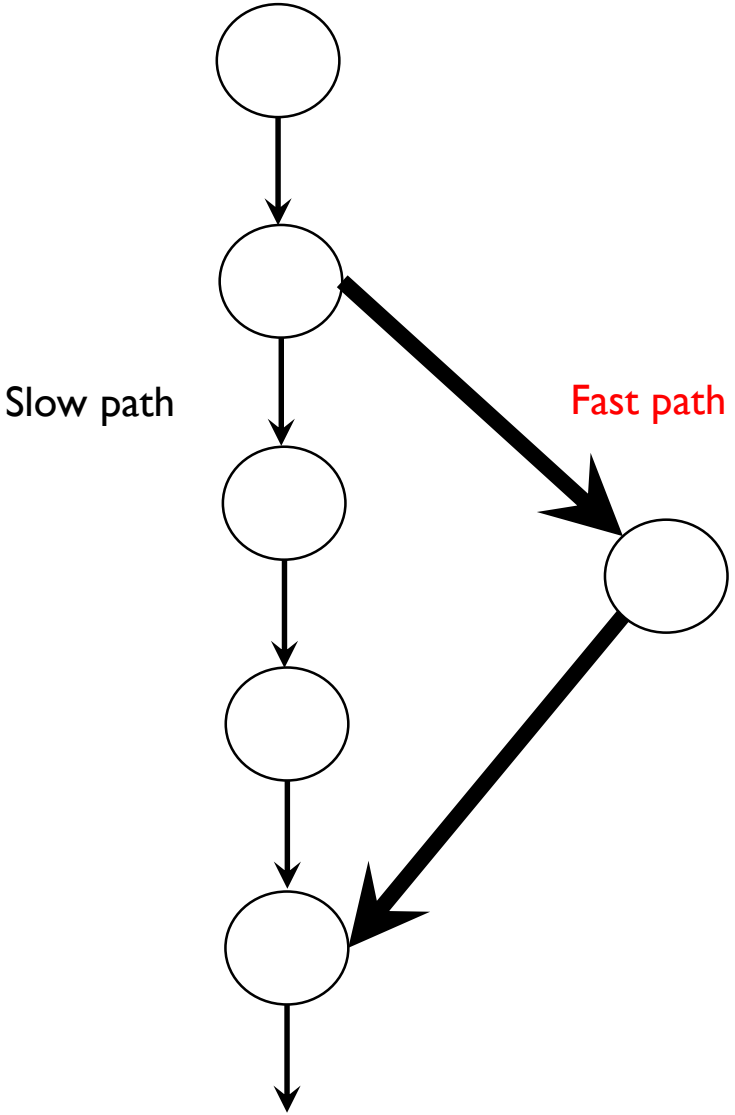
Pallas: Semantic-Aware Checking for Finding Deep Bugs in Fast Path

Jian Huang † Michael Allen-Bond **Xuechen Zhang**

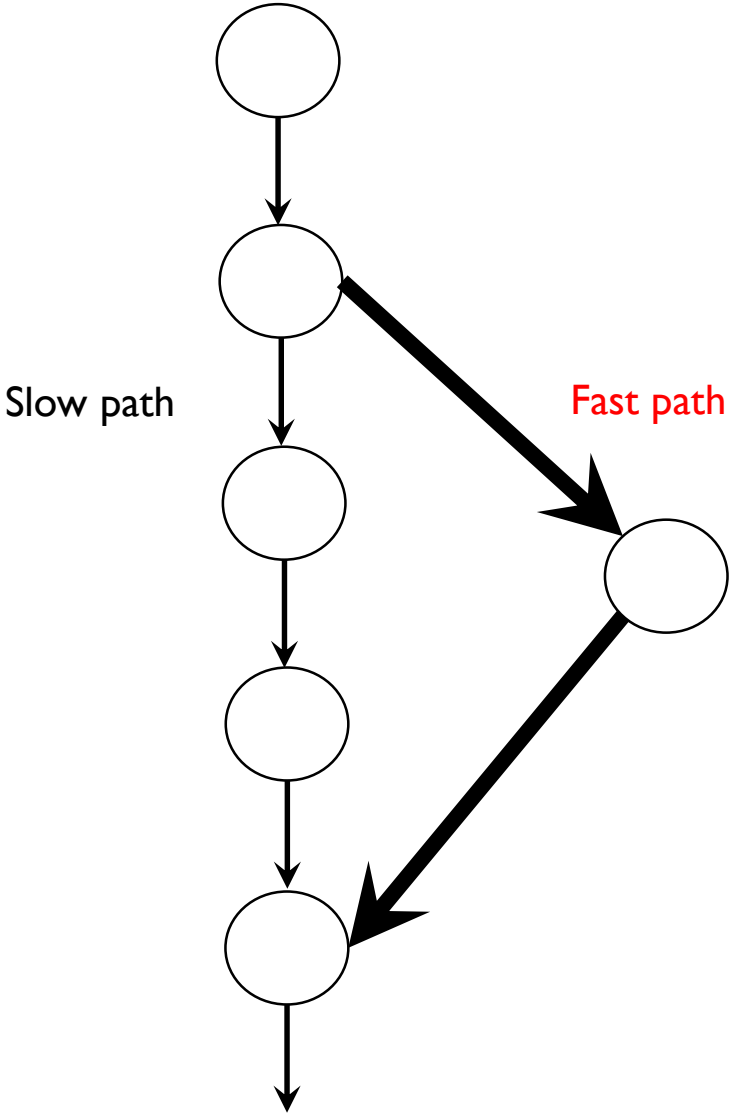
† Georgia Institute of Technology

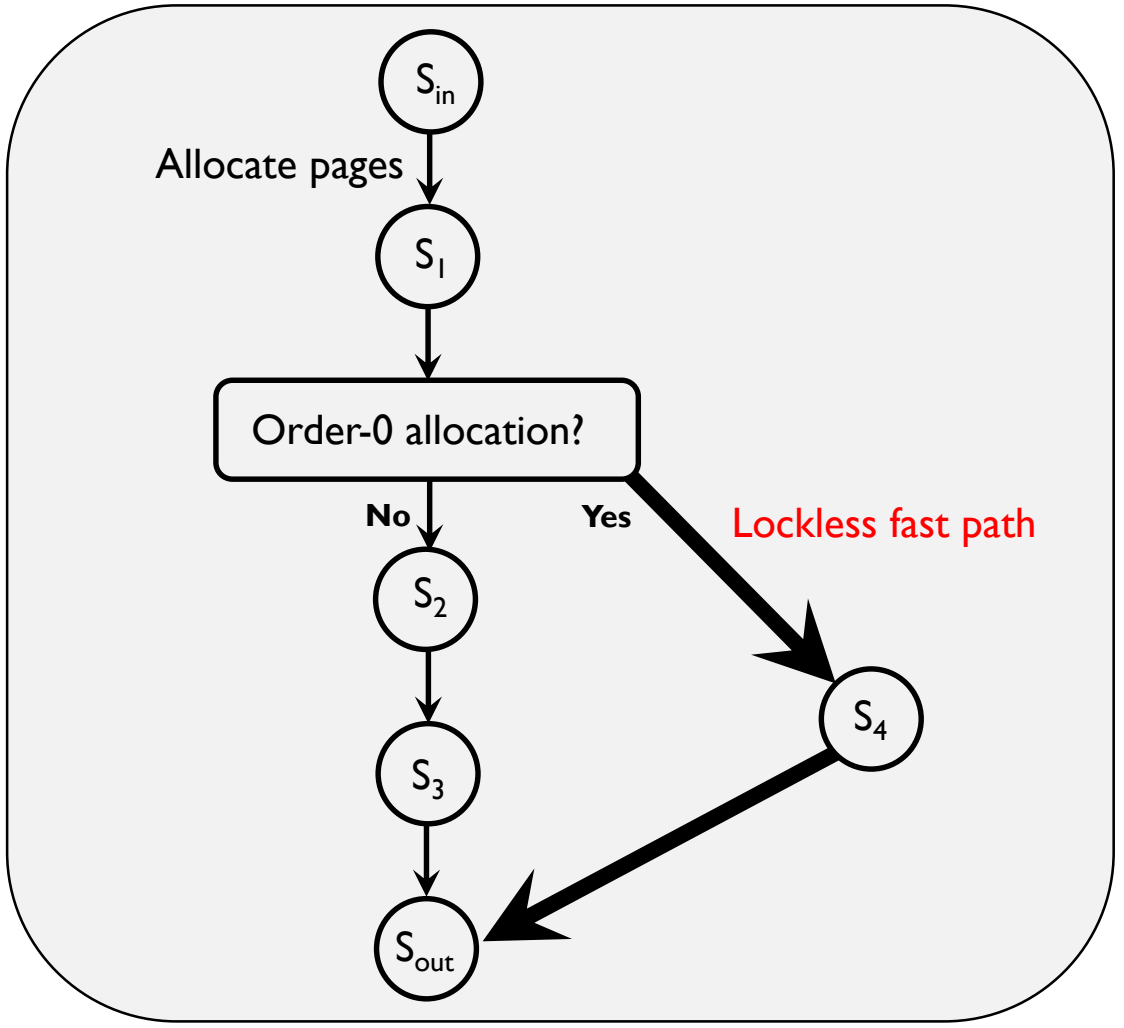
Washington State University

What is Fast Path?



What is Fast Path?

