller and Ralf Hildebrandt, Simplifying and Isolating Failure-Inducing Input, TSE, 2002

Delta Debugging*

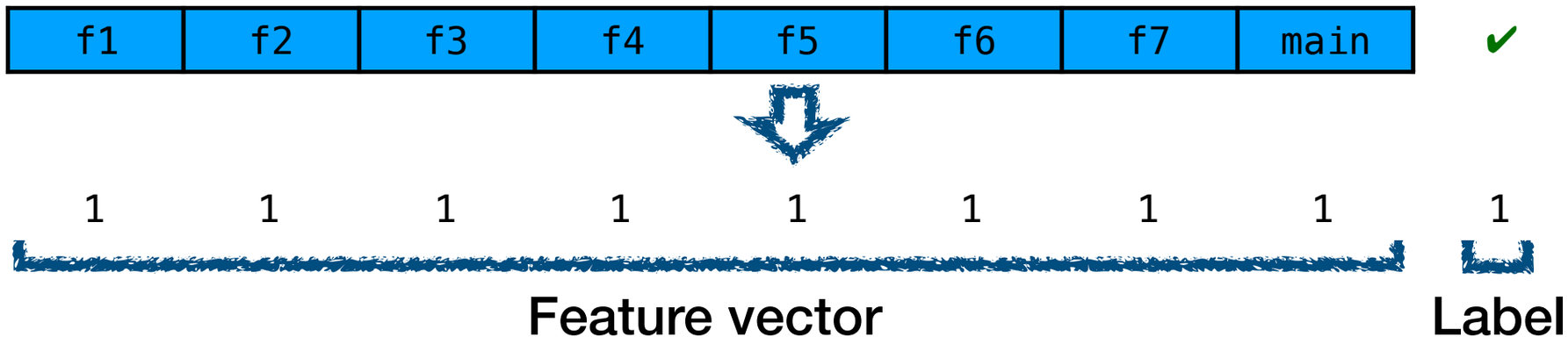
	f1	f2	f3	f4	f5	f6	f7	main	✔
1	f1	f2	f3	f4	f5	f6	f7	main	✘
2	f1	f2	f3	f4	f5	f6	f7	main	✘
3	f1	f2	f3	f4	f5	f6	f7	main	✘
4	f1	f2	f3	f4	f5	f6	f7	main	✘
5	f1	f2	f3	f4	f5	f6	f7	main	✘
6	f1	f2	f3	f4	f5	f6	f7	main	✘
7	f1	f2	f3	f4	f5	f6	f7	main	✘
8	f1	f2	f3	f4	f5	f6	f7	main	✔
9	f1	f2	f3	f4	f5	f6	f7	main	✔
10	f1	f2	f3	f4	f5	f6	f7	main	✘
11	f1	f2	f3	f4	f5	f6	f7	main	✘
12	f1	f2	f3	f4	f5	f6	f7	main	✘
13	f1	f2	f3	f4	f5	f6	f7	main	✘
14	f1	f2	f3	f4	f5	f6	f7	main	✘
15	f1	f2	f3	f4	f5	f6	f7	main	✔
16	f1	f2	f3	f4	f5	f6	f7	main	✔

