Racing on the Negative Force:

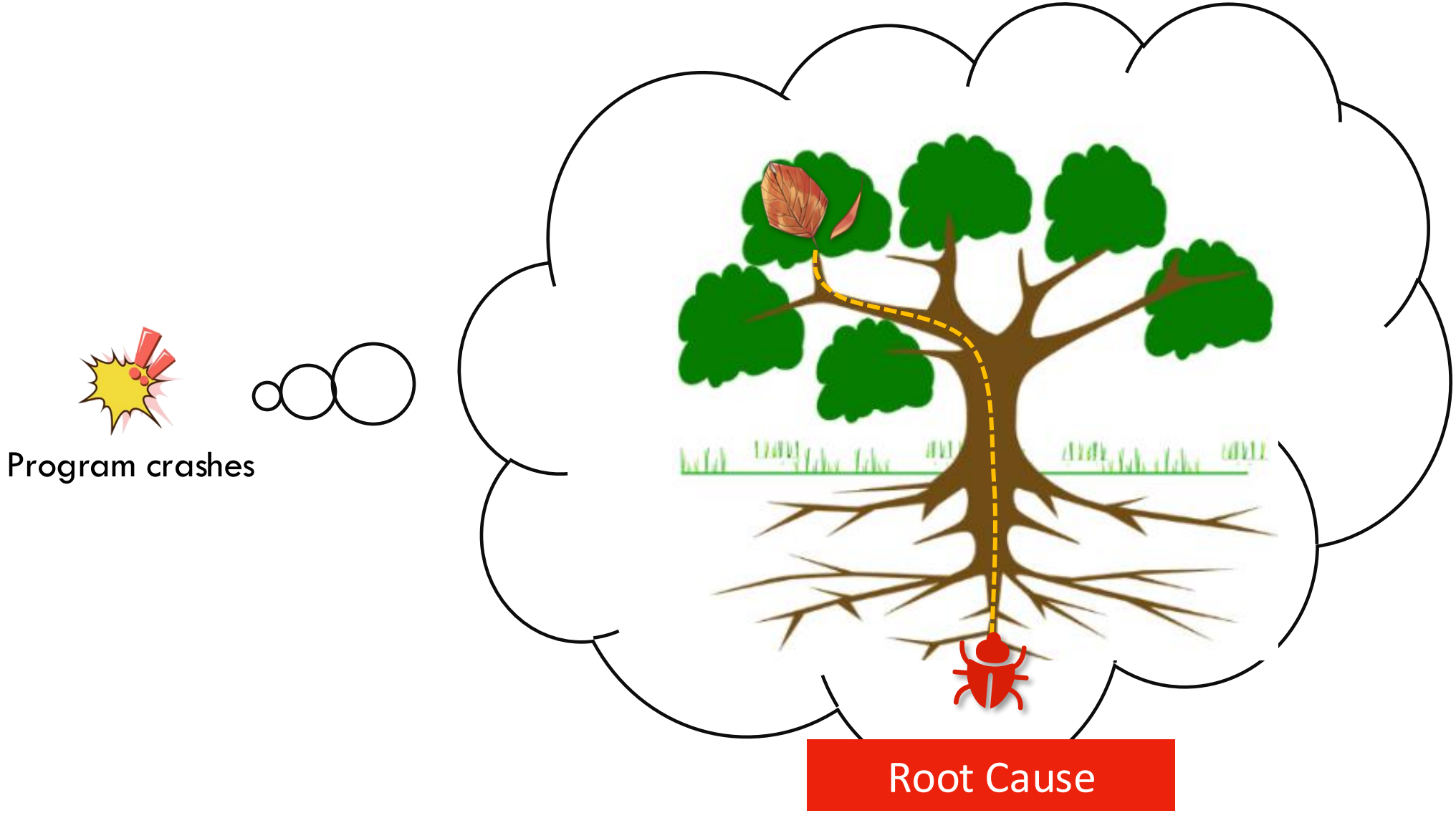
Efficient Vulnerability Root-Cause Analysis through Reinforcement Learning on Counterexamples

Dandan Xu, Di Tang, Yi Chen, XiaoFeng Wang, Kai Chen, Haixu Tang, Longxing Li
{xudandan, chenkai, lilongxing}@iie.ac.cn
{tangd, chen481, xw7, hatang}@iu.edu

中国科学院信息工程研究所
INSTITUTE OF INFORMATION ENGINEERING,CAS

INDIANA UNIVERSITY
BLOOMINGTON

What is the Root Cause?



What is the Root Cause?

```
16182  eopt = get_data (NULL, filedata, options_offset, 1,
16183                   sect->sh_size, _("options"));
16184  if (eopt)
16185  {
16186    iopt = (Elf_Internal_Options *)
16187          cmalloc ((sect->sh_size / sizeof (eopt)), sizeof (* iopt));
       ...

16194    offset = 0;
16195    option = iopt;
16196
16197    while (offset <= sect->sh_size - sizeof (* eopt))
16198    {
16199      Elf_External_Options * option;
16200
16201      eoption = (Elf_External_Options *)((char *) eopt + offset);
16202
16203      option->kind = BYTE_GET (eoption->kind);
16204      option->size = BYTE_GET (eoption->size);
16205
16216      offset += option->size;
16217      ++option;
16218    }
       ...
   }
```

Program crashes

An example: CVE-2019-9077

What is the Root Cause?

An example: CVE-2019-9077

```
16182  eopt = get_data (NULL, filedata, options_offset, 1,
16183                   sect->sh_size, _("options"));
16184  if (eopt)
16185  {
16186    iopt = (Elf_Internal_Options *)
16187          cmalloc ((sect->sh_size / sizeof (eopt)), sizeof (* iopt));
       ...
16194    offset = 0;
16195    option = iopt;
16196
16197    while (offset <= sect->sh_size - sizeof (* eopt))
16198    {
16199      Elf_External_Options * option;
16200
16201      eoption = (Elf_External_Options *)((char *) eopt + offset);
16202
16203      option->kind = BYTE_GET (eoption->kind);
16204      option->size = BYTE_GET (eoption->size);
16205
16216      offset += option->size;
16217      ++option;
16218    }
       ...
     }
```

