Carl von Ossietzky
Universität
Oldenburg

Benchmark generation for software verifiers with semantics preserving transformations

Florian Dyck, 05.10.2022

Problem

```
 1 extern void reach_error();
 2
 3
 4
 5
 6 int main(){
 7     int i = 0;
 8     while(i < 5) i++;
 9     if(i < 5) reach_error();
10     return 0;
11 }
```

Carl von Ossietzky
Universität
Oldenburg

Problem

Programm	P'
Ground Truth	?

Problem

Programm	P'
Ground Truth	?

CPAchecker

Problem

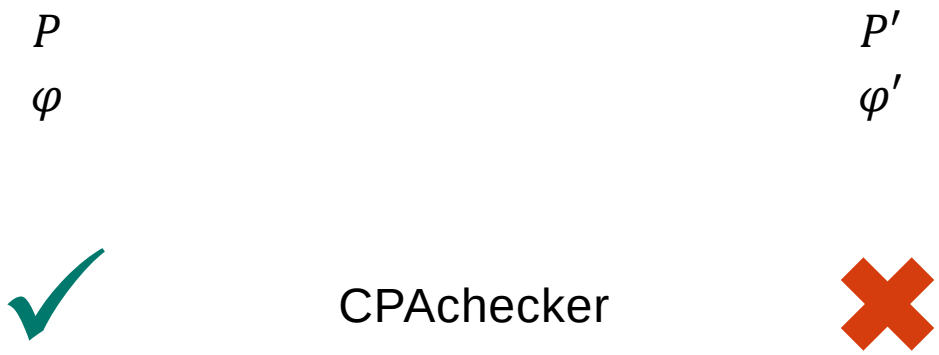
Programm	P'
Ground Truth	?
CPAchecker	

Approach

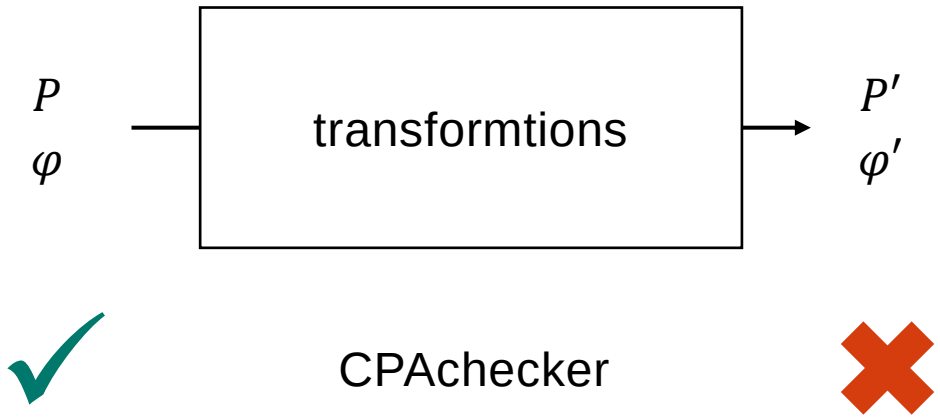
Programm P'
Ground Truth φ'

CPAchecker 

Approach

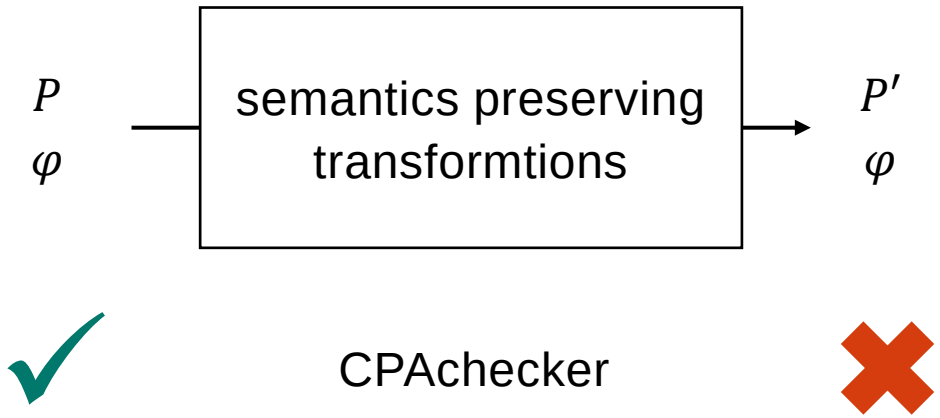


Approach



Carl von Ossietzky
Universität
Oldenburg

Approach



Carl von Ossietzky Universität Oldenburg

Example transformations – deepen_while

```
 1 extern void reach_error();
 2
 3
 4
 5
 6 int main(){
 7     int i = 0;
 8     while(i < 5) i++;
 9     if(i < 5) reach_error();
10     return 0;
11 }
```