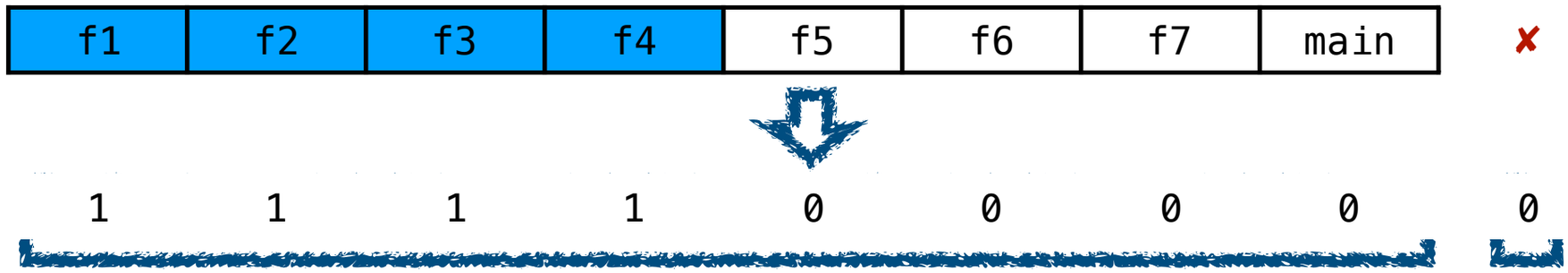
Delta Debugging*

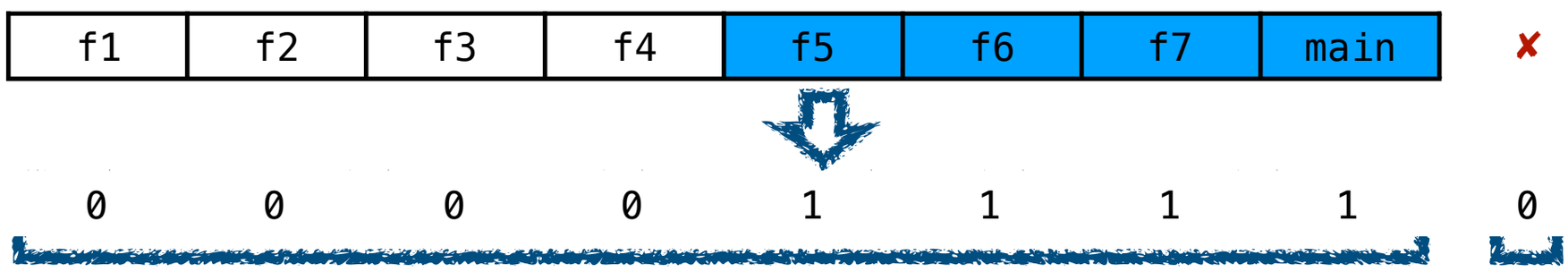
	f1	f2	f3	f4	f5	f6	f7	main	✔
1	f1	f2	f3	f4	f5	f6	f7	main	✘
2	f1	f2	f3	f4	f5	f6	f7	main	✘
3	f1	f2	f3	f4	f5	f6	f7	main	✘
4	f1	f2	f3	f4	f5	f6	f7	main	✘
5	f1	f2	f3	f4	f5	f6	f7	main	✘
6	f1	f2	f3	f4	f5	f6	f7	main	✘
7	f1	f2	f3	f4	f5	f6	f7	main	✘
8	f1	f2	f3	f4	f5	f6	f7	main	✔
9	f1	f2	f3	f4	f5	f6	f7	main	✔
10	f1	f2	f3	f4	f5	f6	f7	main	✘
11	f1	f2	f3	f4	f5	f6	f7	main	✘
12	f1	f2	f3	f4	f5	f6	f7	main	✘
13	f1	f2	f3	f4	f5	f6	f7	main	✘
14	f1	f2	f3	f4	f5	f6	f7	main	✘
15	f1	f2	f3	f4	f5	f6	f7	main	✔
16	f1	f2	f3	f4	f5	f6	f7	main	✔

Can we learn useful knowledge to guide the search from trial and error?

Training Data

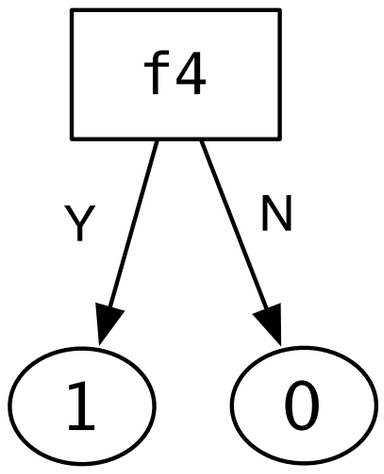






Guided Search

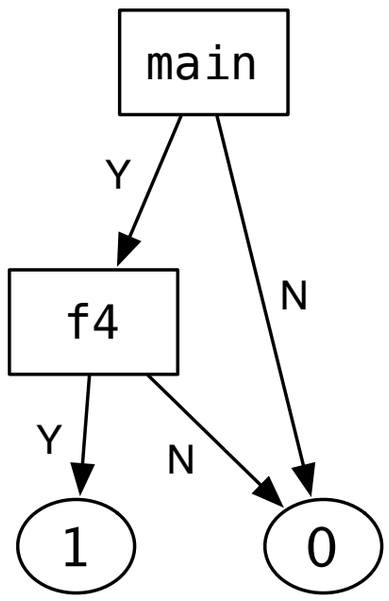
	f1	f2	f3	f4	f5	f6	f7	main
1	f1	f2	f3	f4	f5	f6	f7	main



Chunks including f4 will pass the test.