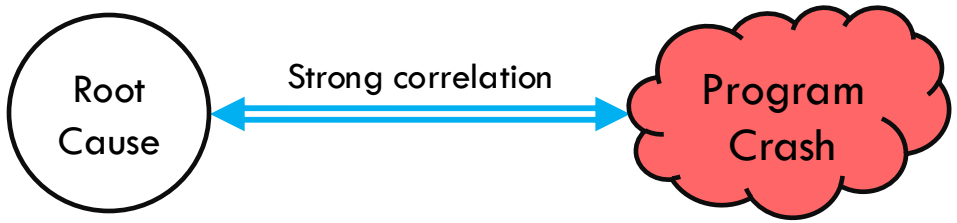
sect->sh_size=1 < 8

sect->sh_size=1 < 8

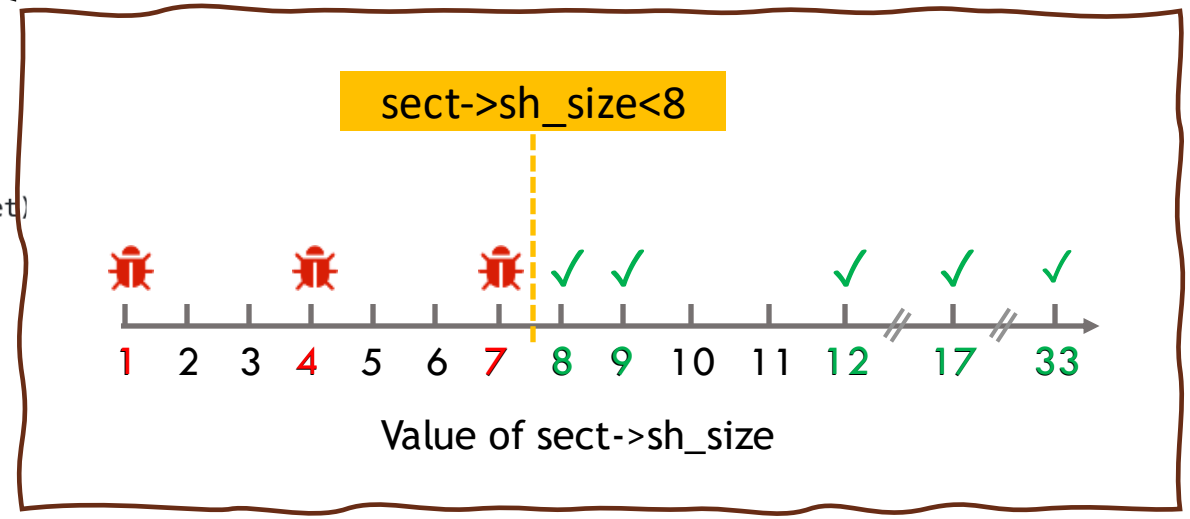
Root Cause

Program crashes

Statistical RCA



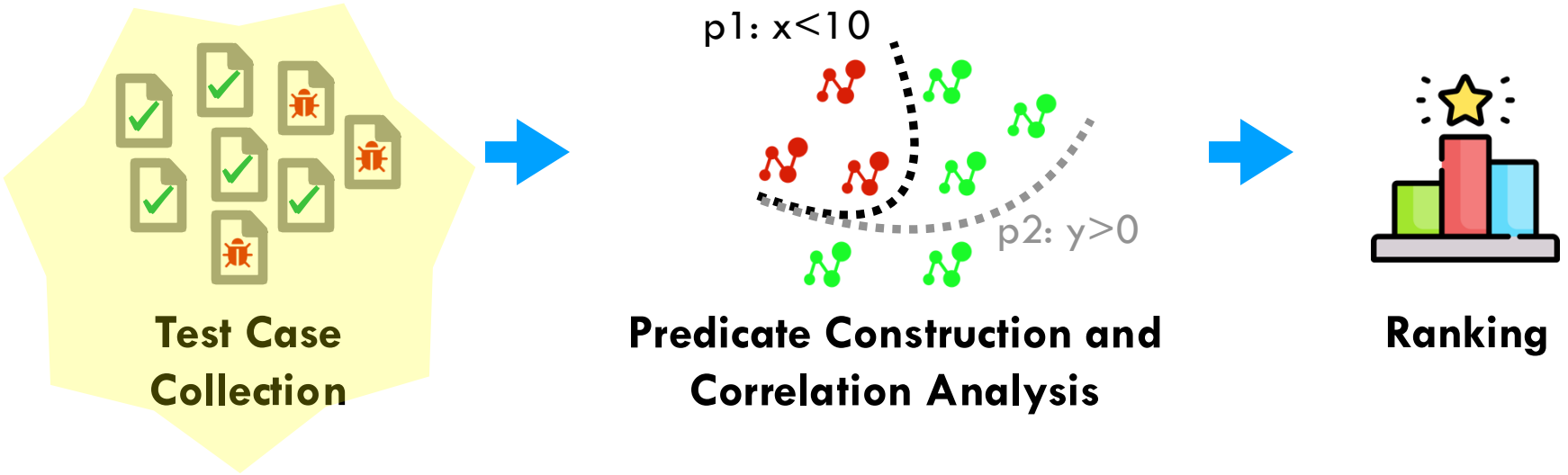

```
16197  while (offset <= sect->sh_size - sizeof (* eopt))
16198  {
16199    Elf_External_Options * option;
16200
16201    eoption = (Elf_External_Options *)((char *) eopt + offset);
16202
16203    option->kind = BYTE_GET (eoption->kind);
16204    option->size = BYTE_GET (eoption->size);
16205
16216    offset += option->size;
16217    ++option;
16218  }
```



A potential root cause!