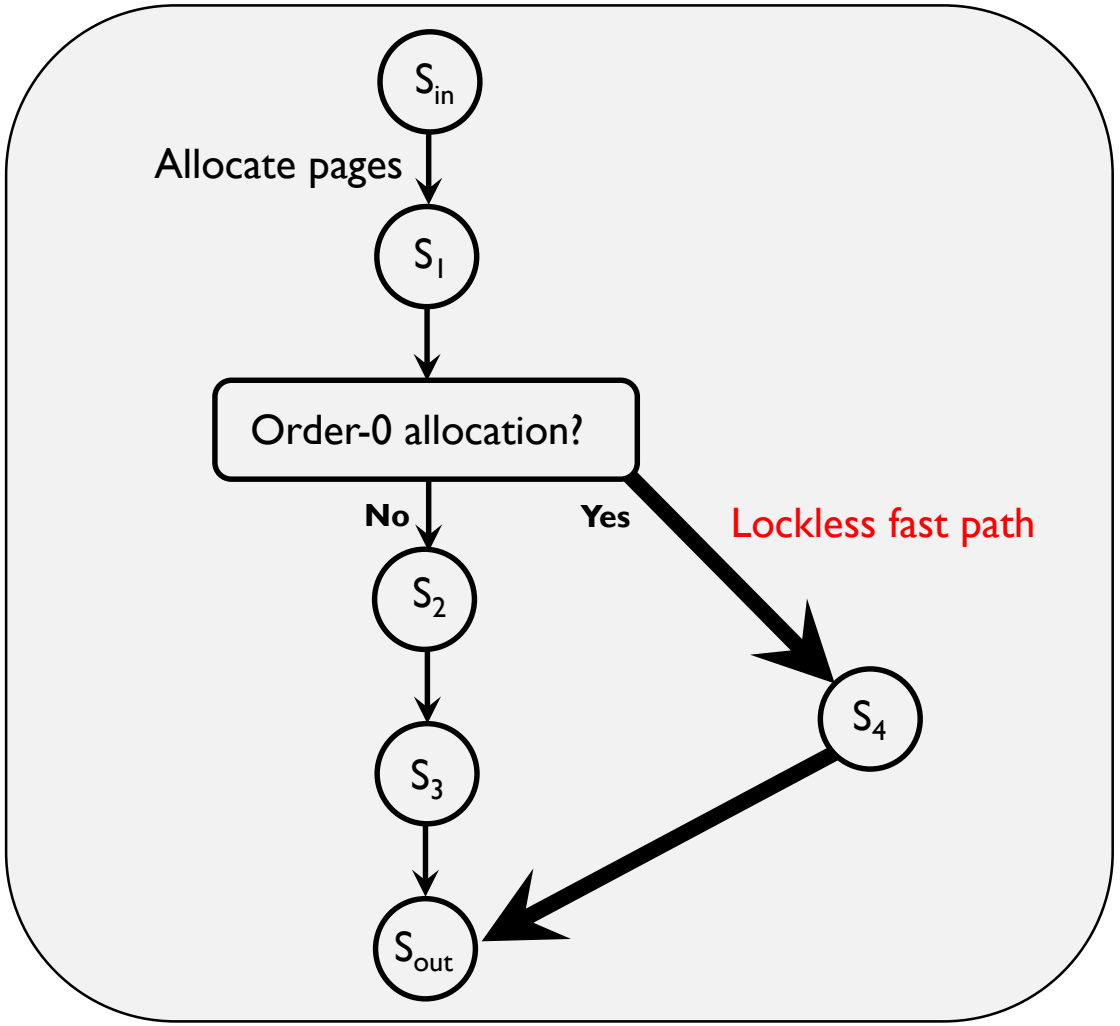




What is Fast Path?

Fast path is derived from slow path or vice versa

They share the start and end entries in the workflow



Fast Path is Everywhere

Memory allocation

Fast Path is Everywhere

Operating system

Web browser

Memory allocation

Web page loading

Fast Path is Everywhere

Operating system

Memory allocation

Web browser

Web page loading

Mobile OS

File system inode search

Fast Path is Everywhere

Operating system

Memory allocation

Web browser

Web page loading

Mobile OS

File system inode search

Software-defined network

Packet forwarding

Fast Path Introduces Semantic Bugs

	Memory manager	File system	Network	Device driver
# of fast paths	16	21	14	14
# of bugs per path	19	17	11	5
Fix time (days)	3	8	5	12

Fast Path Introduces Semantic Bugs

	Memory manager	File system	Network	Device driver
# of fast paths	16	21	14	14
# of bugs per path	19	17	11	5
Fix time (days)	3	8	5	12

Fast-path bugs are related to software semantics and these bugs are hard to detect

Detecting Fast-Path Bugs is Challenging

Program Verification

seL4[SOSP'09], Ironclad[OSDI'14], etc

But systems may not have verification frameworks

Detecting Fast-Path Bugs is Challenging

Program Verification	seL4[SOSP'09], Ironclad[OSDI'14], etc But systems may not have verification frameworks
Model Checker	SAMC[OSDI'14], FiSC[OSDI'04], etc. But they require models for specific systems

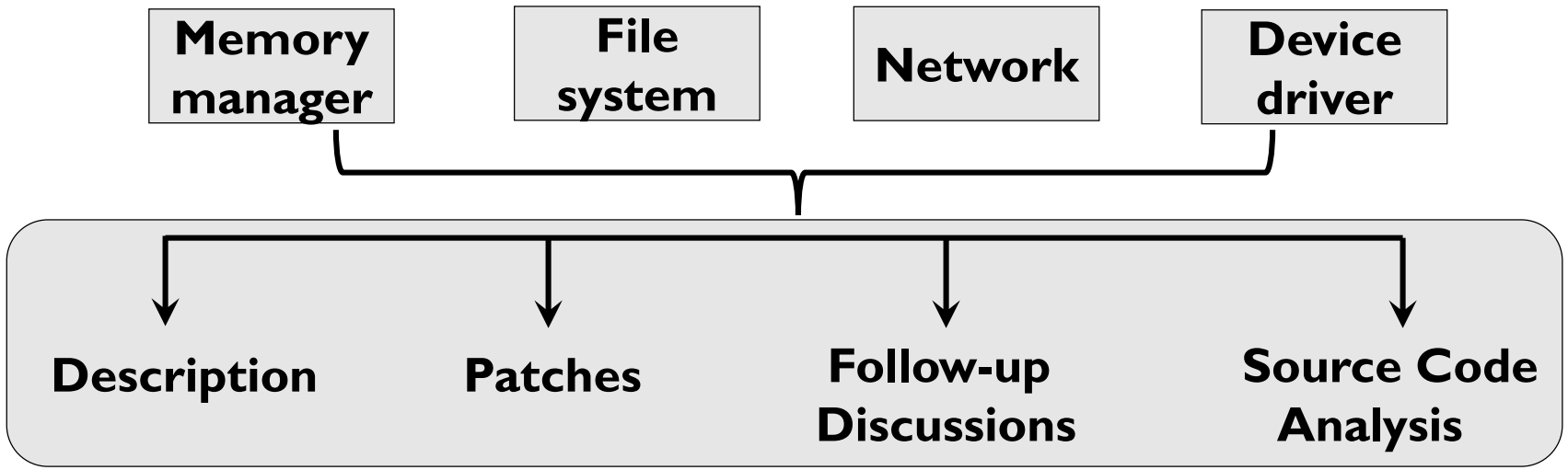
Detecting Fast-Path Bugs is Challenging

Program Verification	seL4[SOSP'09], Ironclad[OSDI'14], etc But systems may not have verification frameworks
Model Checker	SAMC[OSDI'14], FiSC[OSDI'04], etc. But they require models for specific systems

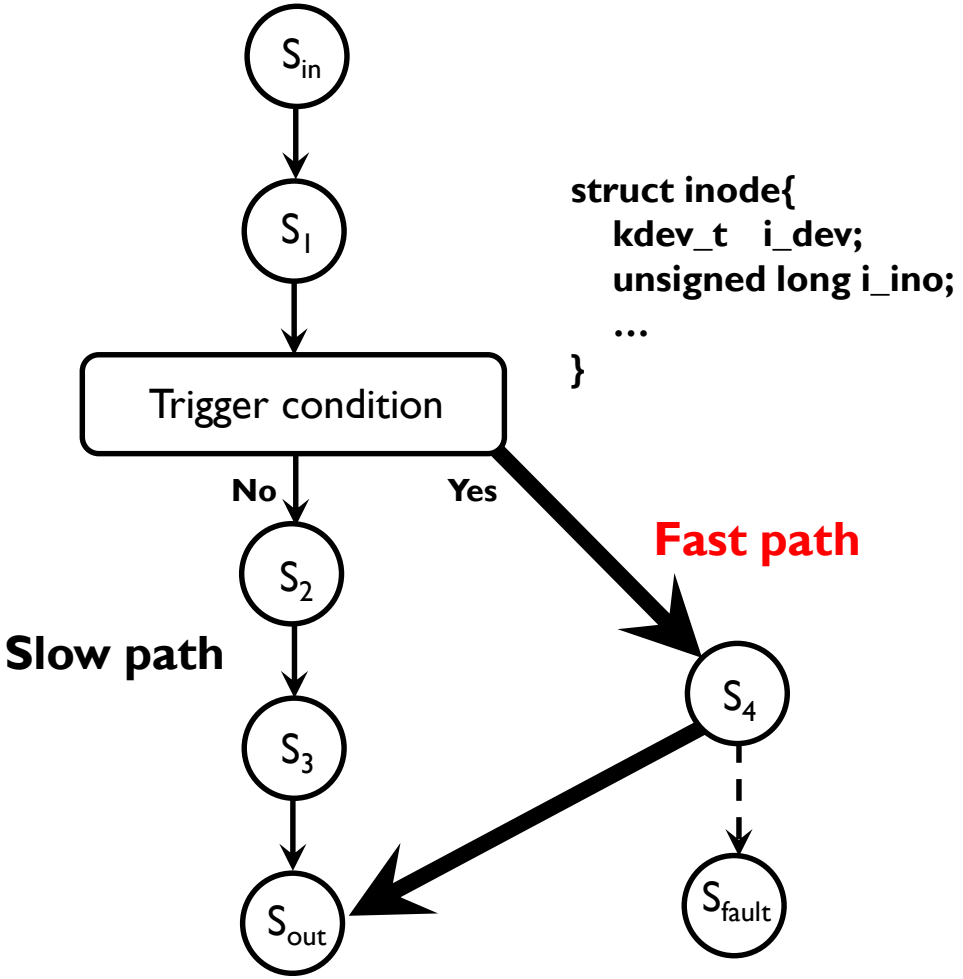
Can we use static analysis to detect fast-path bugs?