Example transformations – deepen_while

```
 1 extern void reach_error();
 2
 3
 4
 5
 6 int main(){
 7     int i = 0;
 8     while(i < 5) while(i < 5) i++;
 9     if(i < 5) reach_error();
10     return 0;
11 }
```

Example transformations – deepen_while

```
 1 extern void reach_error();
 2 extern int VERIFIER_nondet_int();
 3
 4
 5
 6 int main(){
 7     int i = 0;
 8     while(i < 5) while(i < 5 & VERIFIER_nondet_int()) i++;
 9     if(i < 5) reach_error();
10     return 0;
11 }
```

Example transformations – to_function

```
 1 extern void reach_error();
 2 extern int VERIFIER_nondet_int();
 3
 4
 5
 6 int main(){
 7     int i = 0;
 8     while(i < 5) while(i < 5 & VERIFIER_nondet_int()) i++;
 9     if(i < 5) reach_error();
10     return 0;
11 }
```

Carl von Ossietzky Universität Oldenburg

Example transformations – to_function

```
 1 extern void reach_error();
 2 extern int VERIFIER_nondet_int();
 3 void function(int * i){
 4
 5 }
 6 int main(){
 7     int i = 0;
 8     while(i < 5) function(&i);& VERIFIER_nondet_int()) i++;
 9     if(i < 5) reach_error();
10     return 0;
11 }
```

Carl von Ossietzky Universität Oldenburg

Example transformations – to_function

```
 1 extern void reach_error();
 2 extern int VERIFIER_nondet_int();
 3 void function(int * i){
 4     while(*i < 5 & VERIFIER_nondet_int()) (*i)++;
 5 }
 6 int main(){
 7     int i = 0;
 8     while(i < 5) function(&i);
 9     if(i < 5) reach_error();
10     return 0;
11 }
```

Example transformations – to_array

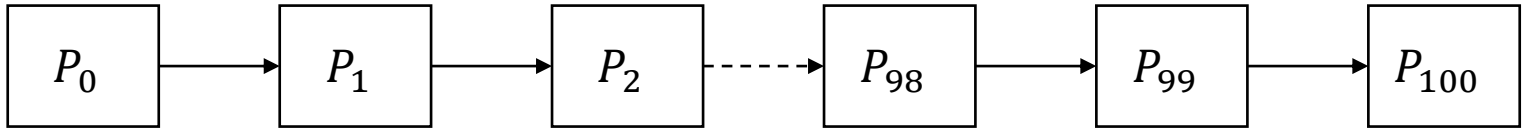
```
 1 extern void reach_error();
 2 extern int VERIFIER_nondet_int();
 3 void function(int * i){
 4     while(*i < 5 & VERIFIER_nondet_int()) (*i)++;
 5 }
 6 int main(){
 7     int i = 0;
 8     while(i < 5) function(&i);
 9     if(i < 5) reach_error();
10     return 0;
11 }
```

Example transformations – to_array

```
 1 extern void reach_error();
 2 extern int VERIFIER_nondet_int();
 3 void function(int * i){
 4     while(*i < 5 & VERIFIER_nondet_int()) (*i)++;
 5 }
 6 int main(){
 7     int i[1] = {0};
 8     while(i[0] < 5) function(&(i[0]));
 9     if(i[0] < 5) reach_error();
10     return 0;
11 }
```

Benchmark

- Generated from 2129 programs of the svcomp22 with reachability properties
- 100 transformations applied iteratively to each



- Verified with CPAchecker (predicateAnalysis and SVComp22)
  - 900 s Timeout
  - 8 GB Memory

Carl von Ossietzky Universität Oldenburg

Verification results

Analysis	predicateAnalysis			SVComp22	
Benchmark	original	transformed	no_pointers	original	no_pointers
Correct	635	536	**445**	1346	**724**
proofs	324	**533**	229	863	465
alarms	311	3	216	483	259
Incorrect	17	193	**105**	4	**134**
proofs	0	**186**	19	0	37
alarms	17	**7**	86	4	97
Timeout	651	**0**	601	740	1133
OOM	6	3	13	23	106
Error	638	**1325**	925	5	7