Guided Search

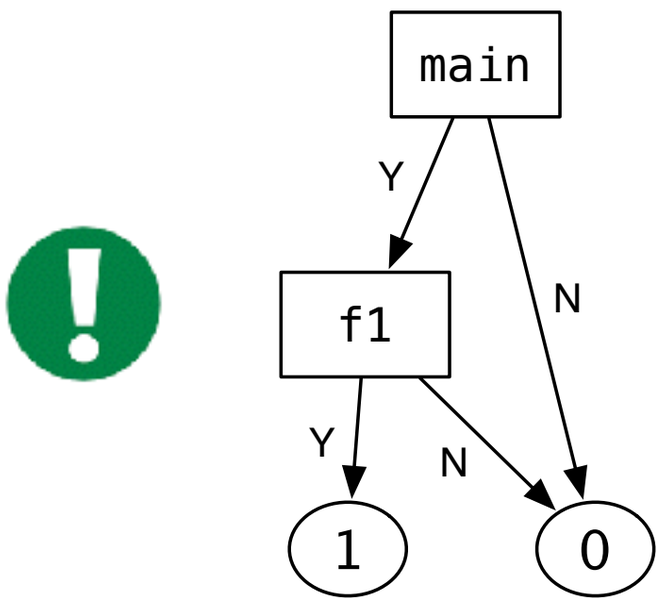
	f1	f2	f3	f4	f5	f6	f7	main	✔
1	f1	f2	f3	f4	f5	f6	f7	main	✘
2	f1	f2	f3	f4	f5	f6	f7	main	✘



Chunks including `f4` and `main` will pass the test.

Guided Search

	f1	f2	f3	f4	f5	f6	f7	main	✔
1	f1	f2	f3	f4	f5	f6	f7	main	✘
2	f1	f2	f3	f4	f5	f6	f7	main	✘
3	f1	f2	f3	f4	f5	f6	f7	main	✘
4	f1	f2	f3	f4	f5	f6	f7	main	✔
5	f1	f2	f3	f4	f5	f6	f7	main	✔
6	f1	f2	f3	f4	f5	f6	f7	main	✘



Chunks including `f1` and `main` will pass the test.

Guided Search

	f1	f2	f3	f4	f5	f6	f7	main	✔
1	f1	f2	f3	f4	f5	f6	f7	main	✘
2	f1	f2	f3	f4	f5	f6	f7	main	✘
3	f1	f2	f3	f4	f5	f6	f7	main	✘
4	f1	f2	f3	f4	f5	f6	f7	main	✔
5	f1	f2	f3	f4	f5	f6	f7	main	✔
6	f1	f2	f3	f4	f5	f6	f7	main	✘
7	f1	f2	f3	f4	f5	f6	f7	main	✔
8	f1	f2	f3	f4	f5	f6	f7	main	✔
9	f1	f2	f3	f4	f5	f6	f7	main	✘
10	f1	f2	f3	f4	f5	f6	f7	main	✘