Our Study on Fast-Path Bugs in Linux

- Sampled 65 fast paths and 172 relevant patches
- Committed from 2009-2015



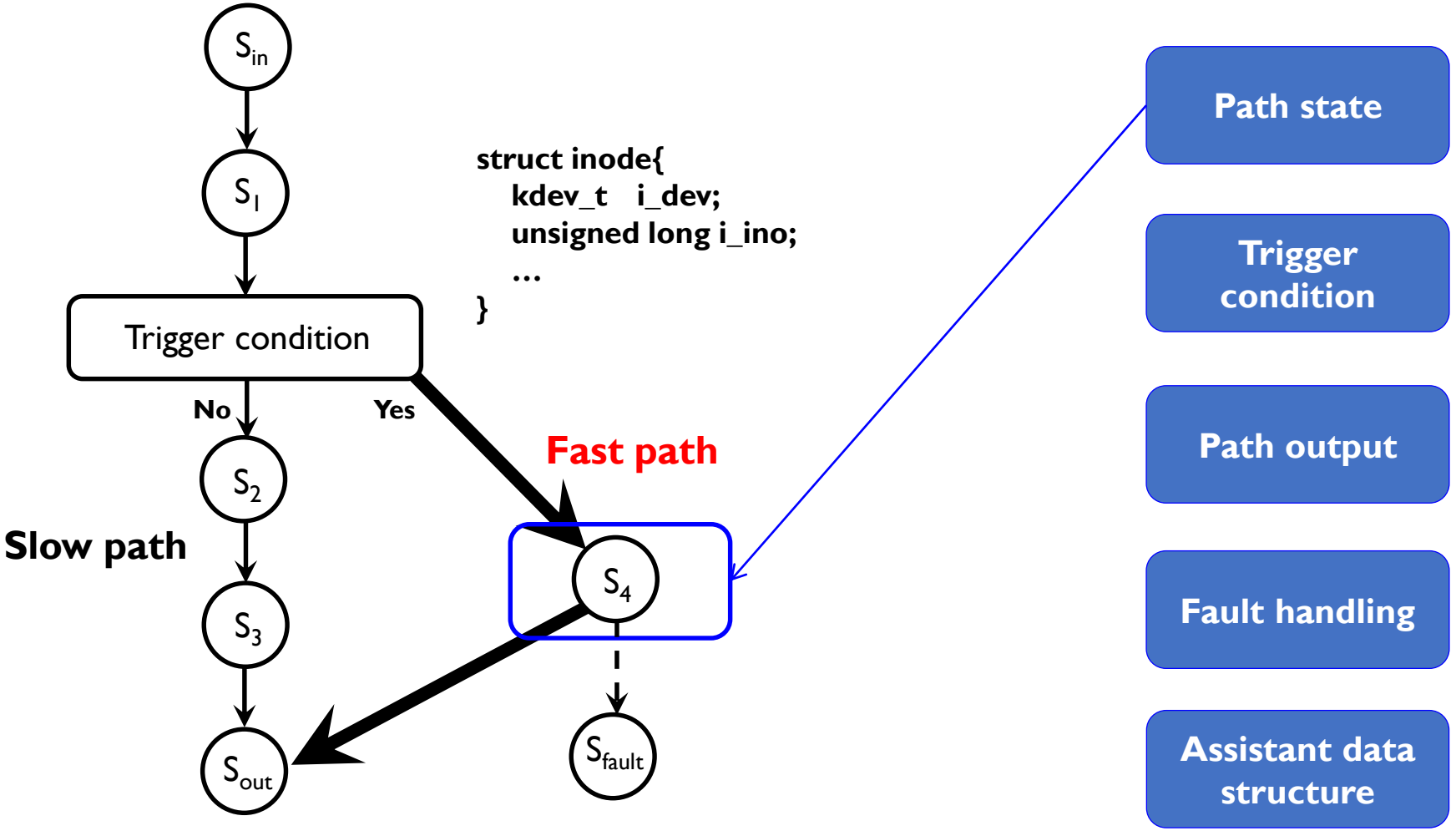
Fast-Path Bug Categorization



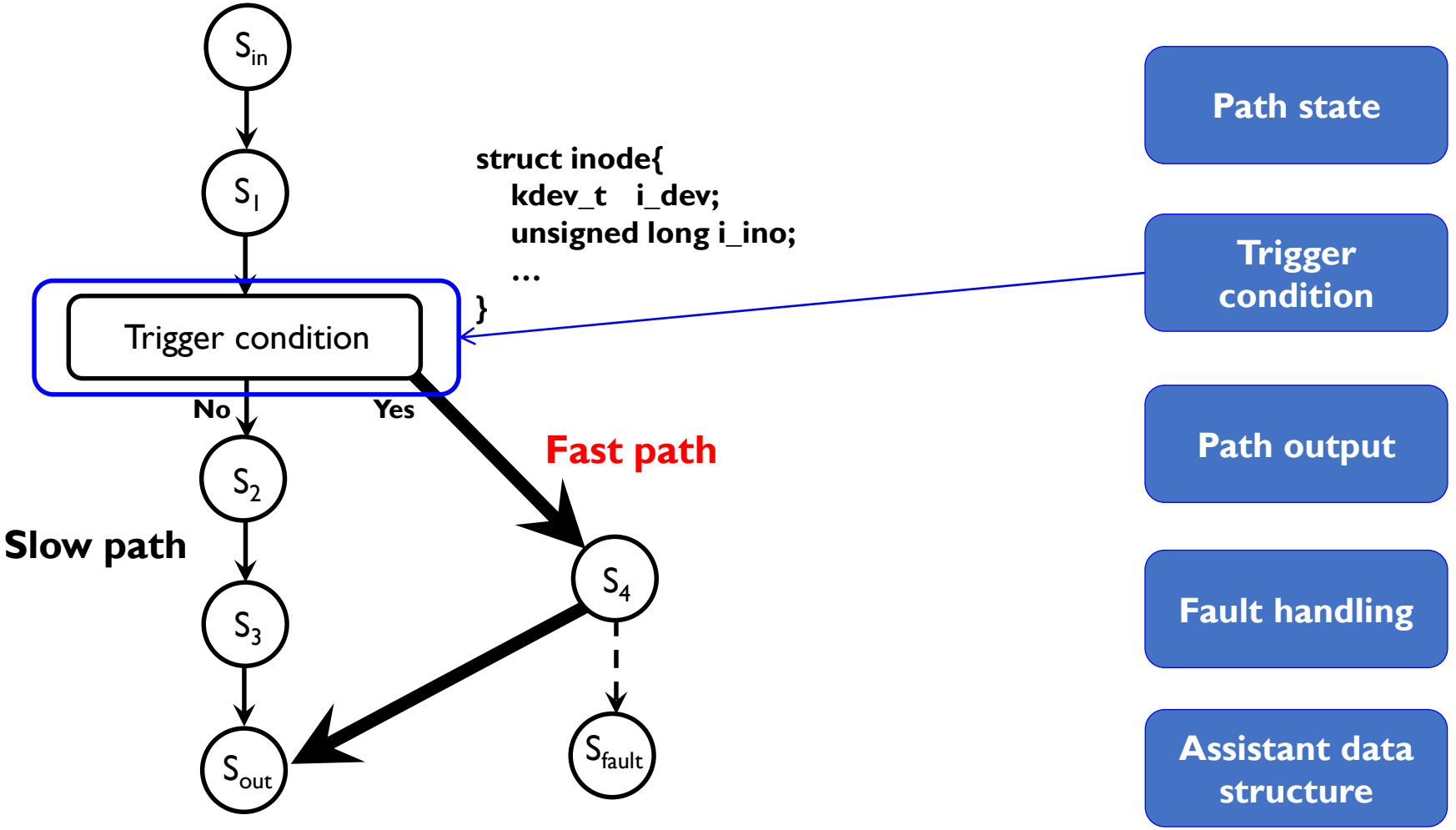

```
struct inode{
    kdev_t    i_dev;
    unsigned long i_ino;
    …
}
```

- Path state
- Trigger condition
- Path output
- Fault handling
- Assistant data structure