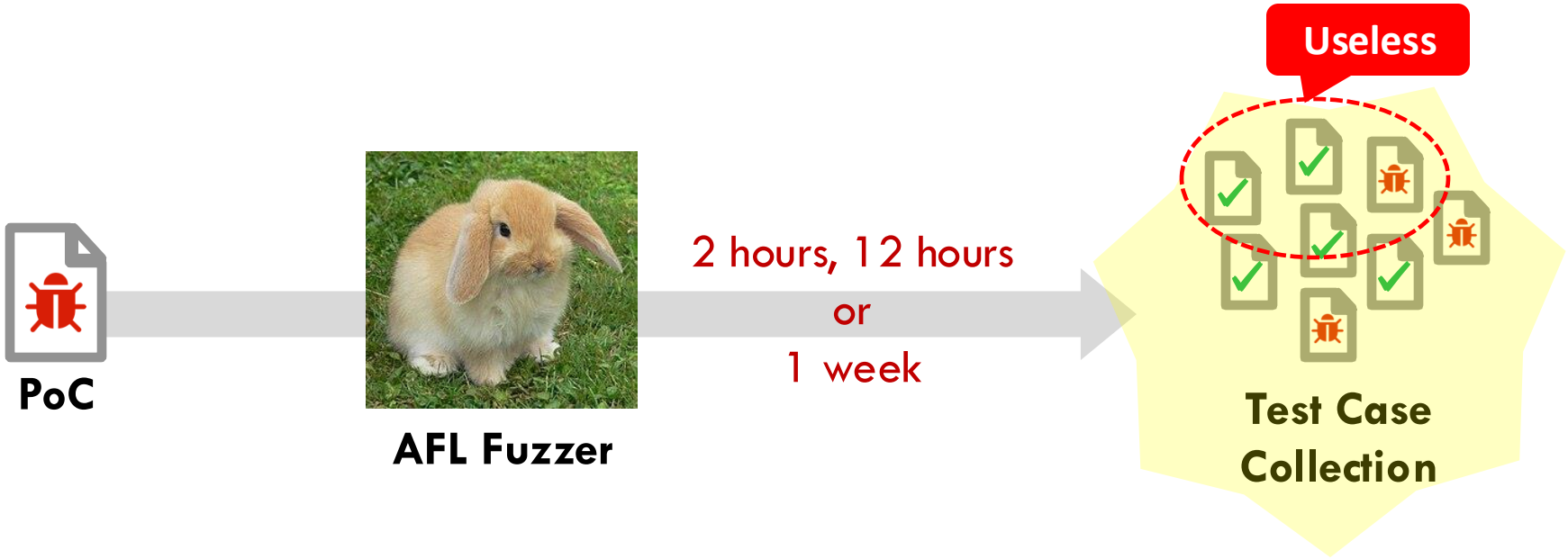
Statistical RCA

- General workflow



Statistical RCA

- Previous work

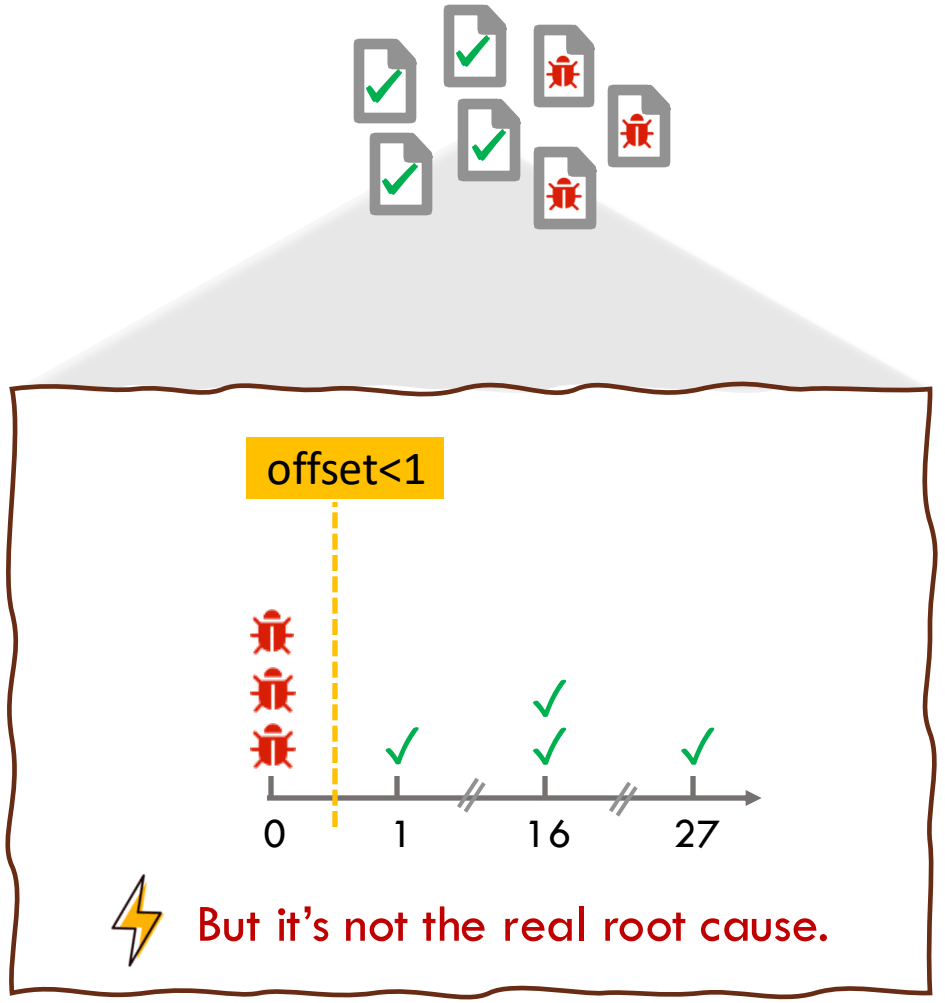


Which Test Cases are Most Useful?

- Example

```
16182  eopt = get_data (NULL, filedata, options_offset, 1,
16183                   sect->sh_size, _("options"));
16184  if (eopt)
16185  {
16186    iopt = (Elf_Internal_Options *)
16187          cmalloc ((sect->sh_size / sizeof (eopt)), sizeof (* iopt));
       ...

16194    offset = 0;
16195    option = iopt;
16196
16197    while (offset <= sect->sh_size - sizeof (* eopt))
16198    {
16199      Elf_External_Options * option;
16200
16201      eoption = (Elf_External_Options *)((char *) eopt + offset);
16202
16203      option->kind = BYTE_GET (eoption->kind);
16204      option->size = BYTE_GET (eoption->size);
16205
16216      offset += option->size;
16217      ++option;
16218    }
       ...
```