Unguided Search
16 iters

vs

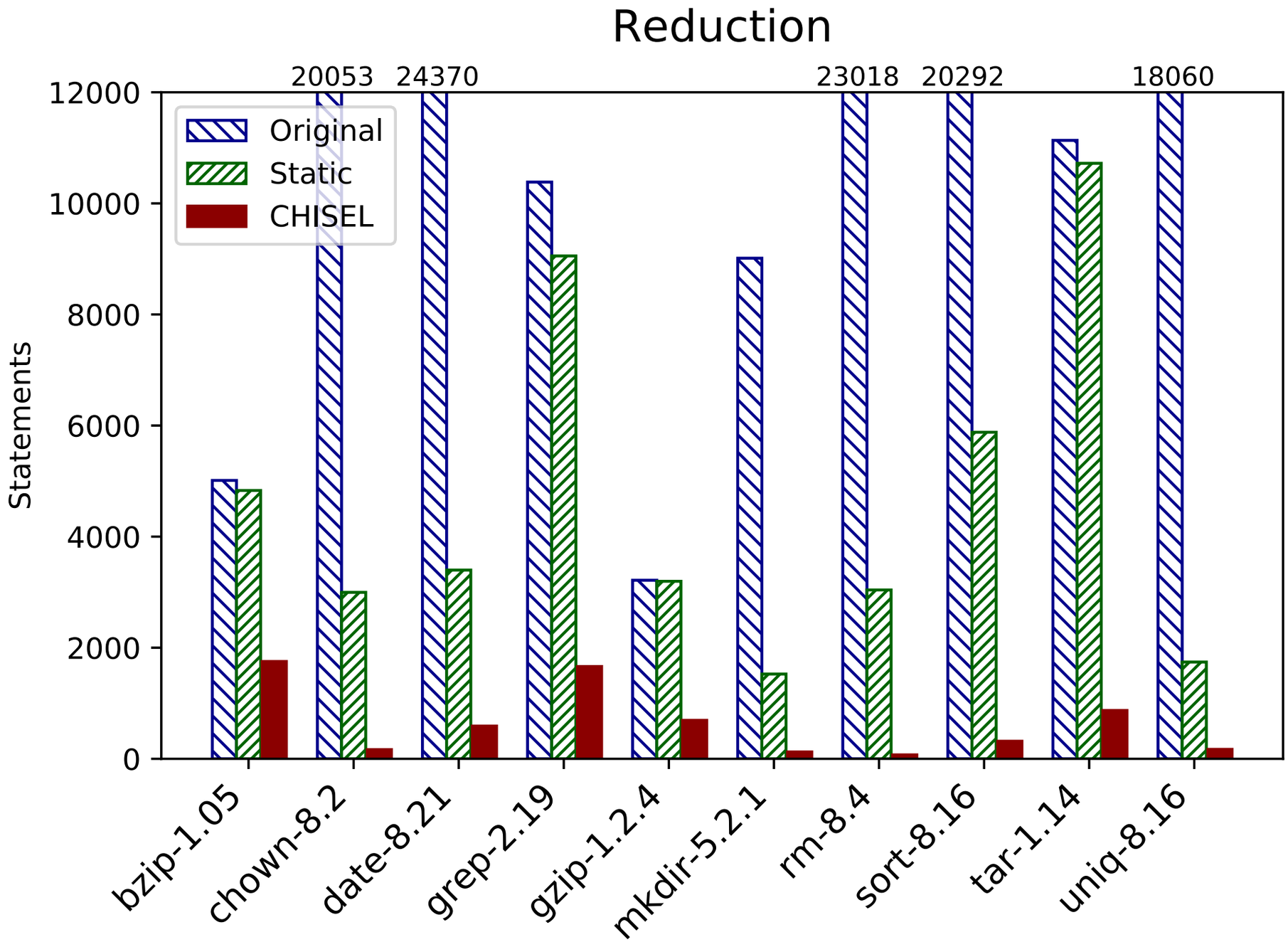
Guided Search
10 iters

Benchmarks

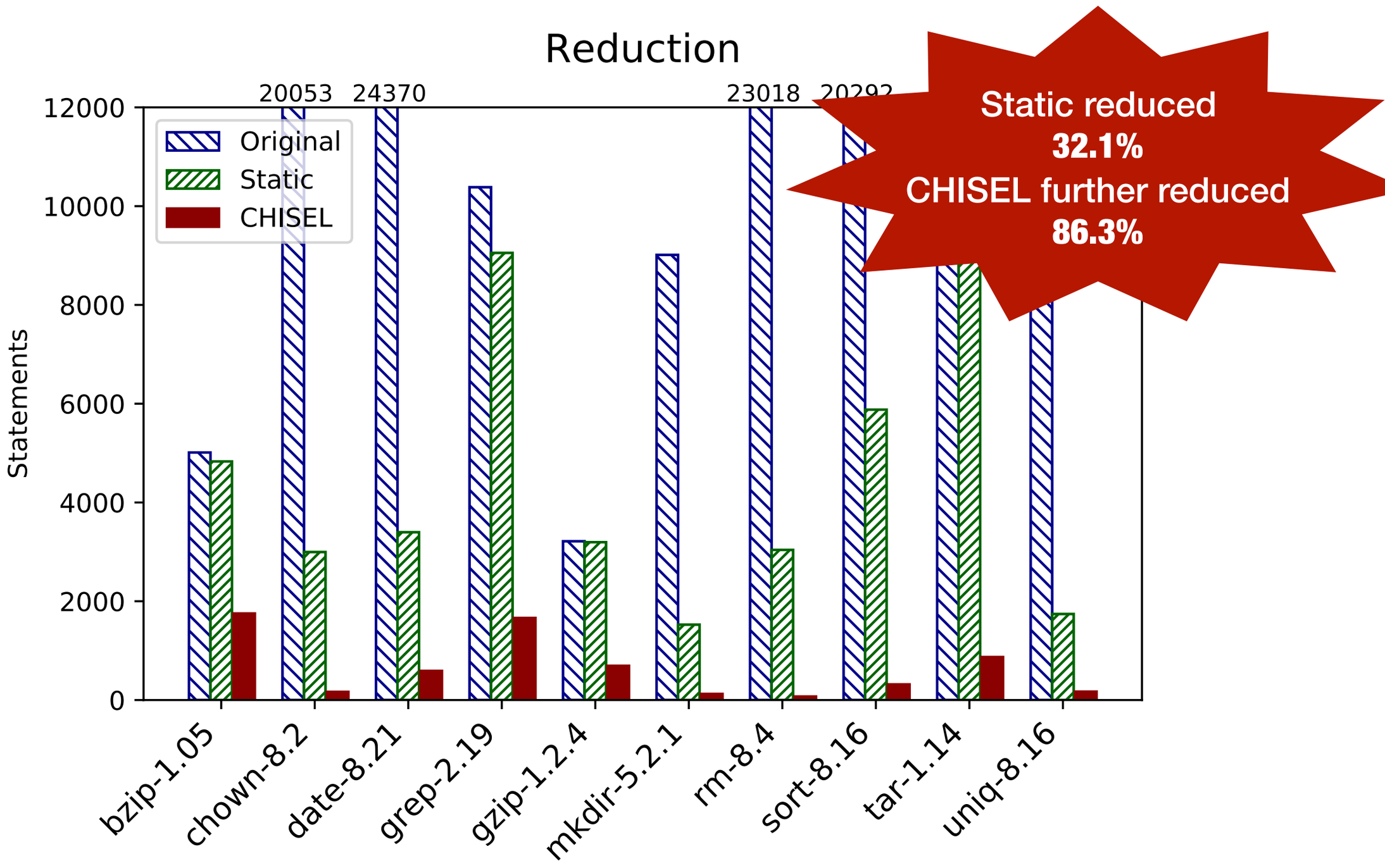
Program	LOC	#Func	#Stmt	CVE ID (CVSS Score)		Vulnerability
bzip-1.05	12,209	108	5,012	CVE-2011-4089	(4.6)	Executing arbitrary code by precreating a temporary directory.
chown-8.2	69,894	757	20,053	CVE-2017-18018	(1.9)	Modifying the ownership of arbitrary files with the "-R -L" option.
date-8.21	98,066	878	24,370	CVE-2014-9471	(7.5)	Executing arbitrary code with the "-d" option.
grep-2.19	49,011	432	10,383	CVE-2015-1345	(2.1)	Causing a crash with the "-F" option.
gzip-1.2.4	8,815	93	3,215	CVE-2005-1228	(5.0)	Writing to arbitrary directories with the "-N" option.
mkdir-5.2.1	28,202	263	9,013	CVE-2005-1039	(3.7)	Modifying the ownership of arbitrary files with the "-m" option.
rm-8.4	89,694	764	23,018	CVE-2015-1865	(3.3)	Modifying the ownership of arbitrary files with the "-rf" option.
sort-8.16	71,315	753	20,292	CVE-2013-0221	(4.3)	Causing a crash with the "-d" or "-M" option.
tar-1.14	31,605	502	11,134	CVE-2016-6321	(5.0)	Writing to arbitrary files.
uniq-8.16	64,915	665	18,060	CVE-2013-0222	(2.1)	Causing a crash with a long input string.
Total	577,350	5,790	157,940			