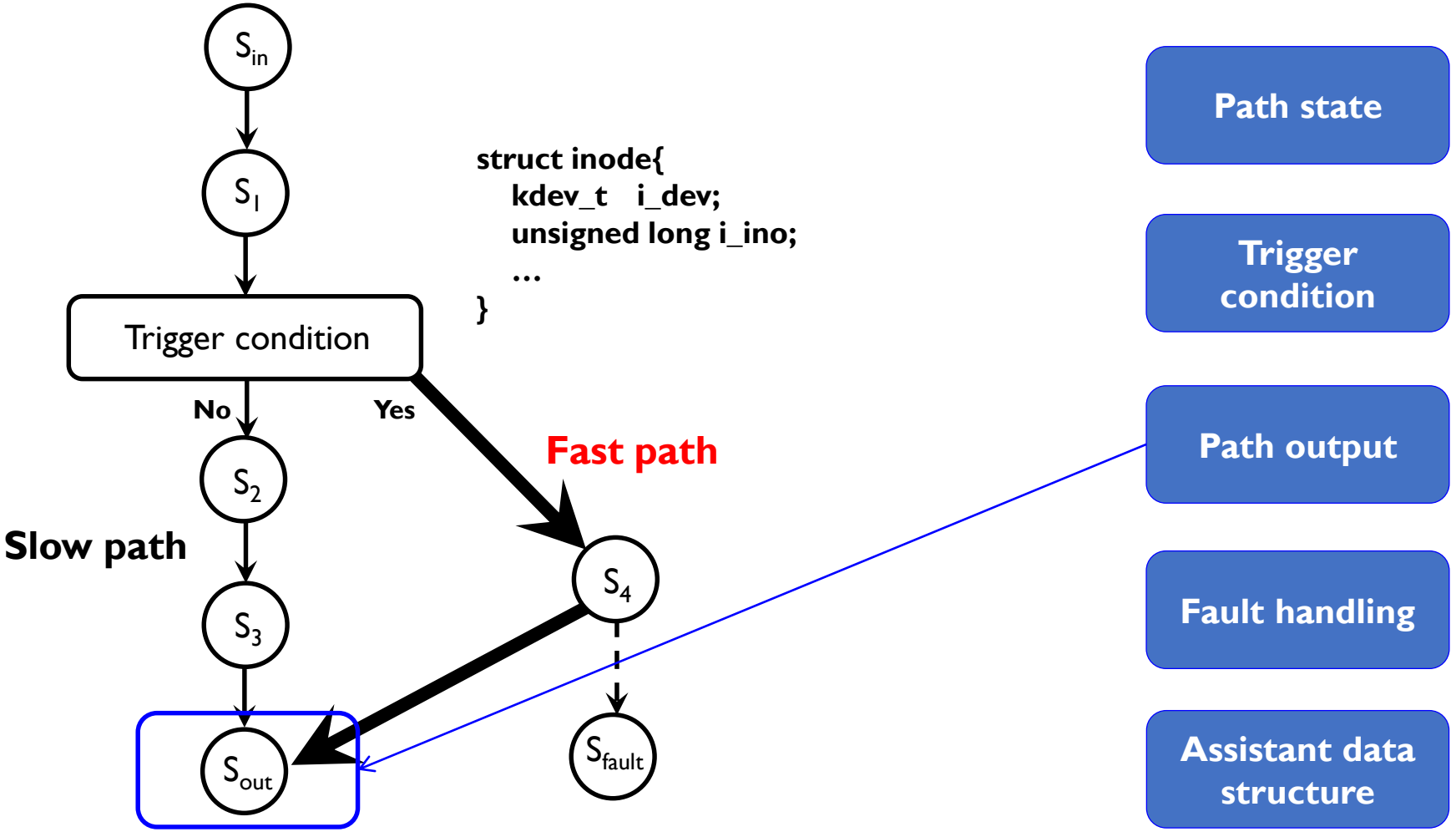
Fast-Path Bug Categorization



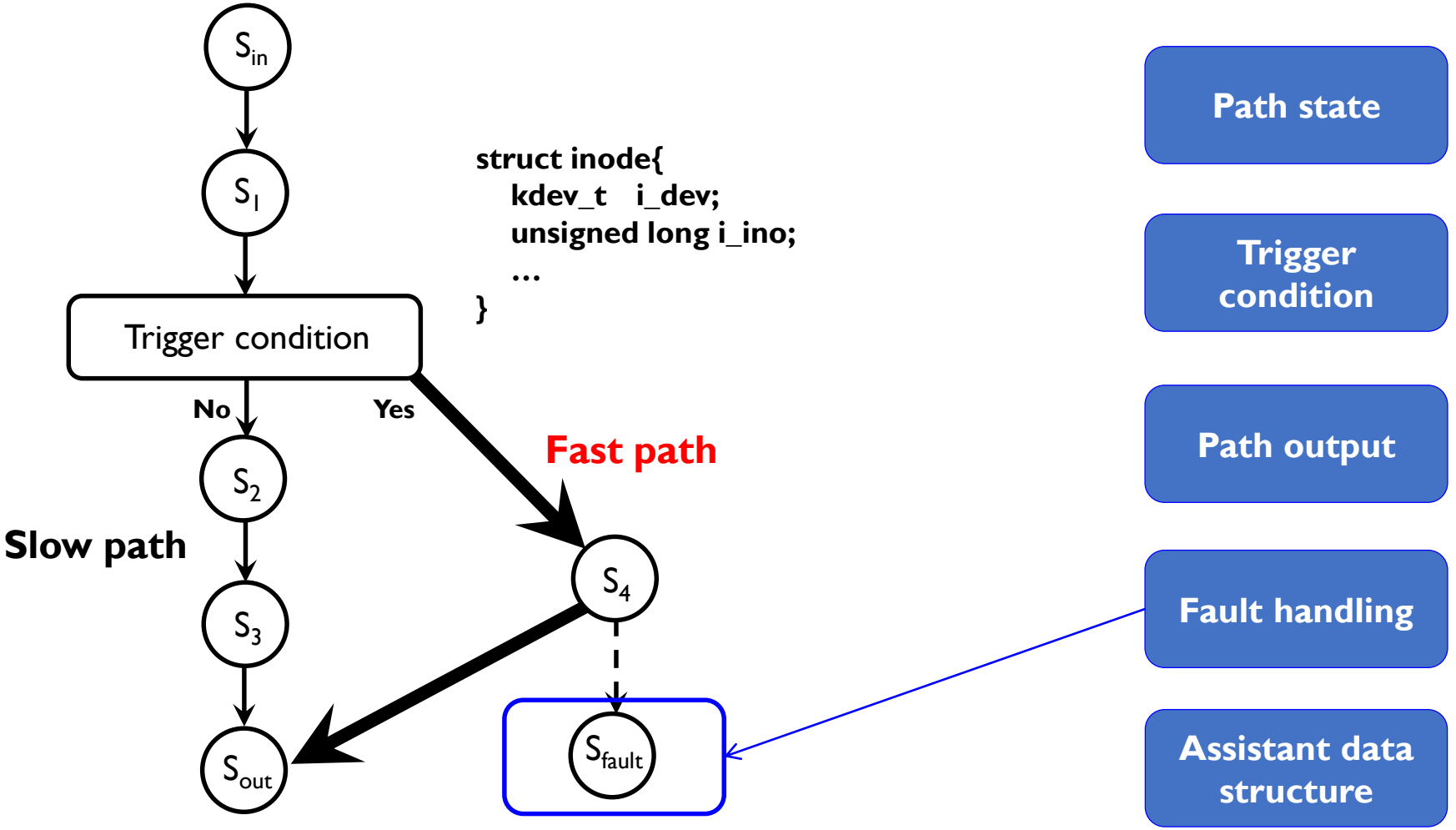
Fast-Path Bug Categorization



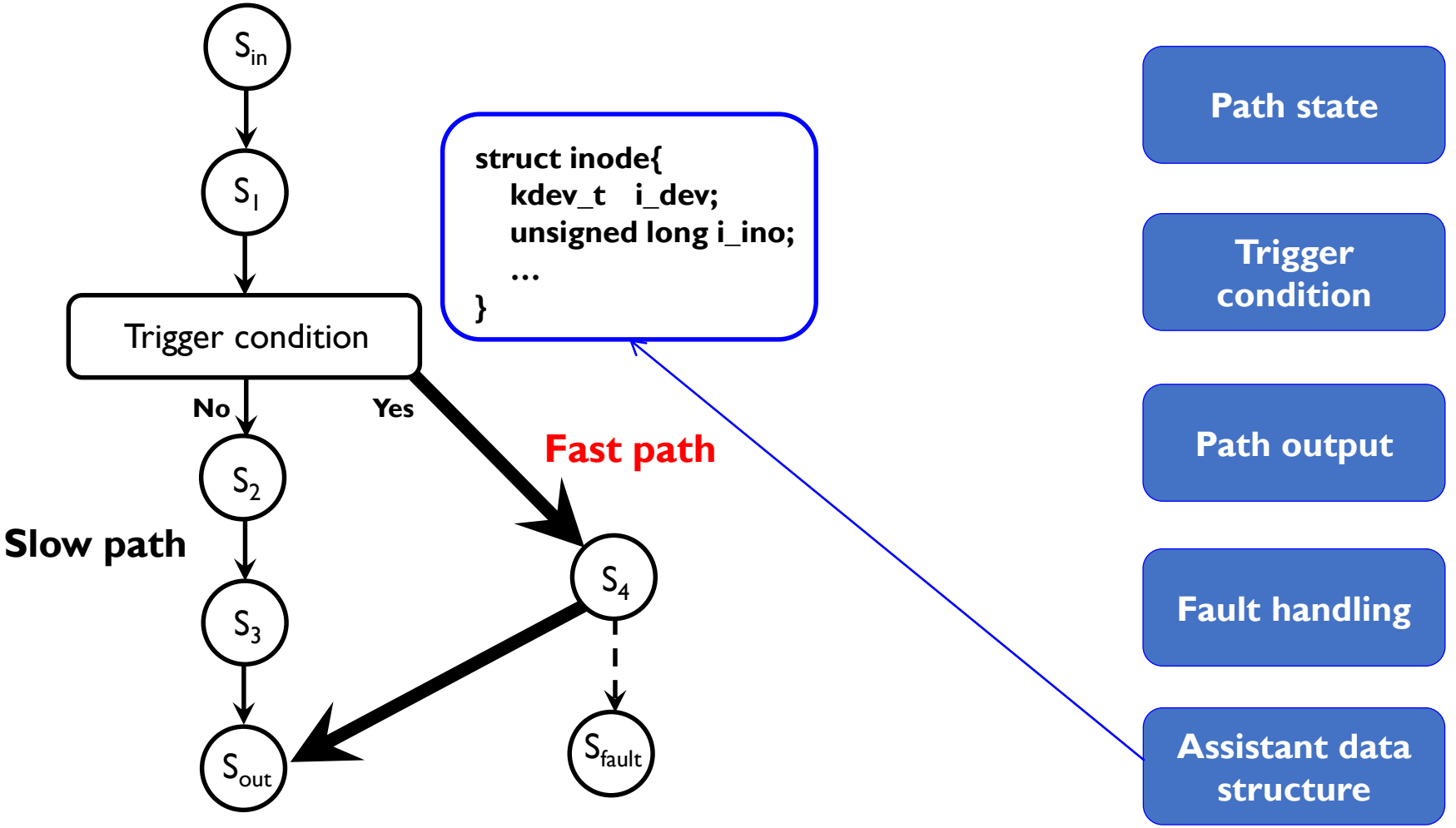
Fast-Path Bug Categorization



Fast-Path Bug Categorization



Fast-Path Bug Categorization

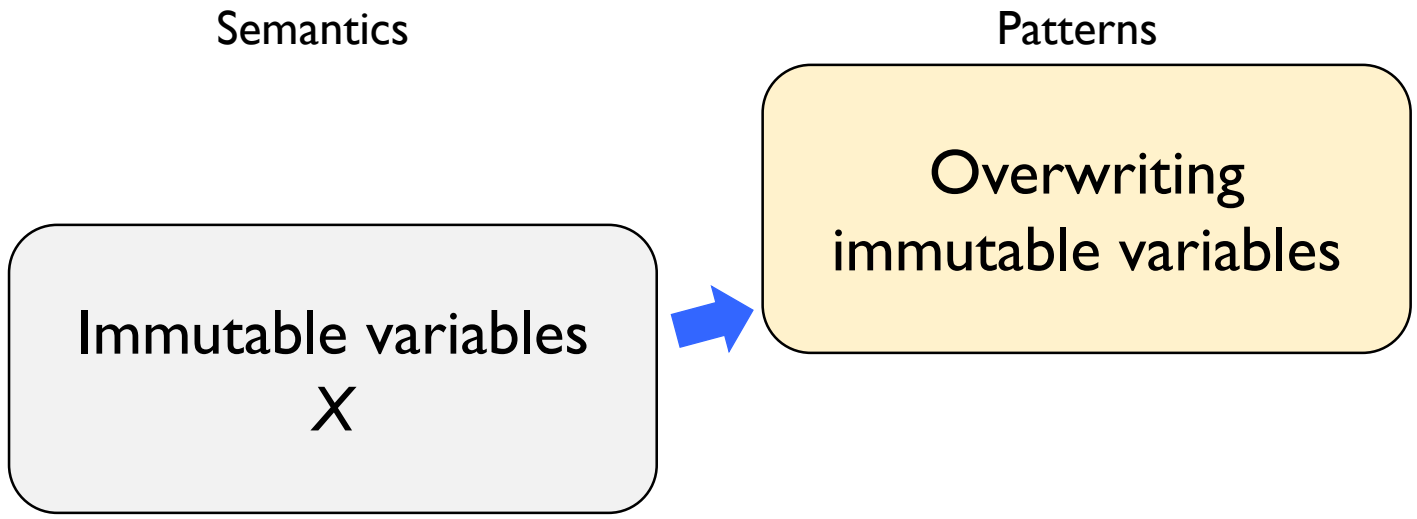


How does Path State Cause Bugs?