Example Bug

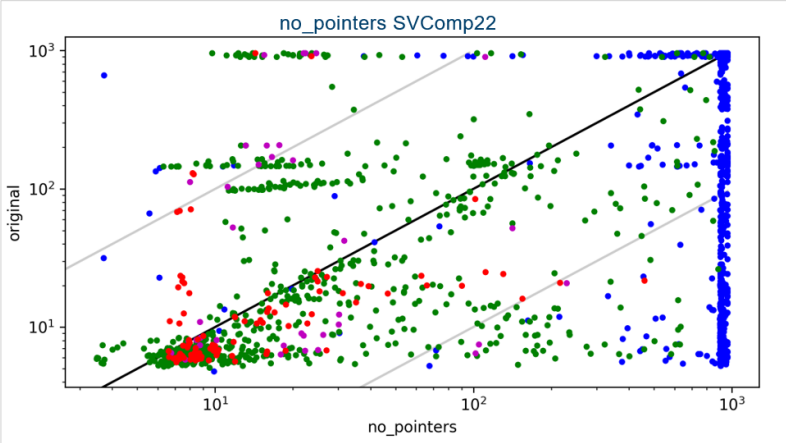
```
 1 void reach_error() {}
 2
 3 void mutex_lock(int *a) {
 4     int cond = *a;
 5     if (cond)
 6         reach_error();
 7     *a = 1;
 8 }
 9
10 int main() {
11     int m = 0;
12     mutex_lock(&m);
13     mutex_lock(&m);
14 }
```

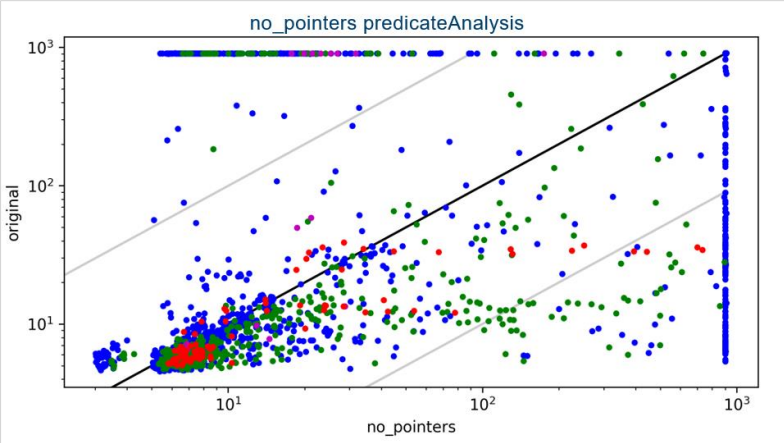
Example Bug

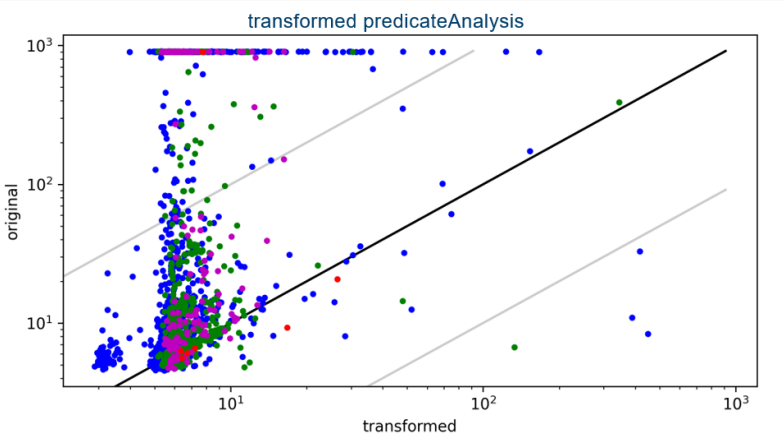
```
 1 void reach_error() {}
 2
 3 void mutex_lock(int *a) {
 4     int cond[1] = {*a};
 5     if (cond[0])
 6         reach_error();
 7     *a = 1;
 8 }
 9
10 int main() {
11     int m = 0;
12     mutex_lock(&m);
13     mutex_lock(&m);
14 }
```

Carl von Ossietzky
Universität
Oldenburg

Ressource usage cpu-time (s)