- Specification:
  - only supporting the same cmd options as BusyBox
  - with regression test cases in the repository

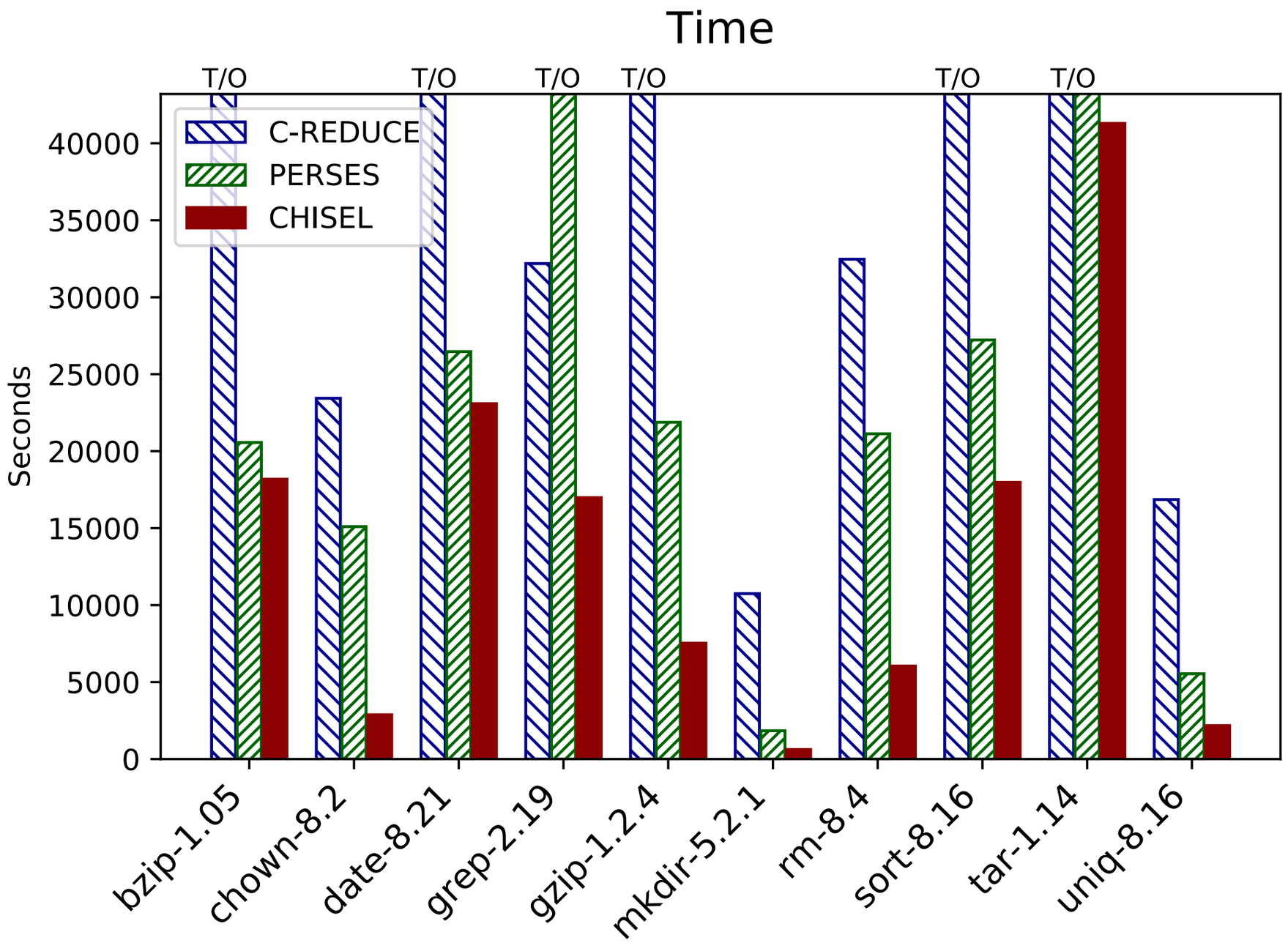
Result


Reduction
20053
24370
23018
20292
18060
12000
10000
8000
6000
4000
2000
0
Statements
Original
Static
CHISEL
bzip-1.05
chown-8.2
date-8.21
grep-2.19
gzip-1.2.4
mkdir-5.2.1
rm-8.4
sort-8.16
tar-1.14
uniq-8.16