Semantics

Immutable variables
X

How does Path State Cause Bugs?



How does Path State Cause Bugs?

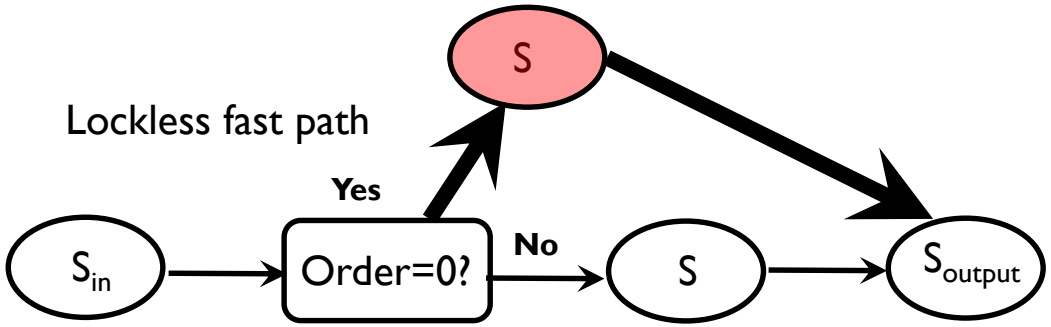
Semantics

Immutable variables
X

Patterns

Overwriting
immutable variables

```
+   if (order == 0) {
+       …
+       gfp_mask = memalloc_noio_flags(gfp_mask);
+       …
+   } else {
```



How does Path State Cause Bugs?

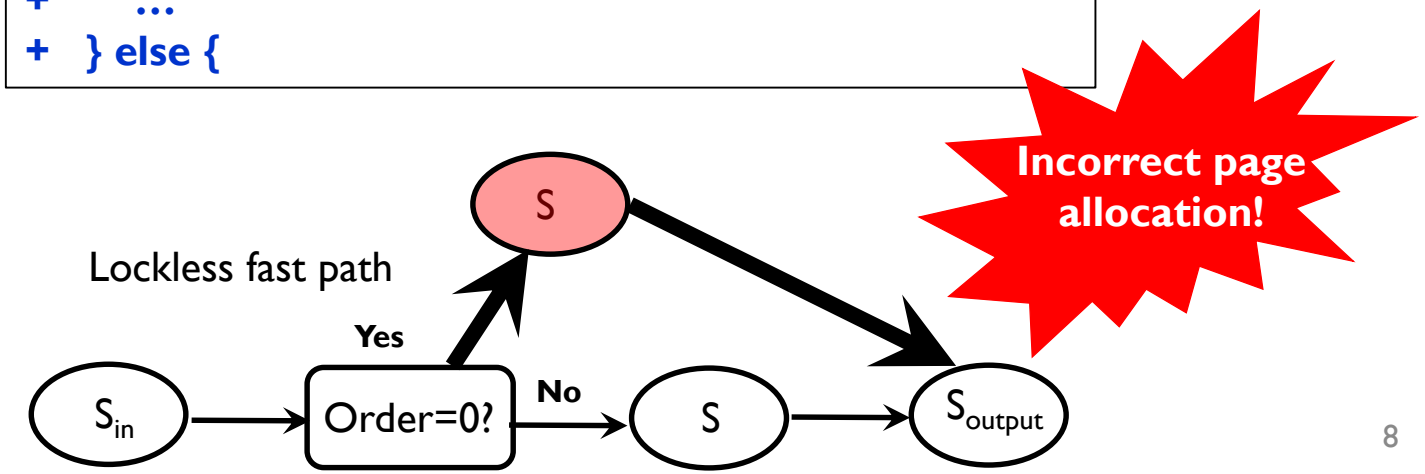
Semantics

Patterns

Immutable variables
X

Overwriting
immutable variables

```
+   if (order == 0) {
+       …
+       gfp_mask = memalloc_noio_flags(gfp_mask);
+       …
+   } else {
```



How does Path State Cause Bugs?