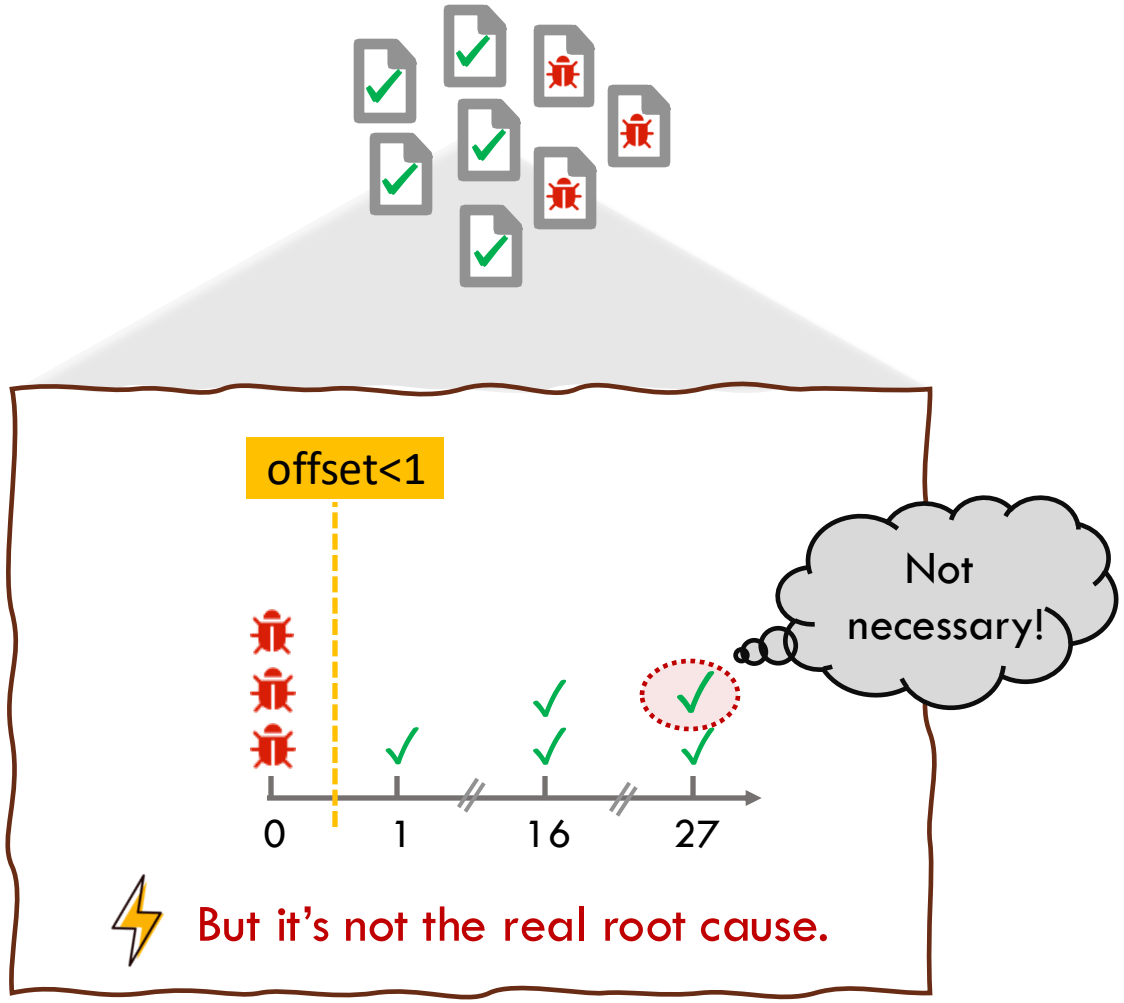


Which Test Cases are Most Useful?

- Example

```
16182  eopt = get_data (NULL, filedata, options_offset, 1,
16183                   sect->sh_size, _("options"));
16184  if (eopt)
16185  {
16186    iopt = (Elf_Internal_Options *)
16187          cmalloc ((sect->sh_size / sizeof (eopt)), sizeof (* iopt));
       ...

16194    offset = 0;
16195    option = iopt;
16196
16197    while (offset <= sect->sh_size - sizeof (* eopt))
16198    {
16199      Elf_External_Options * option;
16200
16201      eoption = (Elf_External_Options *)((char *) eopt + offset);
16202
16203      option->kind = BYTE_GET (eoption->kind);
16204      option->size = BYTE_GET (eoption->size);
16205
16216      offset += option->size;
16217      ++option;
16218    }
       ...
```

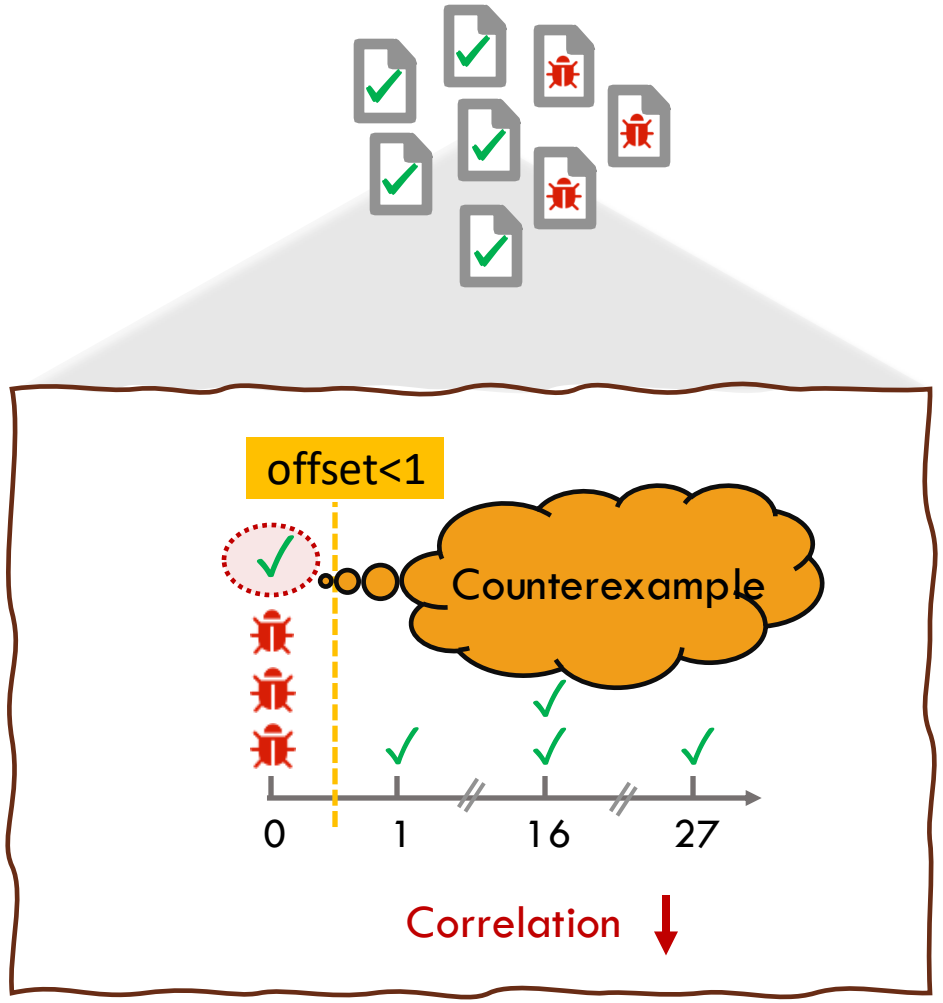


Which Test Cases are Most Useful?