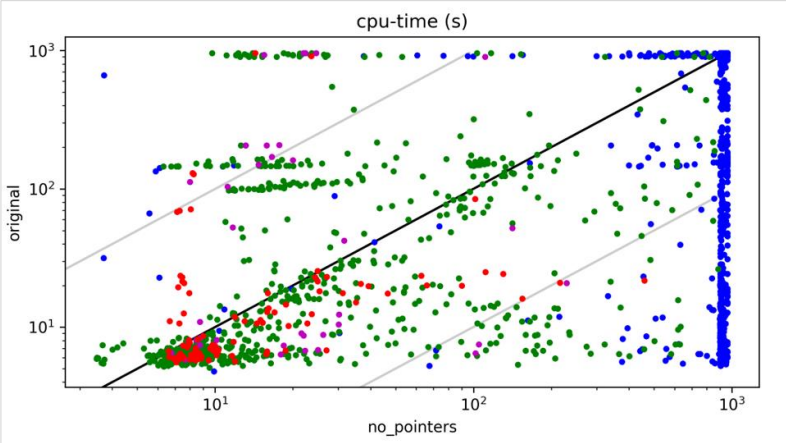


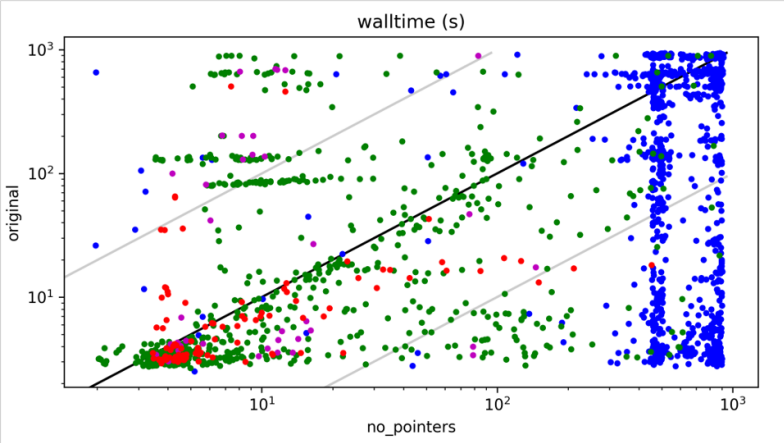


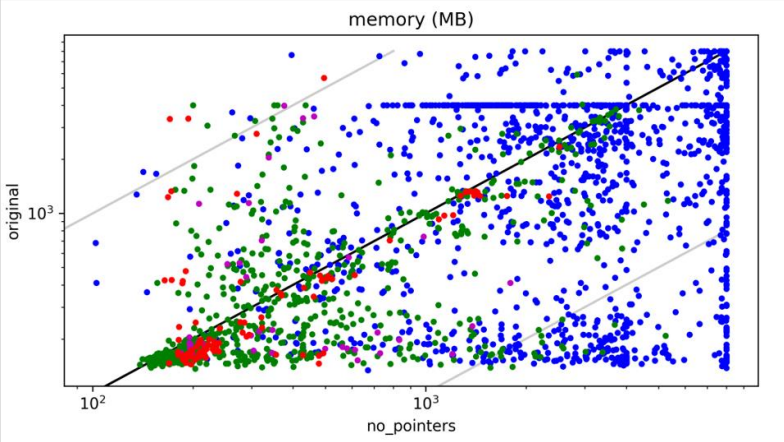


Carl von Ossietzky
Universität
Oldenburg

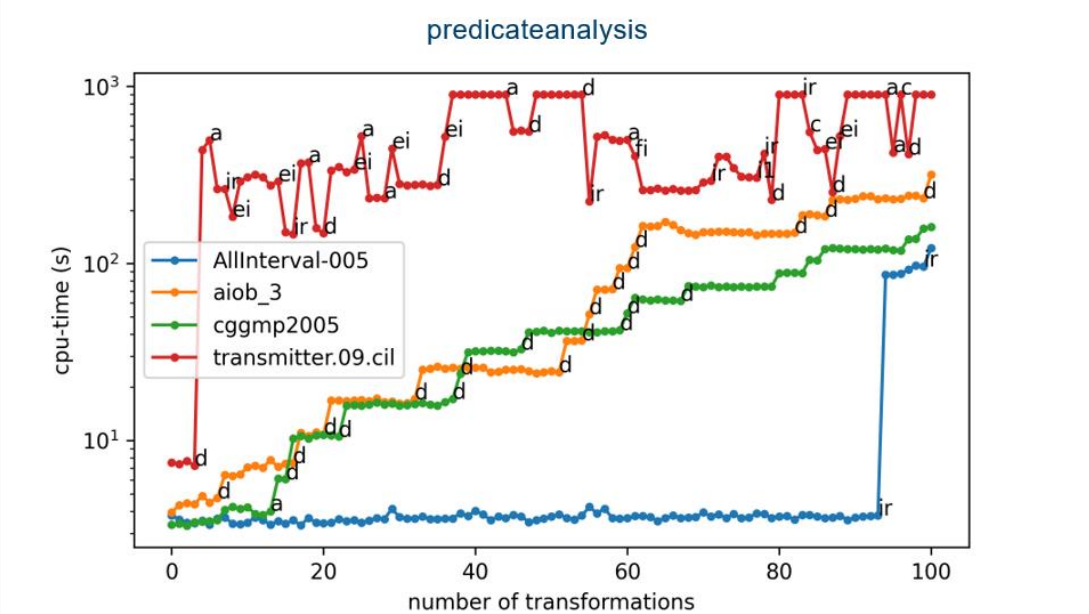
Resource usage no_pointers SVComp22

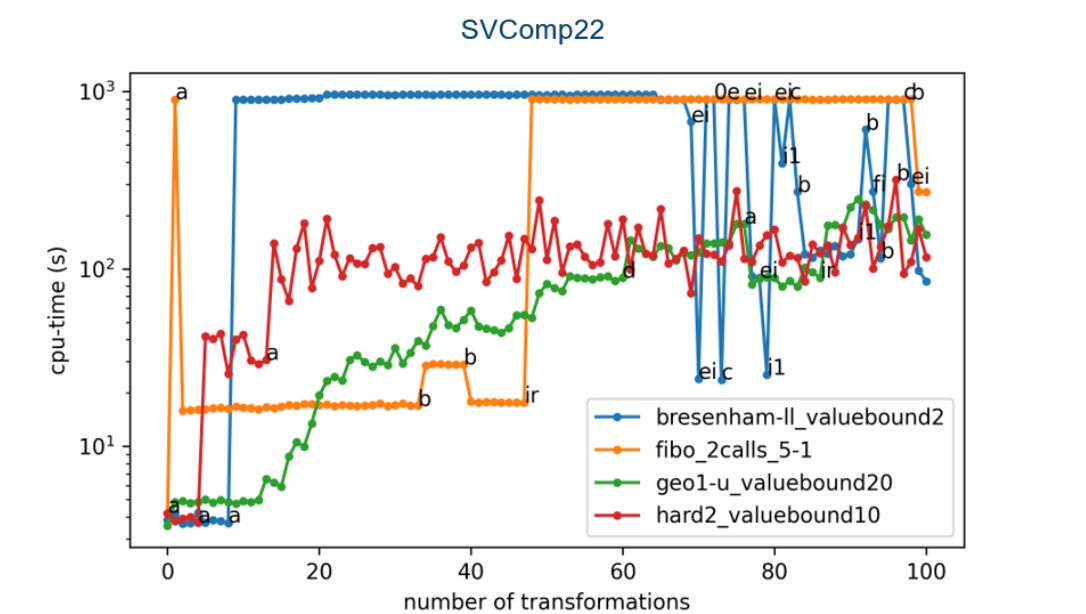






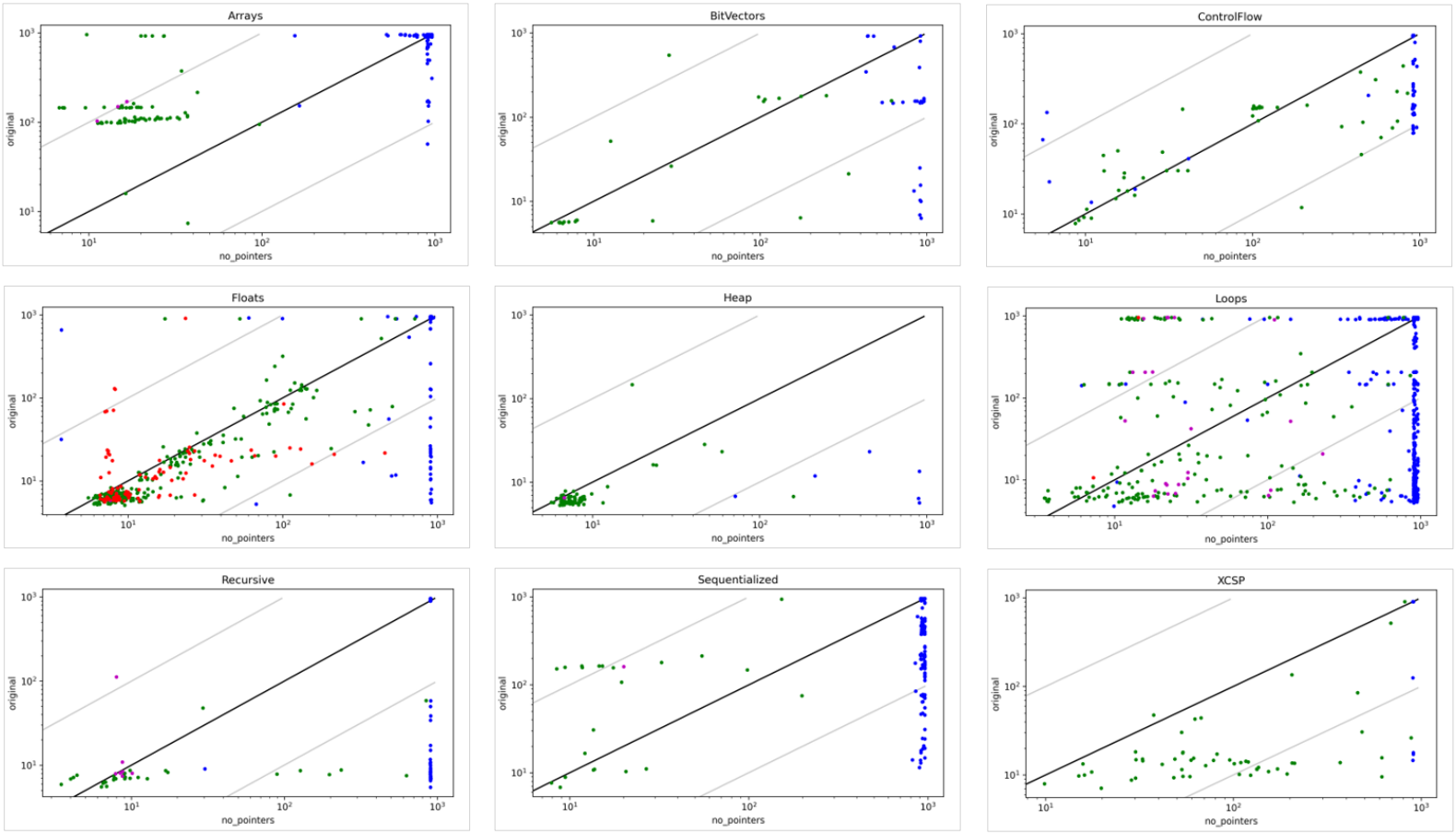
Strongest slowdowns

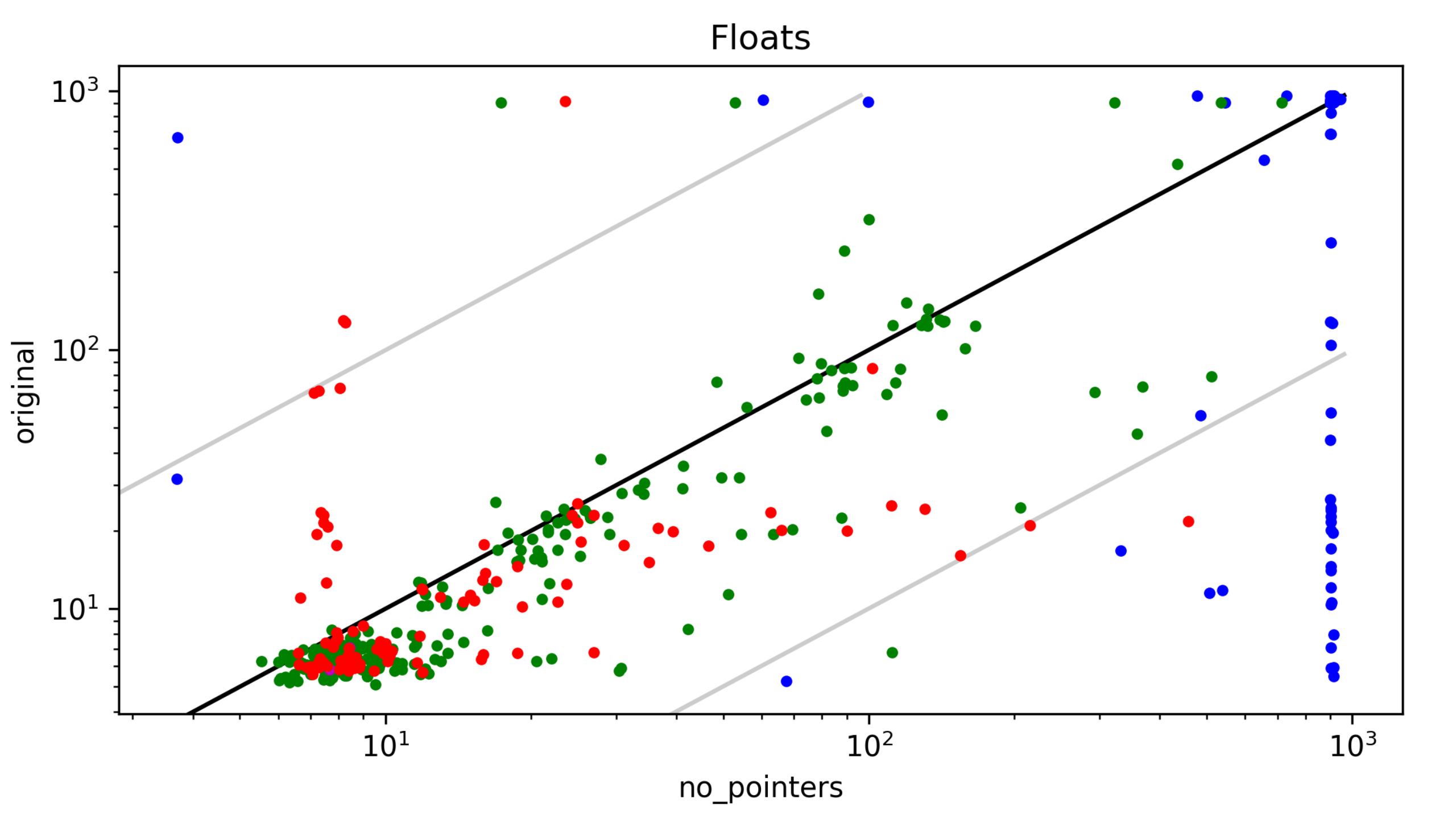


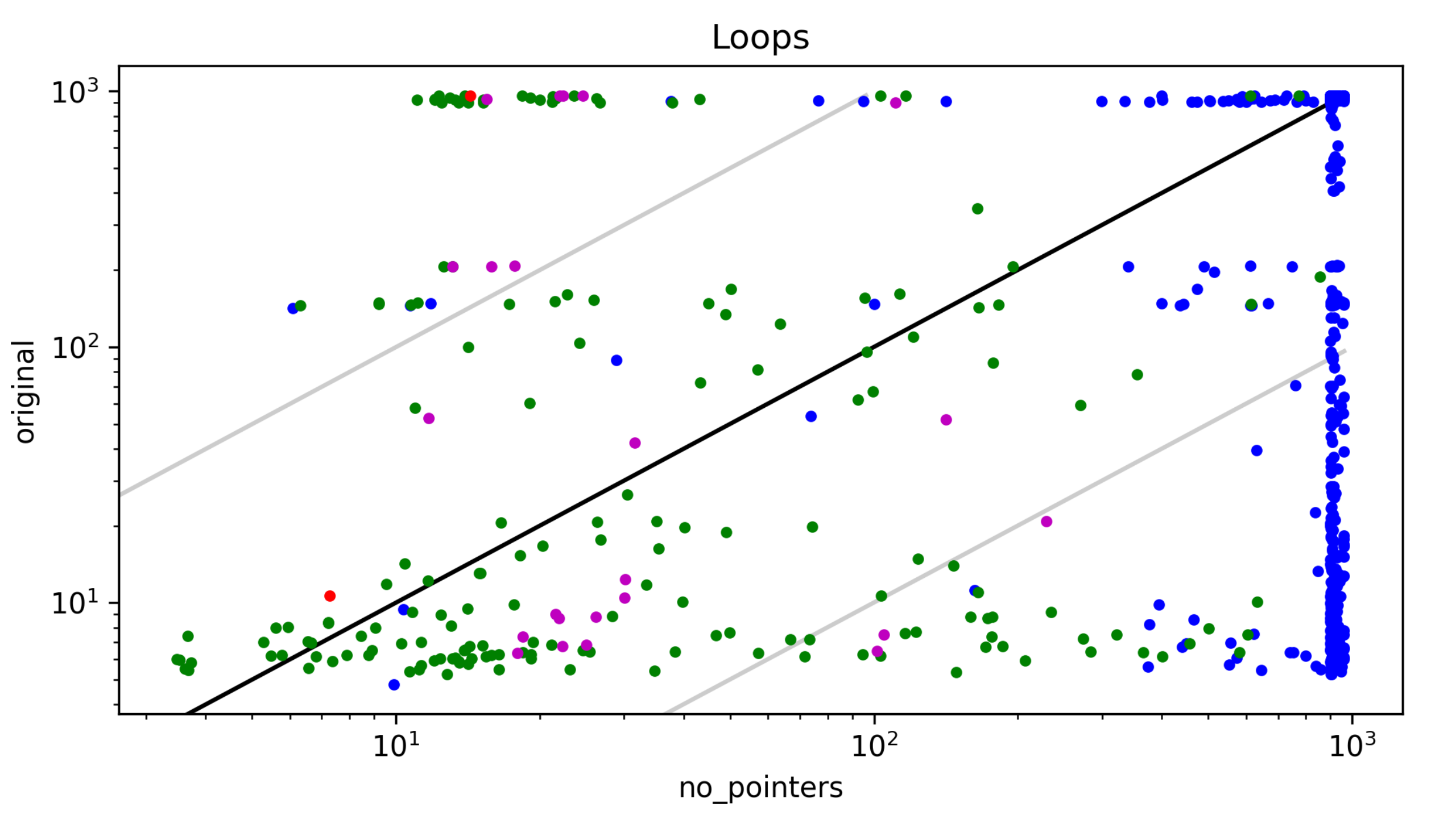


Carl von Ossietzky
Universität
Oldenburg

categories – cpu-time no_pointers SVComp22