Semantics

Immutable variables
X

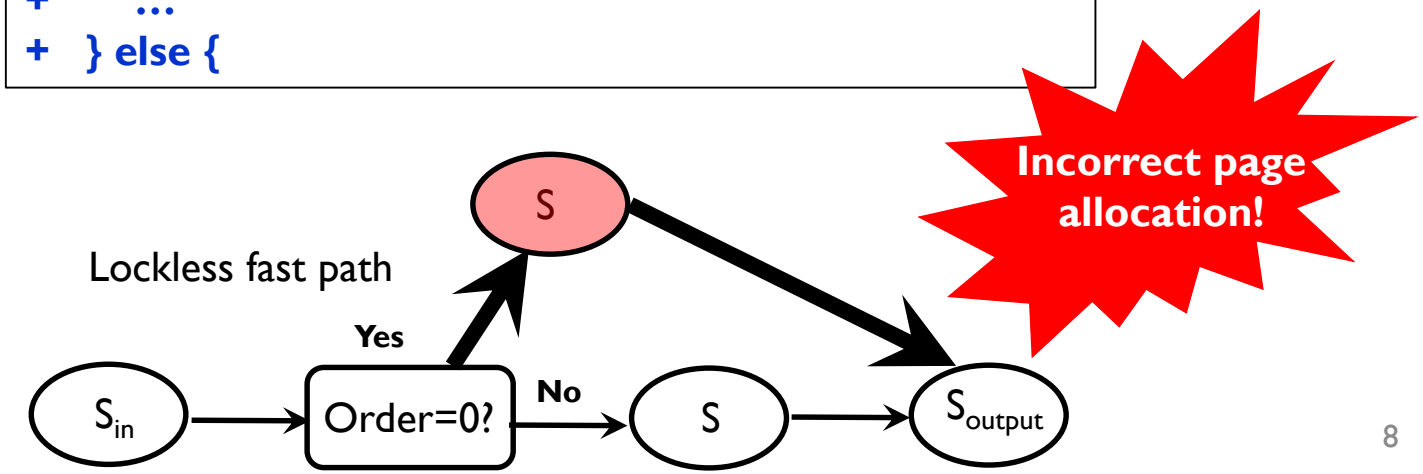
Patterns

Overwriting
immutable variables

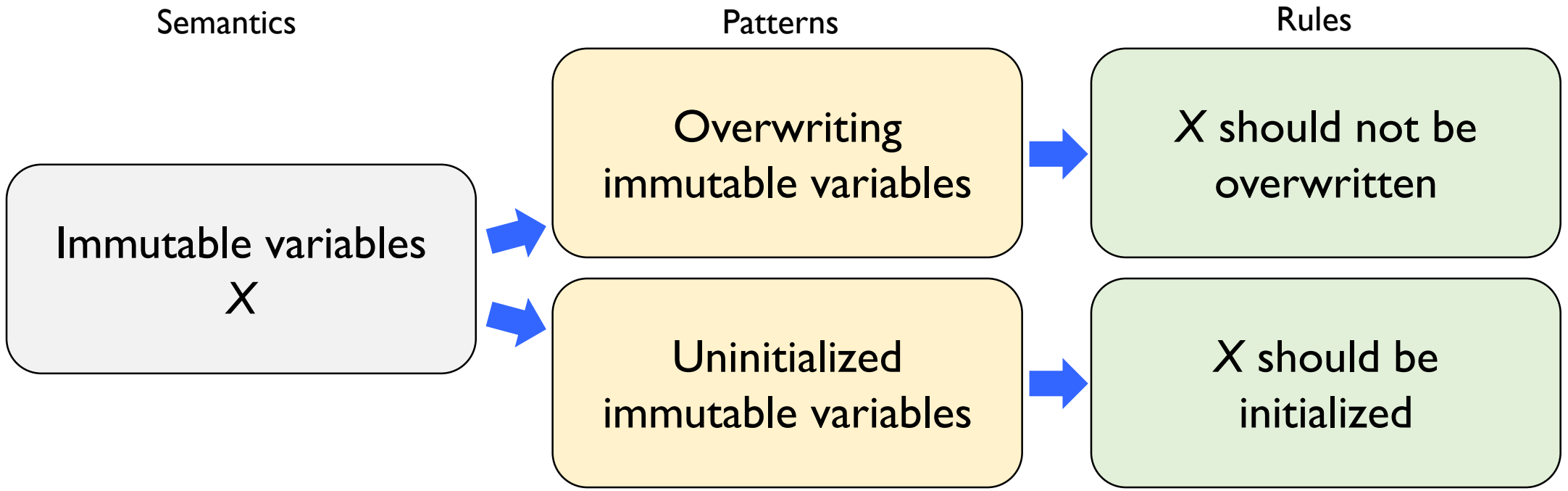
Rules

X should not be
overwritten

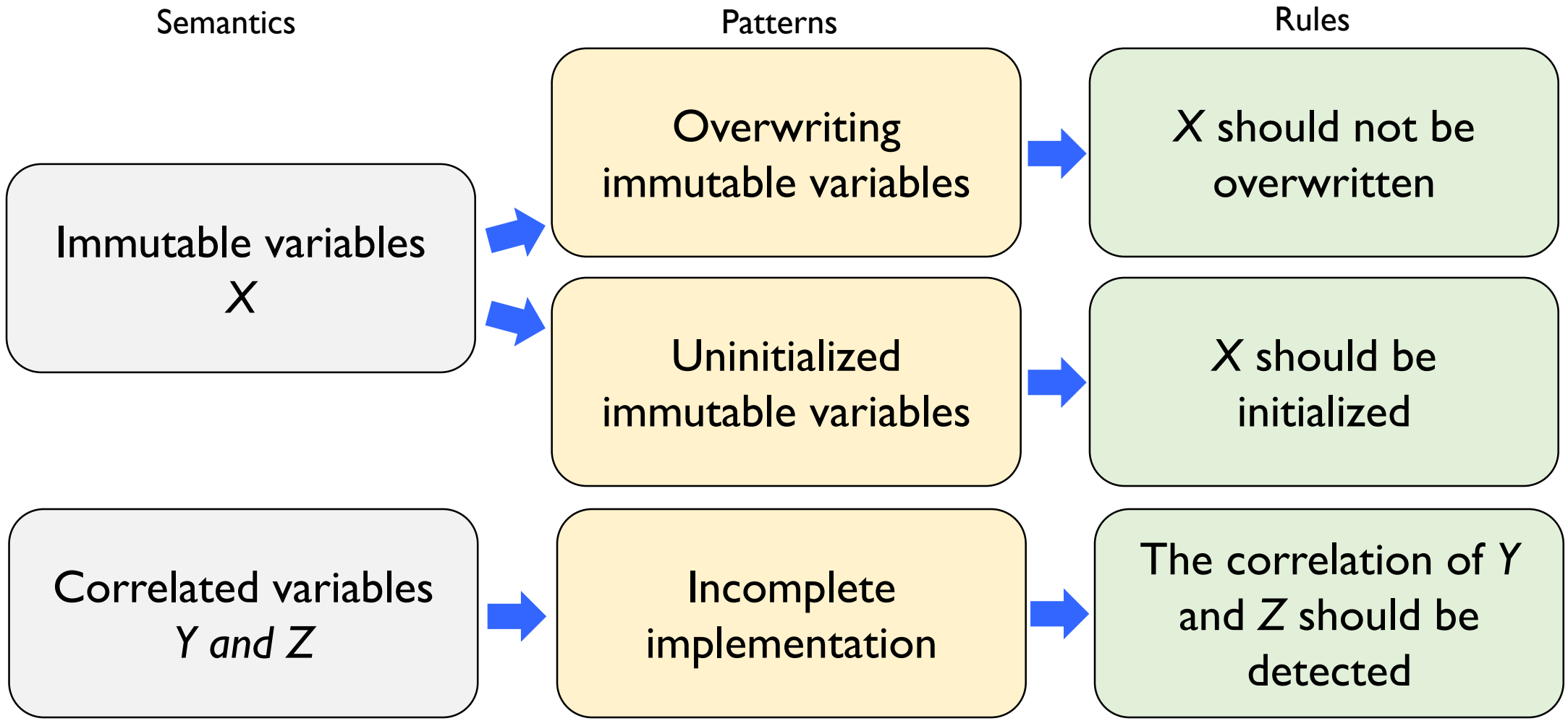
```
+   if (order == 0) {
+       …
+       gfp_mask = memalloc_noio_flags(gfp_mask);
+       …
+   } else {
```



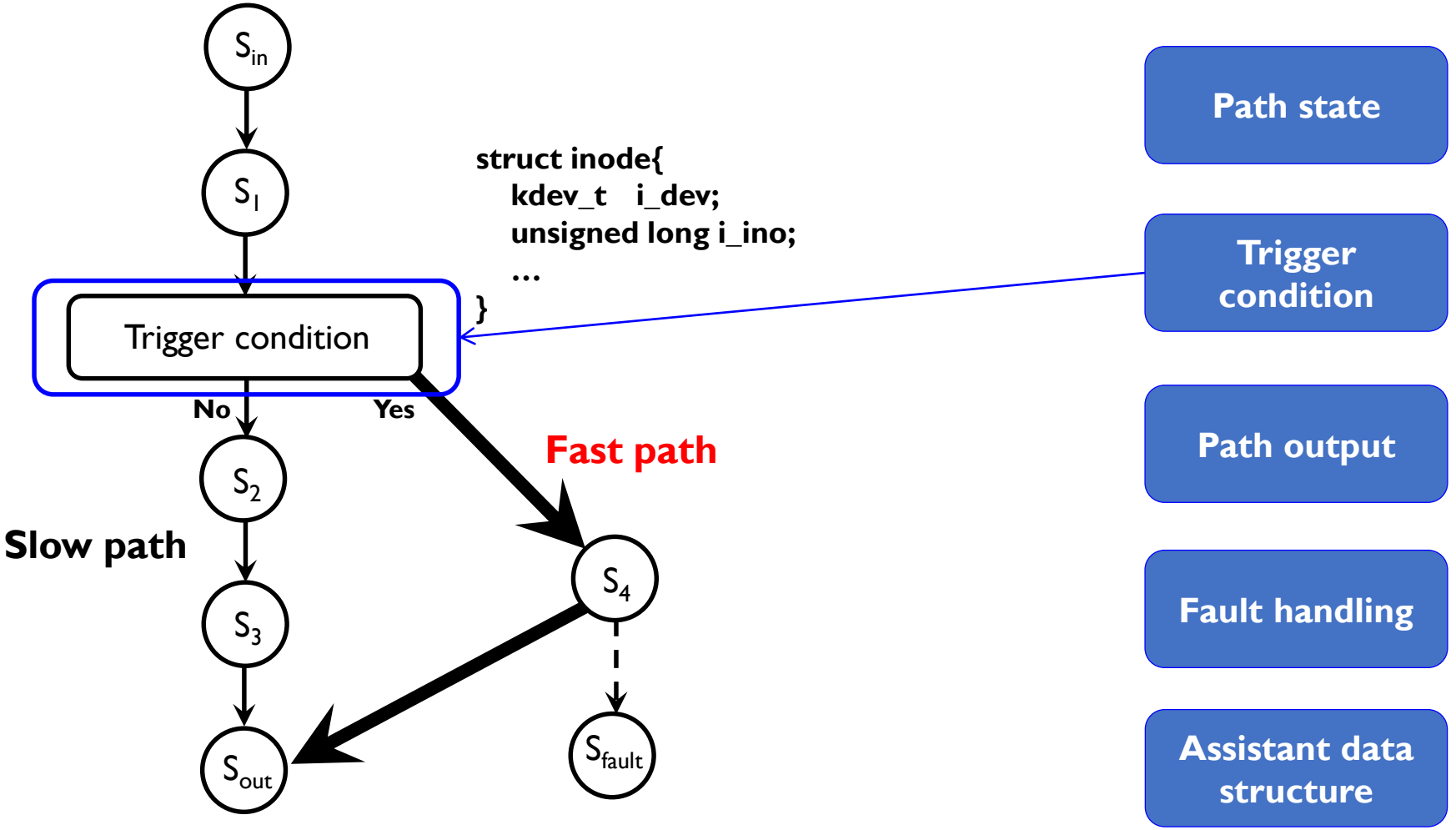
How does Path State Cause Bugs?



How does Path State Cause Bugs?



Fast-Path Bug Categorization

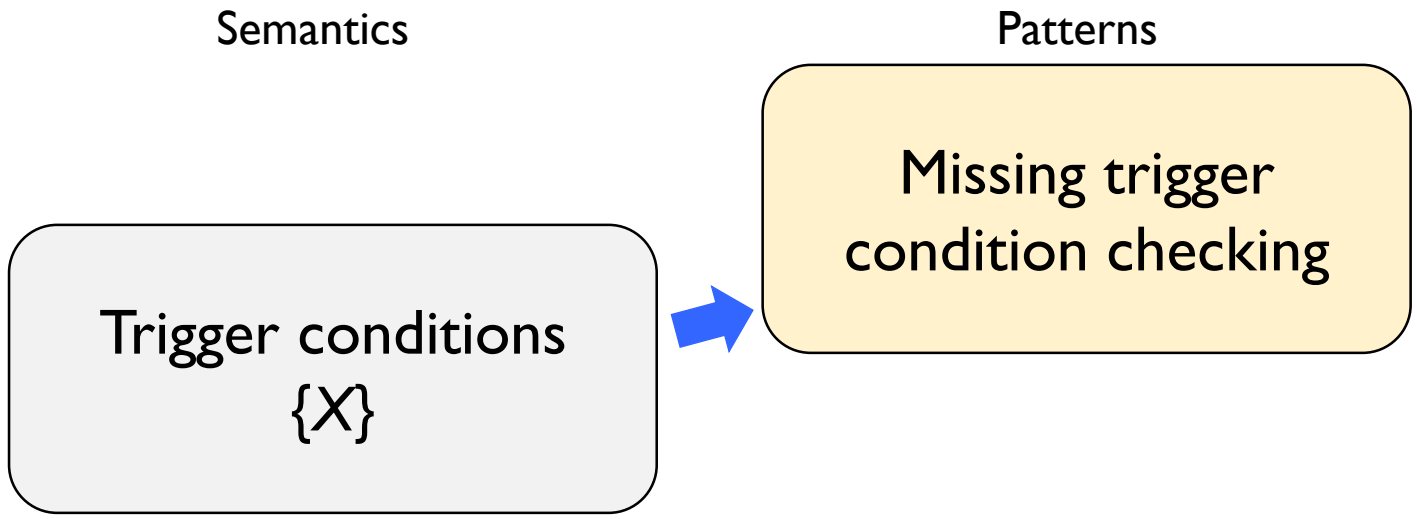


How does Trigger Condition Cause Bugs?

Semantics

Trigger conditions
{X}

How does Trigger Condition Cause Bugs?



How does Trigger Condition Cause Bugs?

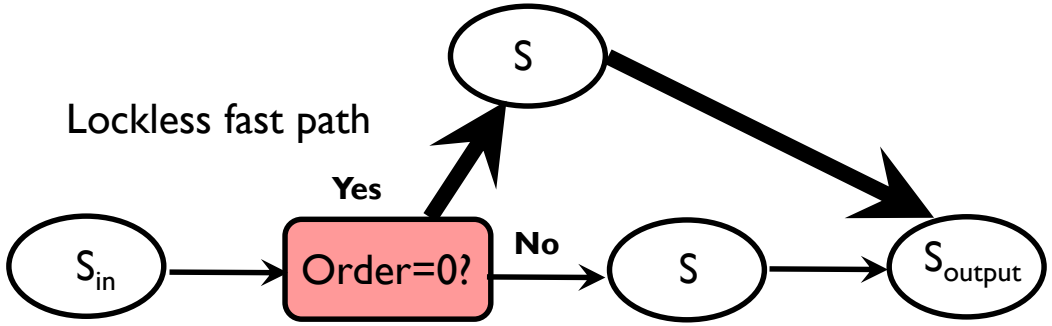
Semantics

Patterns

Trigger conditions
{X}

Missing trigger
condition checking

```
+   if (order == 0) {
+       …
+       page  = list_last_entry(list, struct page, lru);
+       …
+   } else {
```



How does Trigger Condition Cause Bugs?