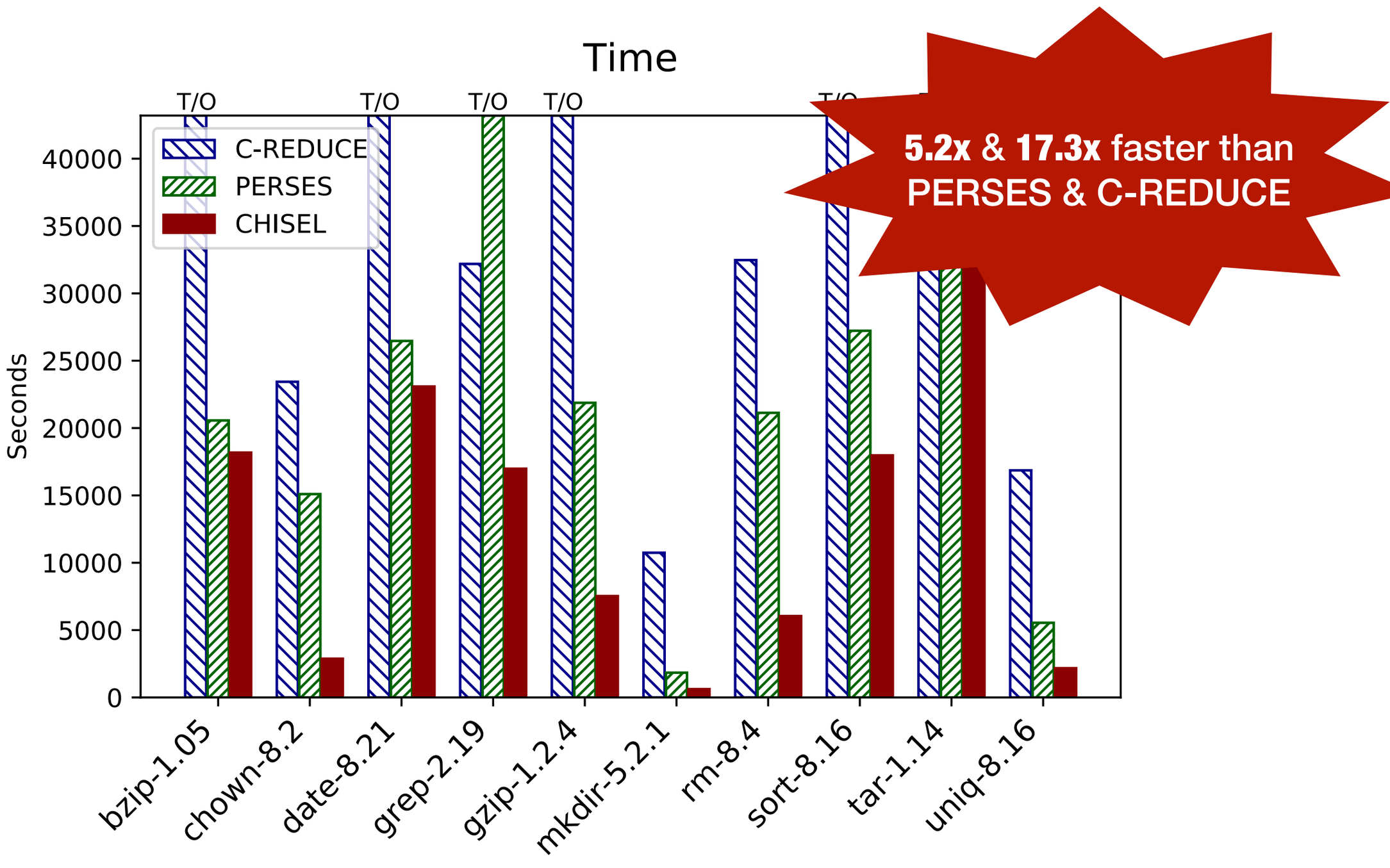
Result


Reduction
20053
24370
23018
20292
12000
10000
8000
6000
4000
2000
0
Statements
Original
Static
CHISEL
Static reduced
32.1%
CHISEL further reduced
86.3%
bzip-1.05
chown-8.2
date-8.21
grep-2.19
gzip-1.2.4
mkdir-5.2.1
rm-8.4
sort-8.16
tar-1.14
uniq-8.16

Result


Time
T/O
T/O
T/O
T/O
T/O
T/O
C-REDUCE
PERSES
CHISEL
Seconds
40000
35000
30000
25000
20000
15000
10000
5000
0
bzip-1.05
chown-8.2
date-8.21
grep-2.19
gzip-1.2.4
mkdir-5.2.1
rm-8.4
sort-8.16
tar-1.14
uniq-8.16

Result


Time
T/O
T/O
T/O
T/O
T/O
C-REDUCE
PERSES
CHISEL
Seconds
40000
35000
30000
25000
20000
15000
10000
5000
0
bzip-1.05
chown-8.2
date-8.21
grep-2.19
gzip-1.2.4
mkdir-5.2.1
rm-8.4
sort-8.16
tar-1.14
uniq-8.16
5.2x & 17.3x faster than
PERSES & C-REDUCE