Floats
original
no_pointers
10^1
10^2
10^3

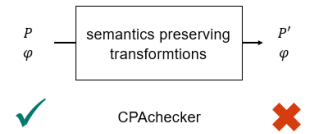

Loops
10^3
10^2
10^1
original
10^1
10^2
10^3
no_pointers

Results

- More incorrect and less correct verifications, caused by:
  - Pointers on functions
  - Arrays in combination with pointers and unions
- More resource usage during verifications, caused by:
  - Deeper loops
  - Rare cases of other transformations

Overview

Approach



Benchmark generation for software verifiers with semantics preserving transformations
Florian Dyck, Carl von Ossietzky Universität Oldenburg

Seite 12
06.10.2022

Example transformations – to_array

```
1 extern void reach_error();
2 extern int VERIFIER_nondet_int();
3 void function(int * i){
4     while(*i < 5 & VERIFIER_nondet_int()) (*i)++;
5 }
6 int main(){
7     int i[1] = {0};
8     while(i[0] < 5) function(&(i[0]));
9     if(i[0] < 5) reach_error();
10    return 0;
11 }
```

Benchmark generation for software verifiers with semantics preserving transformations
Florian Dyck, Carl von Ossietzky Universität Oldenburg

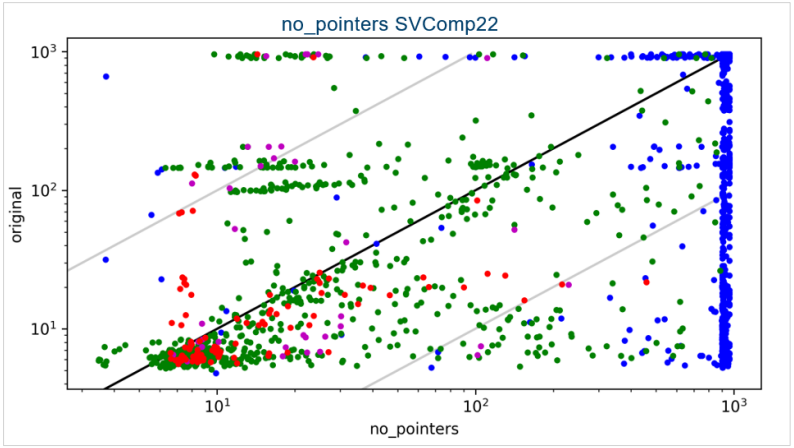
Seite 22
06.10.2022

Verification results

Analysis	predicateAnalysis			SVComp22	
Benchmark	original	transformed	no_pointers	original	no_pointers
Correct	635	536	**445**	1346	**724**
proofs	324	**533**	229	863	465
alarms	311	3	216	483	259
Incorrect	17	193	**105**	4	**134**
proofs	0	**186**	19	0	37
alarms	17	**7**	86	4	97
Timeout	651	**0**	601	740	1133
OOM	6	3	13	23	106
Error	638	**1325**	925	5	7

Benchmark generation for software verifiers with semantics preserving transformations
Florian Dyck, Carl von Ossietzky Universität Oldenburg

Seite 24
06.10.2022



Sources

– **SVComp22:** Dirk Beyer. 11th Competition on Software Verification (SV-COMP 2022). https://sv-comp.sosy-lab.org/2022/benchmarks.php.
[Online; last accessed 09.09.2022]. 2022.

– **CPAchecker:** Dirk Beyer. CPAchecker The Configurable Software-Verification Platform. https://cpachecker.sosy-lab.org/achieve.php.
[Online; last accessed 09.09.2022]. 2022.

– **Presented tool:** https://github.com/Flo0112358/semtransforms