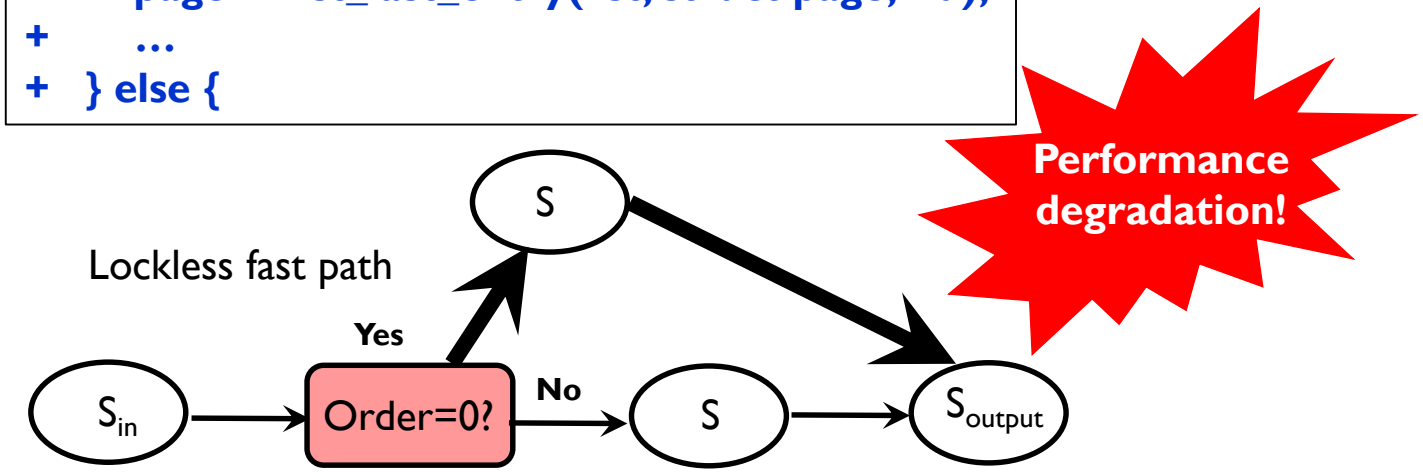
Semantics

Patterns

Trigger conditions
{X}

Missing trigger
condition checking

```
+   if (order == 0) {
+       …
+       page  = list_last_entry(list, struct page, lru);
+       …
+   } else {
```



How does Trigger Condition Cause Bugs?

Semantics

Patterns

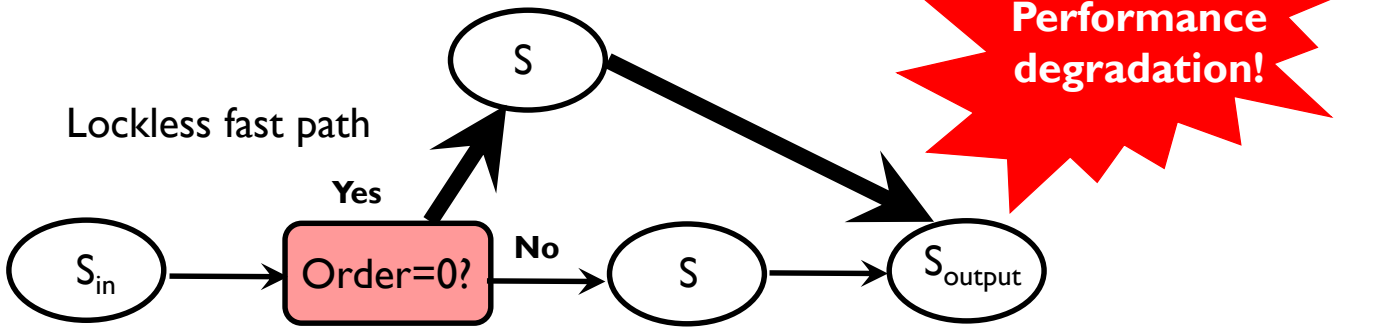
Rules

Trigger conditions {X}

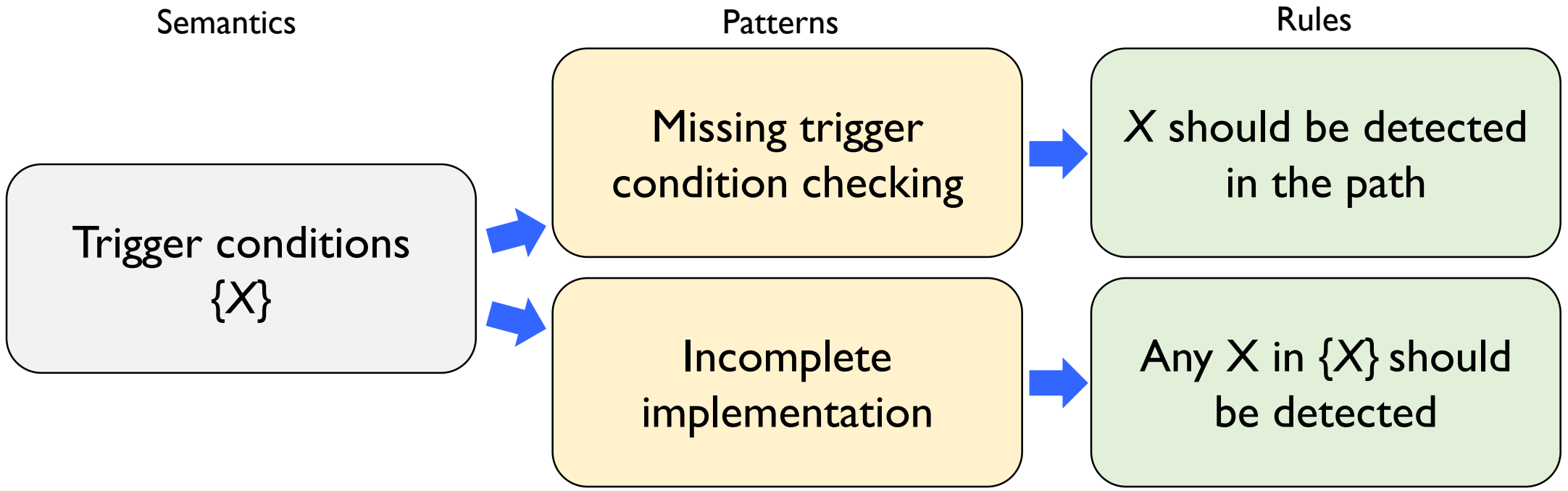
Missing trigger condition checking

X should be detected in the path

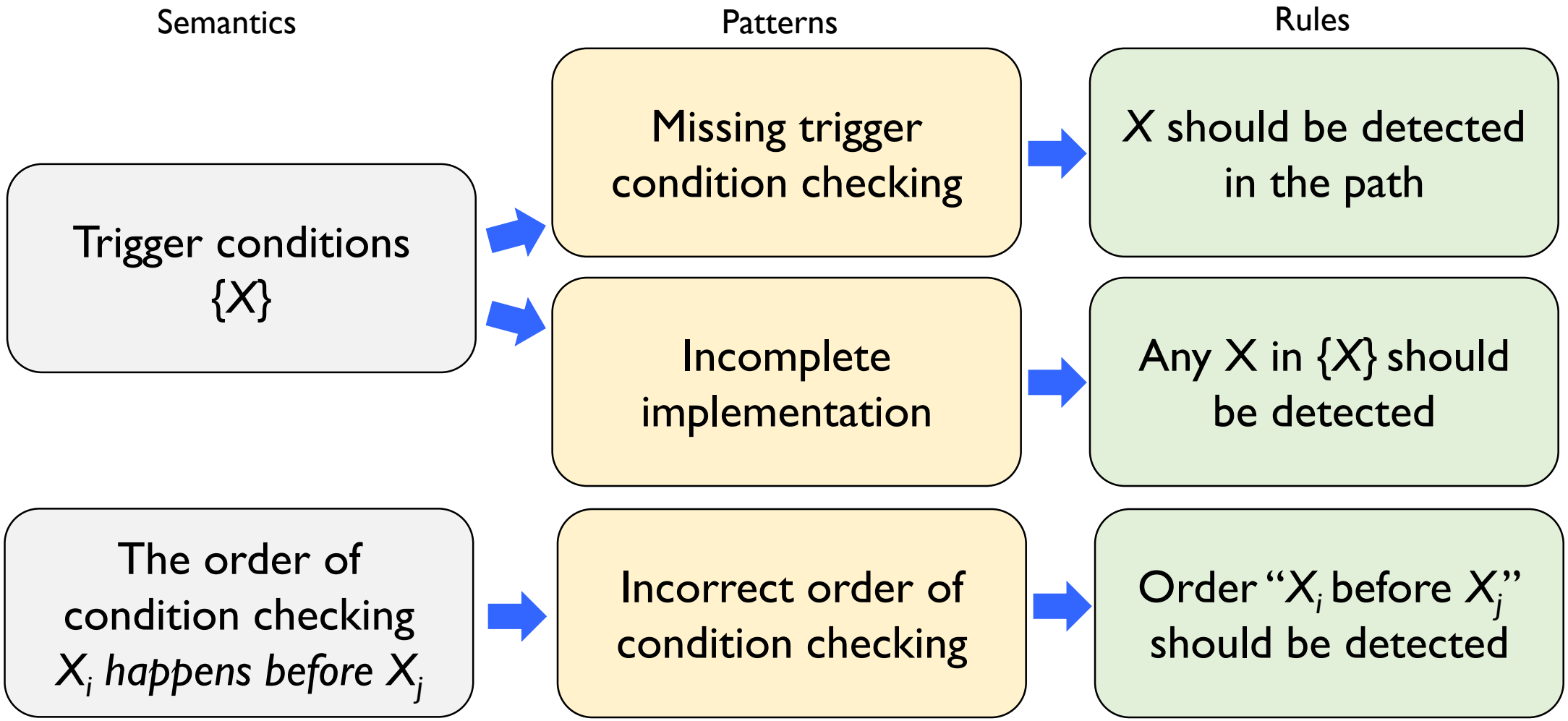
```
+  if (order == 0) {
+      …
+      page  = list_last_entry(list, struct page, lru);
+      …
+  } else {
```