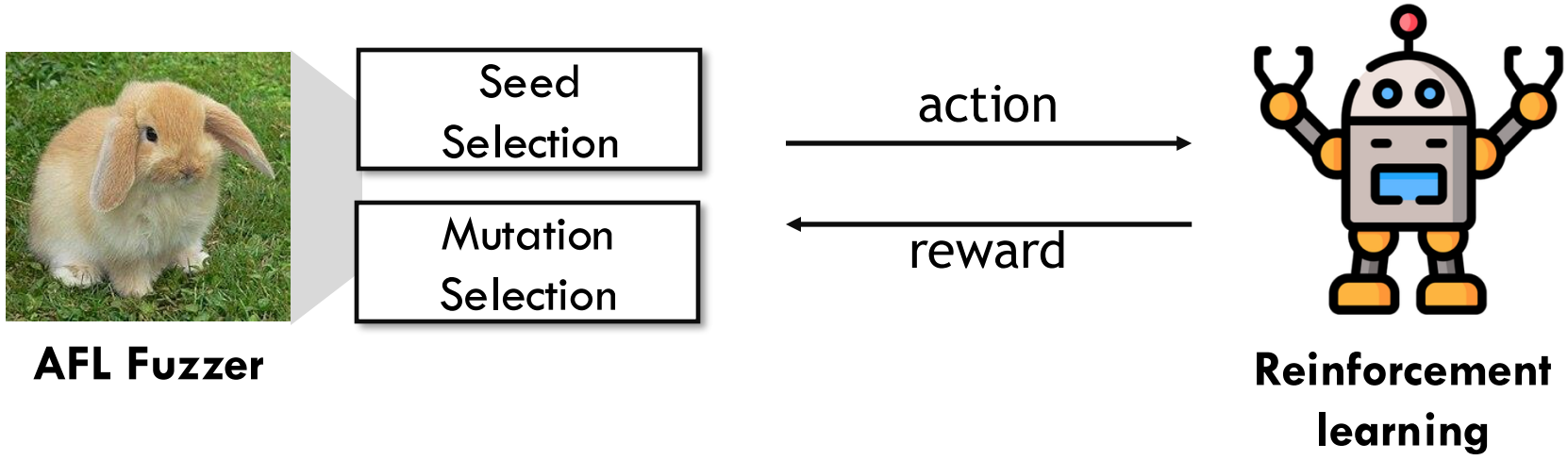
- Example

```
16182  eopt = get_data (NULL, filedata, options_offset, 1,
16183                   sect->sh_size, _("options"));
16184  if (eopt)
16185  {
16186    iopt = (Elf_Internal_Options *)
16187          cmalloc ((sect->sh_size / sizeof (eopt)), sizeof (* iopt));
       ...

16194    offset = 0;
16195    option = iopt;
16196
16197    while (offset <= sect->sh_size - sizeof (* eopt))
16198    {
16199      Elf_External_Options * option;
16200
16201      eoption = (Elf_External_Options *)((char *) eopt + offset);
16202
16203      option->kind = BYTE_GET (eoption->kind);
16204      option->size = BYTE_GET (eoption->size);
16205
16216      offset += option->size;
16217      ++option;
16218    }
       ...
```



How to Generate Counterexamples?

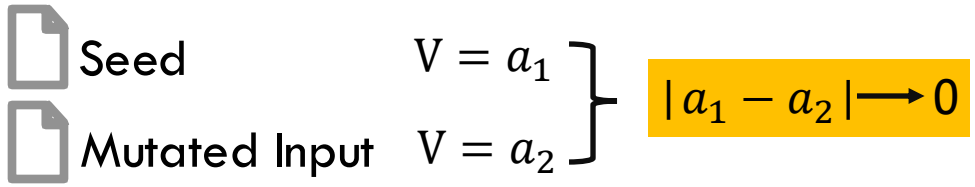


Counterexample for Predicate (CoP)

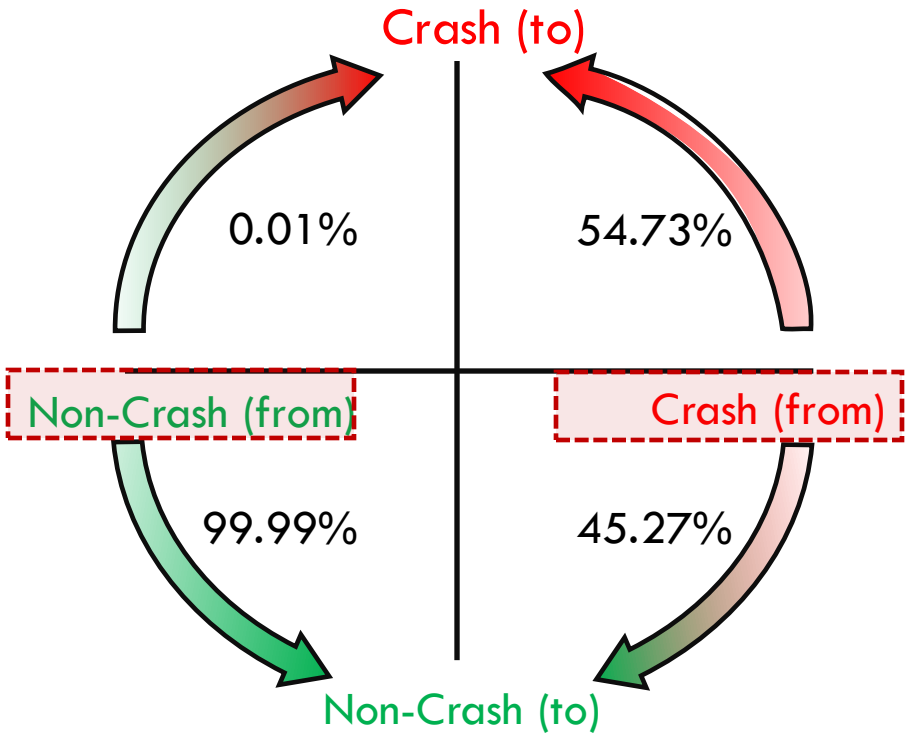
Counterexample for Ranking (CoR)

How to Generate Counterexamples?