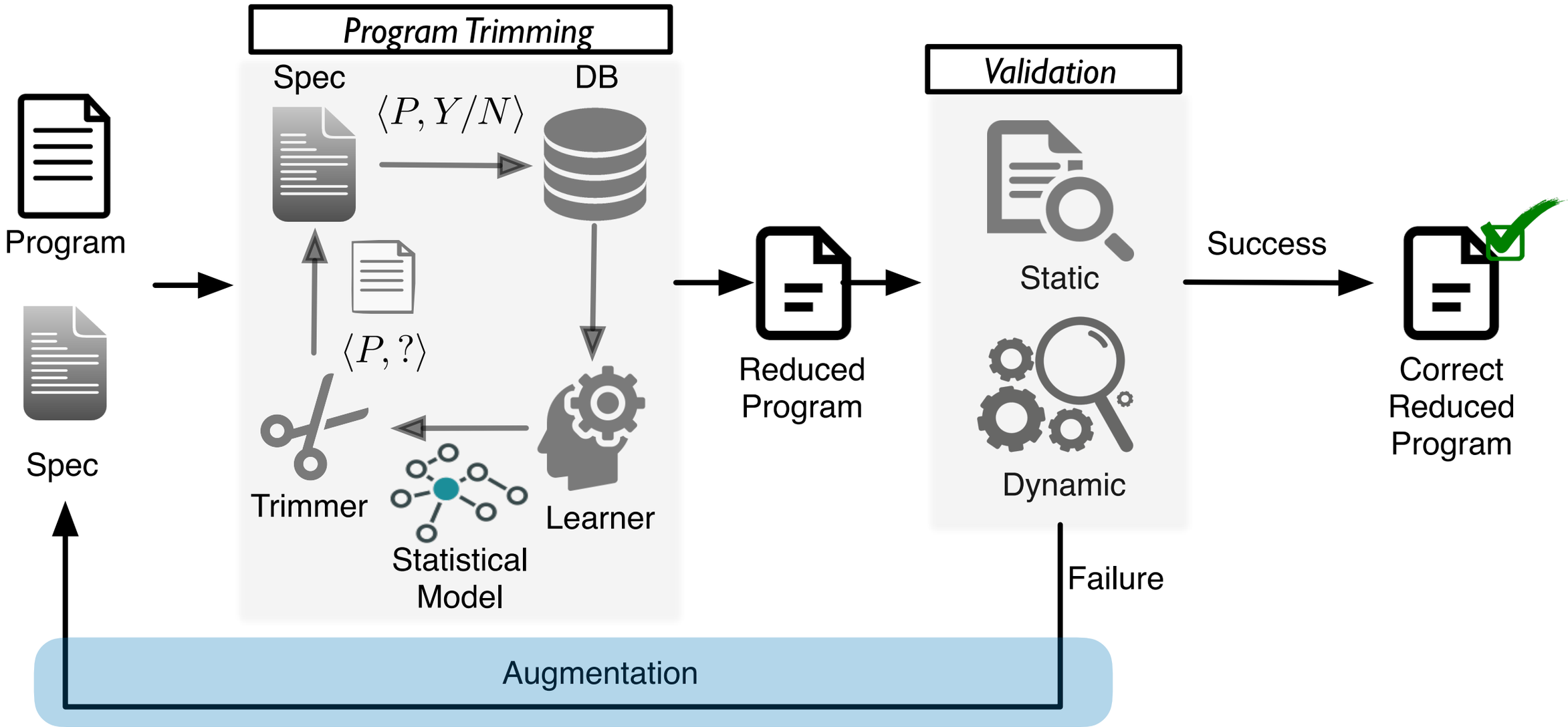
Result

Program	GNU CoreUtil		BusyBox		CHISEL
	#Opt	#Stmt	#Opt	#Stmt	#Stmt
bzip-1.05	15	4,831	5	2,031	1,752
chown-8.2	15	2,997	2	141	166
date-8.21	11	3,398	7	91	590
grep-2.19	45	9,053	16	312	1,662
gzip-1.2.4	18	3,196	3	842	692
mkdir-5.2.1	7	3,045	2	86	123
rm-8.4	12	3,040	3	88	71
sort-8.16	31	5,879	7	58	316
tar-1.14	97	10,721	8	349	870
uniq-8.16	12	1,742	7	39	170
Total	263	47,902	60	4,037	6,412

Result

Program	CVE	#Gadgets		#Alarms	
		Original	Reduced	Original	Reduced
bzip-1.05	✗	708	387 (45.3%)	1,993	320 (83.9%)
chown-8.2	✓	469	178 (62.0%)	47	1 (97.9%)
date-8.21	✓	470	237 (49.6%)	201	23 (88.6%)
grep-2.19	✓	1,169	848 (27.5%)	619	51 (91.8%)
gzip-1.2.4	✓	398	257 (35.4%)	326	128 (60.7%)
mkdir-5.2.1	✗	231	119 (48.5%)	43	2 (95.3%)
rm-8.4	✗	497	99 (80.1%)	48	0 (100.0%)
sort-8.16	✓	846	191 (77.4%)	673	5 (99.3%)
tar-1.14	✓	1,340	528 (60.6%)	1,295	23 (98.2%)
uniq-8.16	✗	313	115 (63.3%)	60	1 (98.3%)
Total		6,441	2,959 (54.1%)	5,305	554 (89.6%)

CHISEL: Program Reducer Framework



Augmentation

```
/* grep-2.19 */
struct dfa {
     token* tokens;
     int talloc;
     int tindex;
}

struct dfa *dfa;

void add_tok (token t) {
     /* removed in the first trial
        and restored after augmentation */
     if (dfa->talloc == dfa->tindex)
          dfa->tokens = (token *) realloc (/* large size */);
     *(dfa->tokens + (dfa->tindex++)) = t;
}
```

Conclusion

- CHISEL: Program debloating via reinforcement learning
- Evaluation
  - debloating Unix utilities and removing existing CVEs
  - without introducing new bugs
  - more efficient than existing state-of-the-art tools

Conclusion

- CHISEL: Program debloating via reinforcement learning
- Evaluation
  - debloating Unix utilities and removing existing CVEs
  - without introducing new bugs
  - more efficient than existing state-of-the-art tools

Thank you