- Observation1



- Observation2

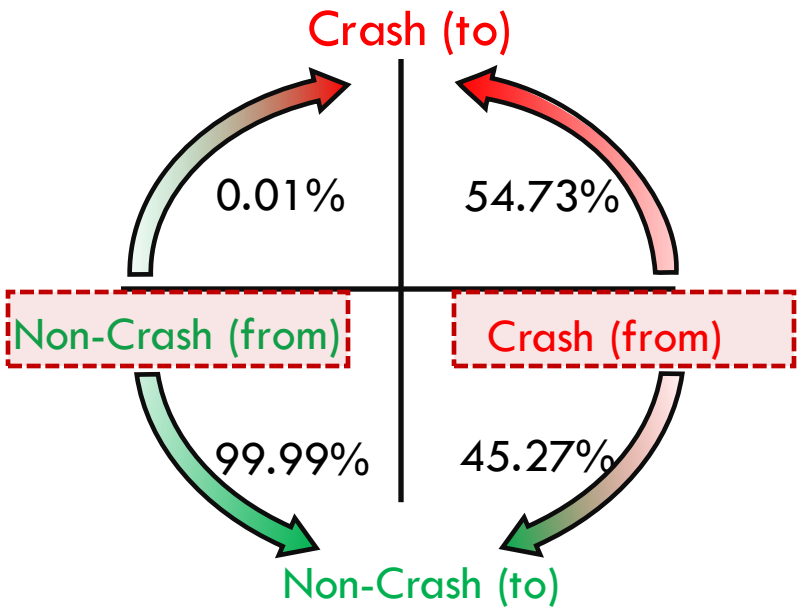


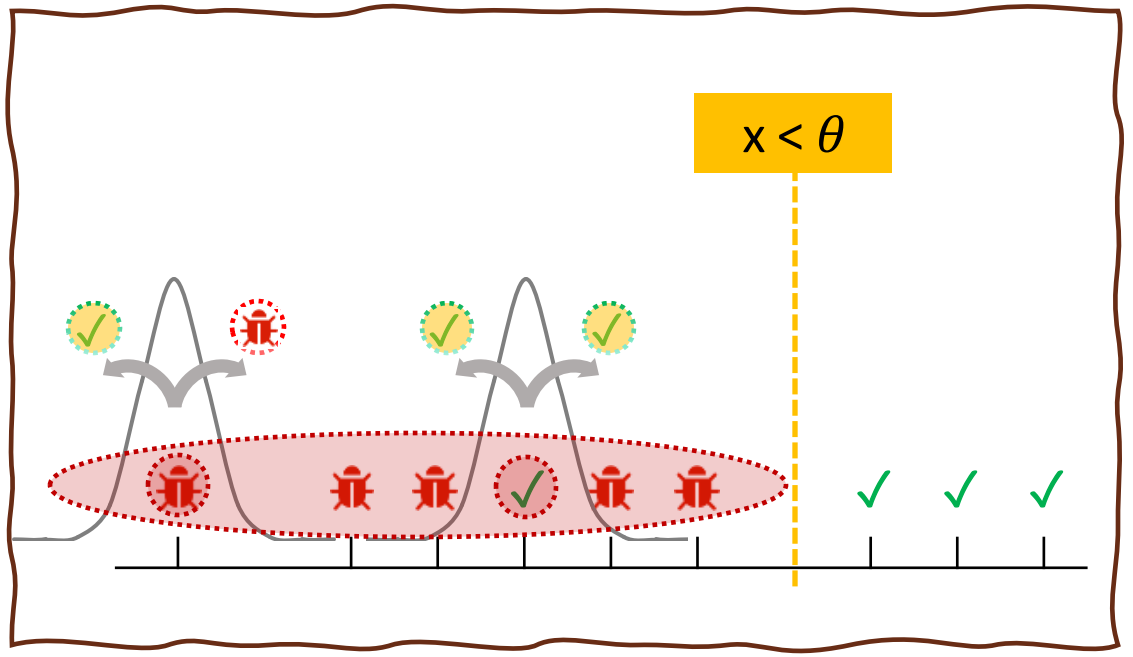
How to Generate Counterexamples?

Seed $V = a_1$

Mutated Input $V = a_2$

$|a_1 - a_2| \longrightarrow 0$





How to Generate Counterexamples?

p_1: offset < 1

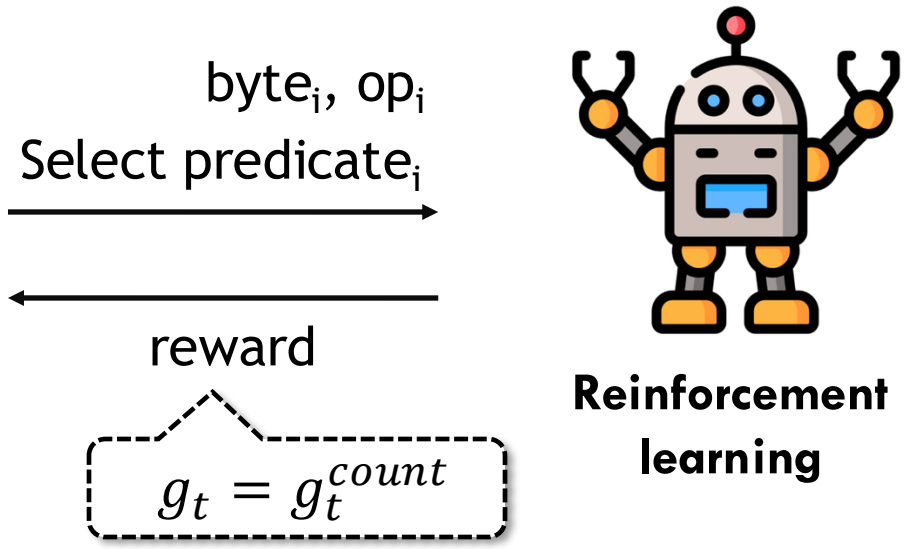
```
16197  while (offset <= sect->sh_size - sizeof (* eopt))
16198  {
16199    Elf_External_O
16200
16201    eoption = (Elf_External_Options *)((char *) eopt + offset);
16202
16203    option->kind = BYTE_GE  ion->kind);
16204    option->size = BYTE_GE  ion->size);
16205
16216    offset += option->size;
16217    ++option;
16218  }
```

p_2: sect->sh_size < 8

p_3: eopt > 0x4000000

...

p_n: option > 0x4000000



The count of the predicates being violated

How to Generate Counterexamples?

- Counterexample for Ranking, CoR

predicate	correlation
p_1: x>10	1.00
p_2: y<8	0.95 ↓
p_3: size<32	0.93
p_4: i>0	0.90

CoP_1 ←

predicate	correlation
p_1: x>10	1.00
p_2: y<8	0.97
p_3: size<32	0.93
p_4: i>0	0.90

CoP_2 →

predicate	correlation
p_1: x>10	1.00
p_3: size<32	0.93
p_2: y<8	0.91 ↓
p_4: i>0	0.90

$$g_t = g_t^{count} + g_t^{order}$$

Evaluation Dataset: 30 vulnerabilities

python

libjpeg-turbo

mruby

libzip

patch

libpng

Libxml2

sleuthkit

libtiff

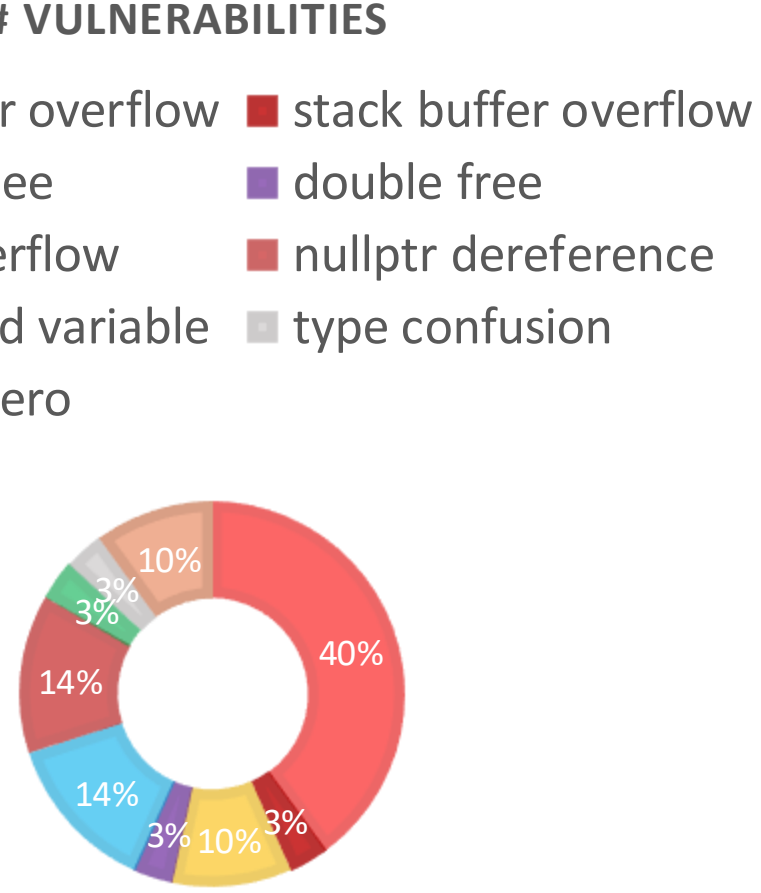
nasm the netwide assembler

BASH THE BOURNE-AGAIN SHELL

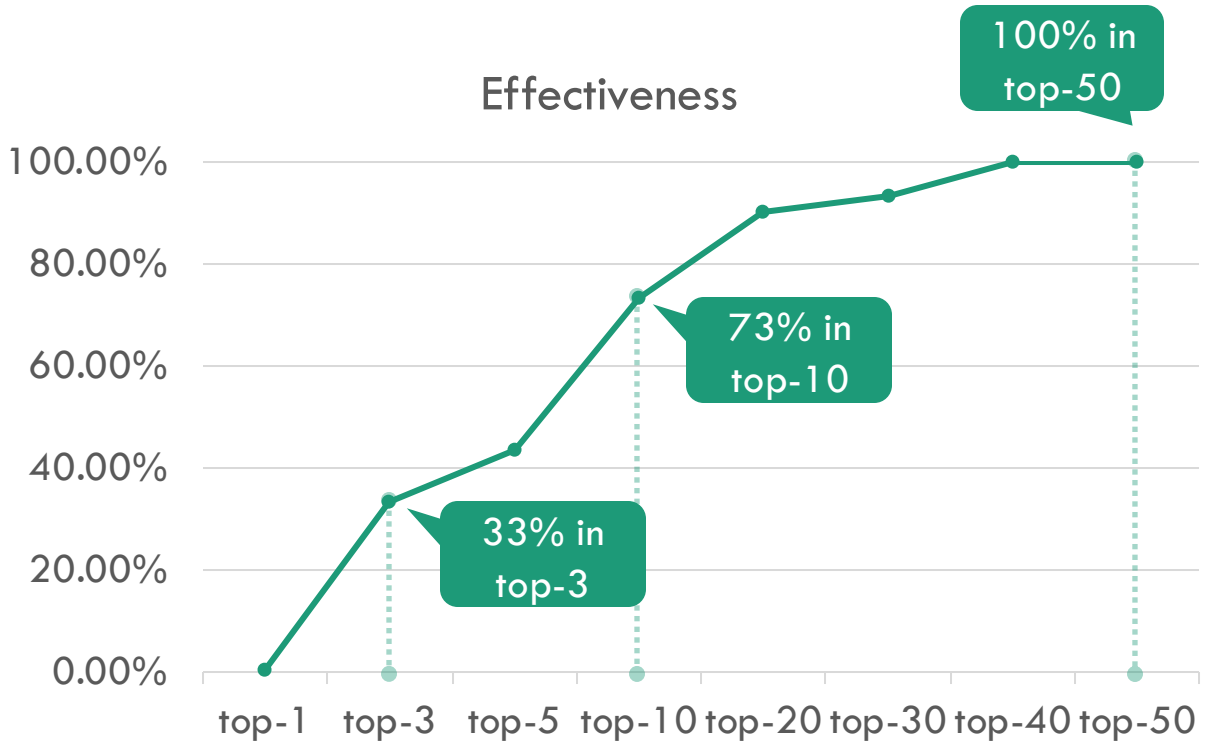
PROGRAMMING REPUBLIC OF PERL

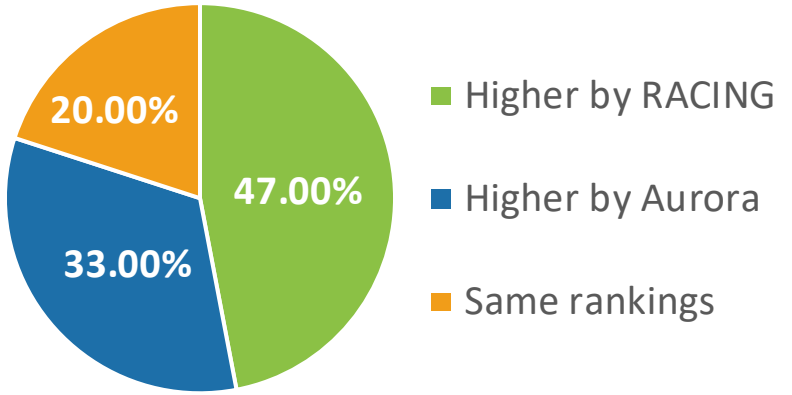
TCPDUMP

libsixel



Evaluation Results I: Effectiveness



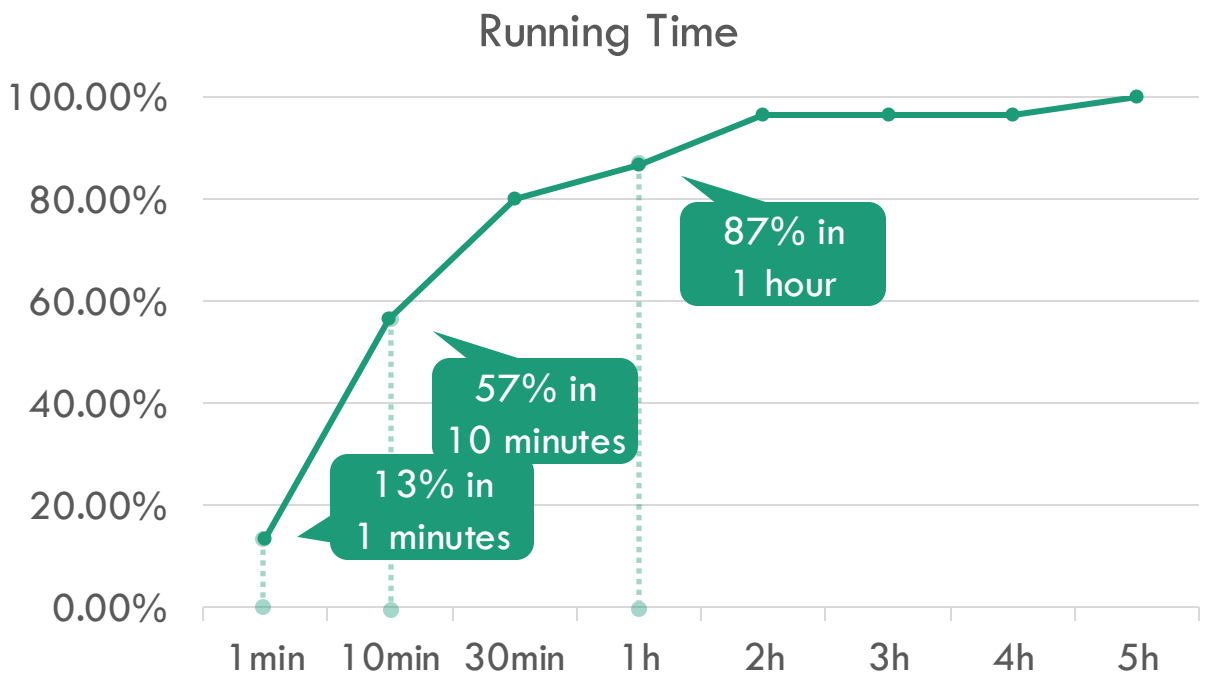


	RaCing	Aurora
nasm the netwide assembler	20	58

	RaCing	Aurora
Average	9.9	13.7

3.8 positions higher

Evaluation Results II: Efficiency

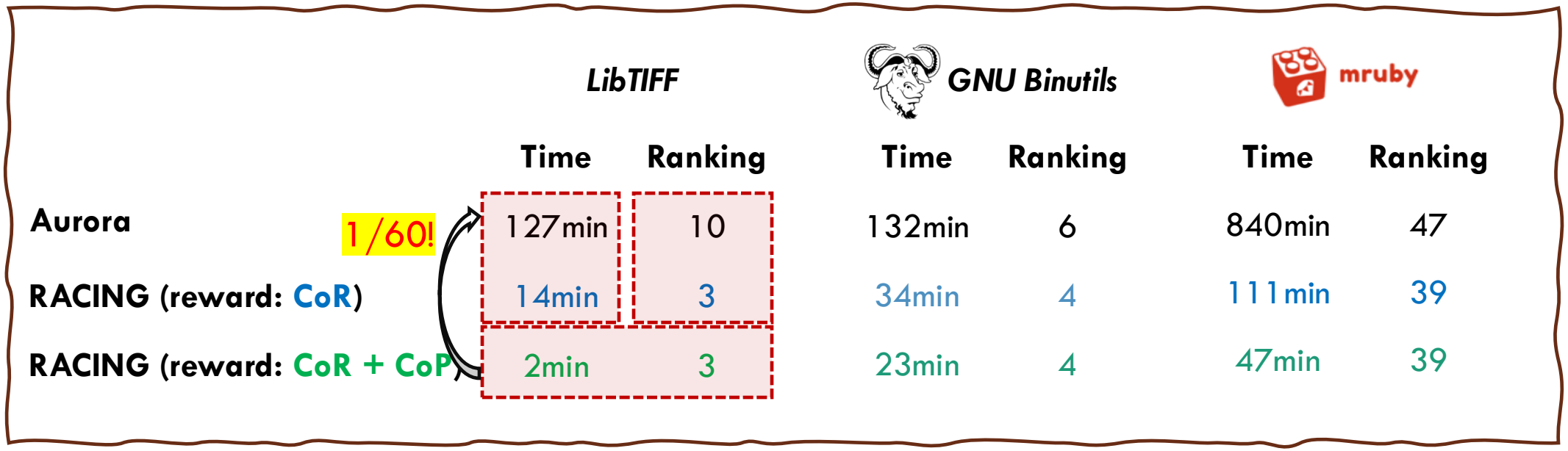


Speedup: 1.2× ~ 602.4×

	RaCing	Aurora
Average	27 minutes	6 hours

13x faster!

Evaluation Results III: Performance of CoRs & CoPs



Conclusion

New understanding

- Counterexamples

New technique

- RL-enhanced

Good results

- 13x faster↑

Thanks for your attention!

Q&A

Paper

Code

https://github.com/0xdd96/Racing-code