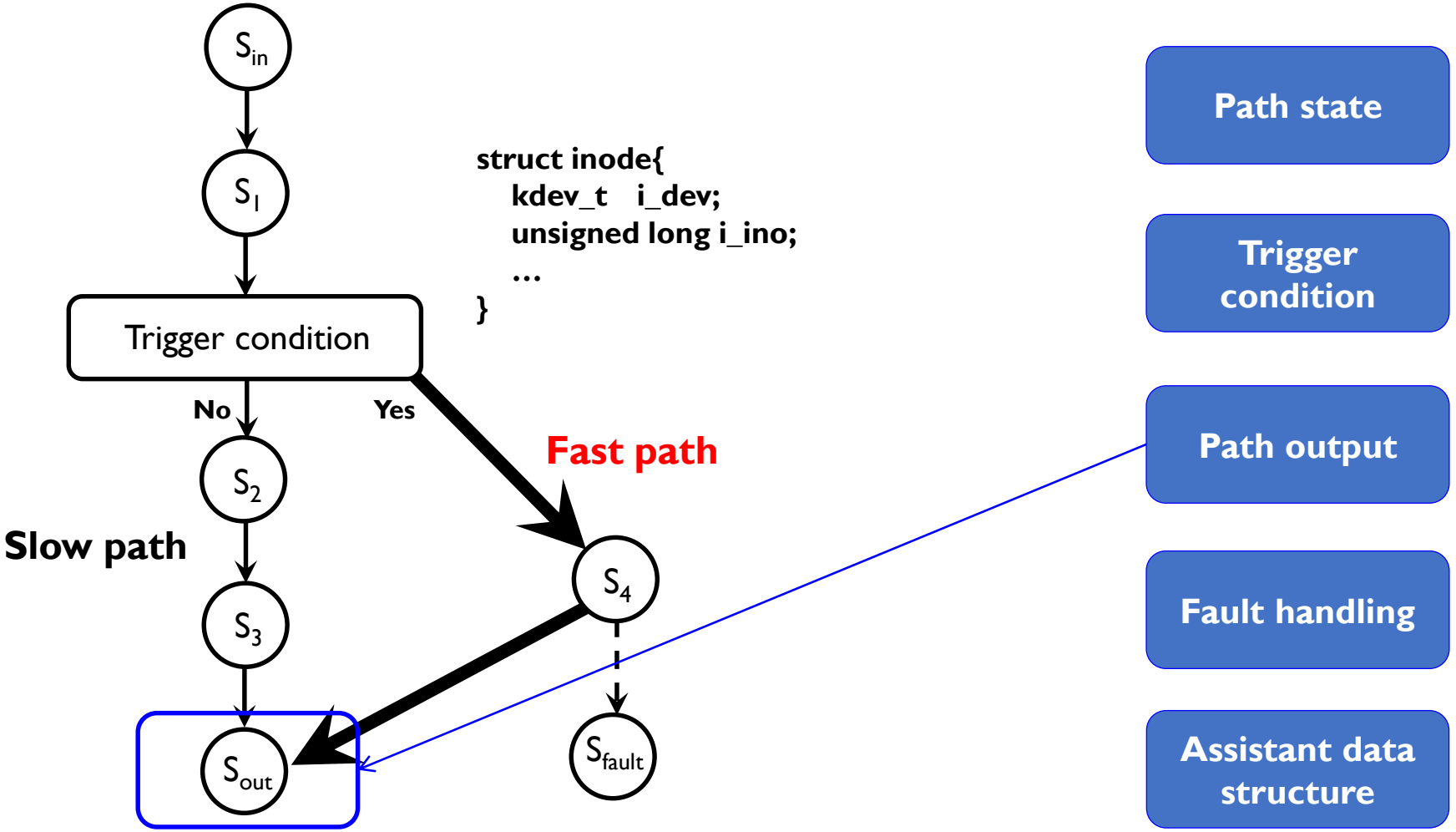
How does Trigger Condition Cause Bugs?



How does Trigger Condition Cause Bugs?



Fast-Path Bug Categorization

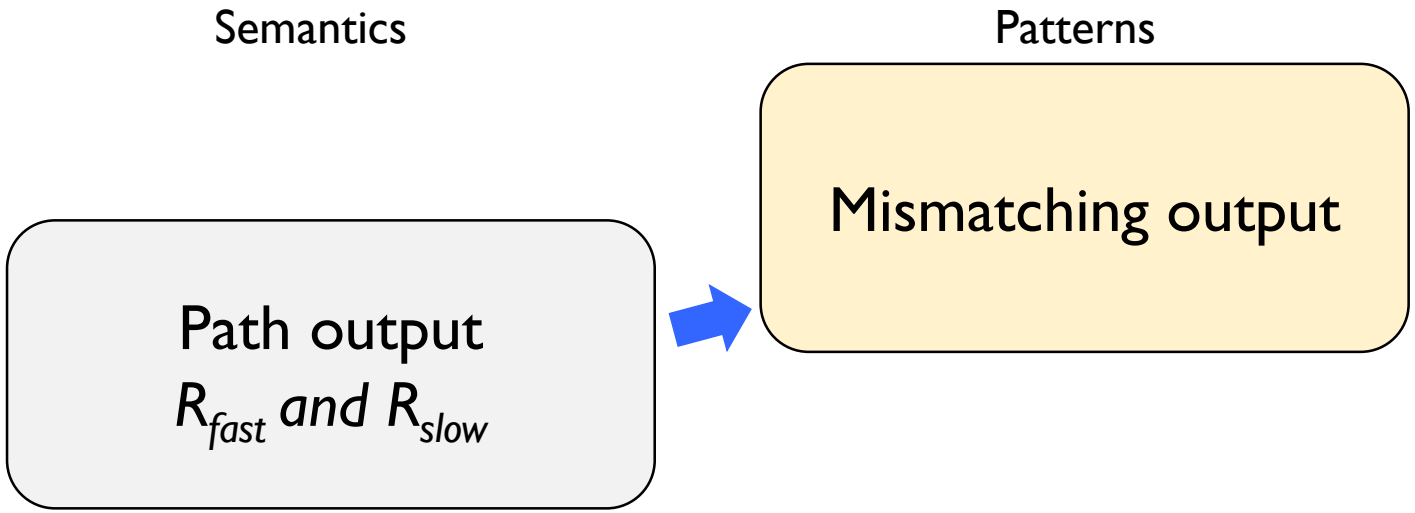


How does Path Output Cause Bugs?

Semantics

Path output
R_{fast} *and* R_{slow}

How does Path Output Cause Bugs?



How does Path Output Cause Bugs?

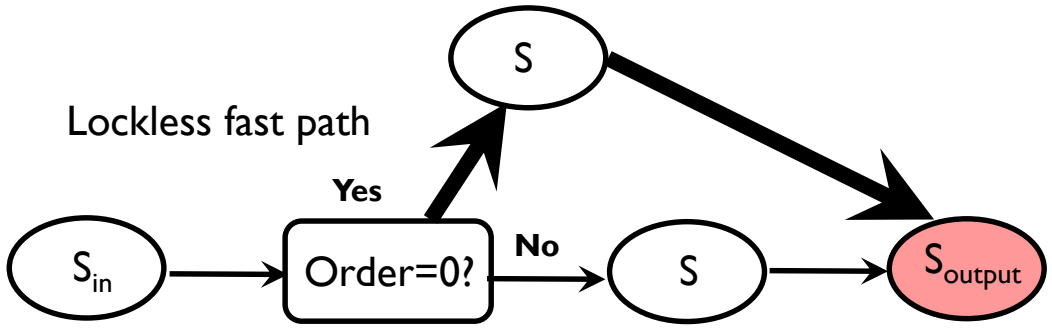
Semantics

Patterns

Path output
R_{fast} *and* R_{slow}

Mismatching output

```
+  if (order == 0) {
+      …
+      SetPageDirty(page)
+      …
+  } else {
       //Set page in dirty state
       SetPageDirty(page);
```



How does Path Output Cause Bugs?

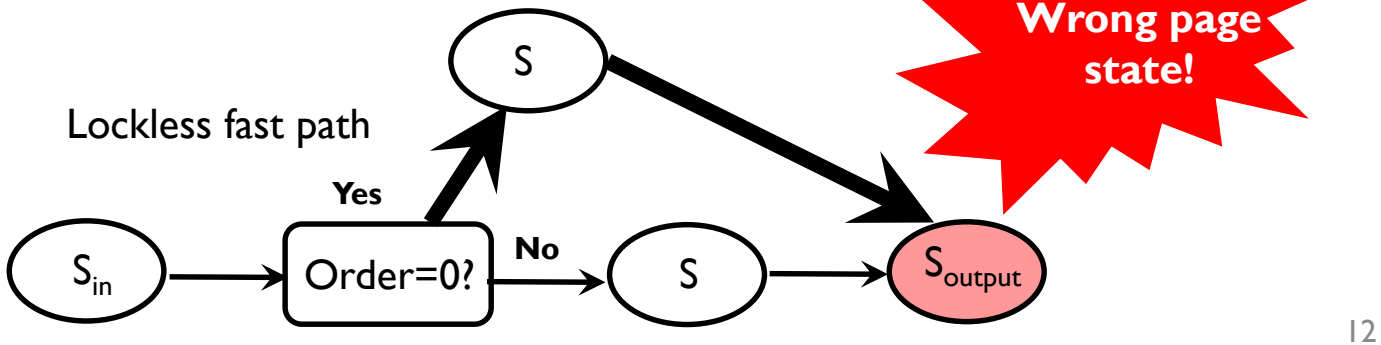
Semantics

Patterns

Path output
R_{fast} *and* R_{slow}

Mismatching output

```
+  if (order == 0) {
+      …
+      SetPageDirty(page)
+      …
+  } else {
       //Set page in dirty state
       SetPageDirty(page);
```



How does Path Output Cause Bugs?

Semantics

Path output
R_{fast} *and* R_{slow}

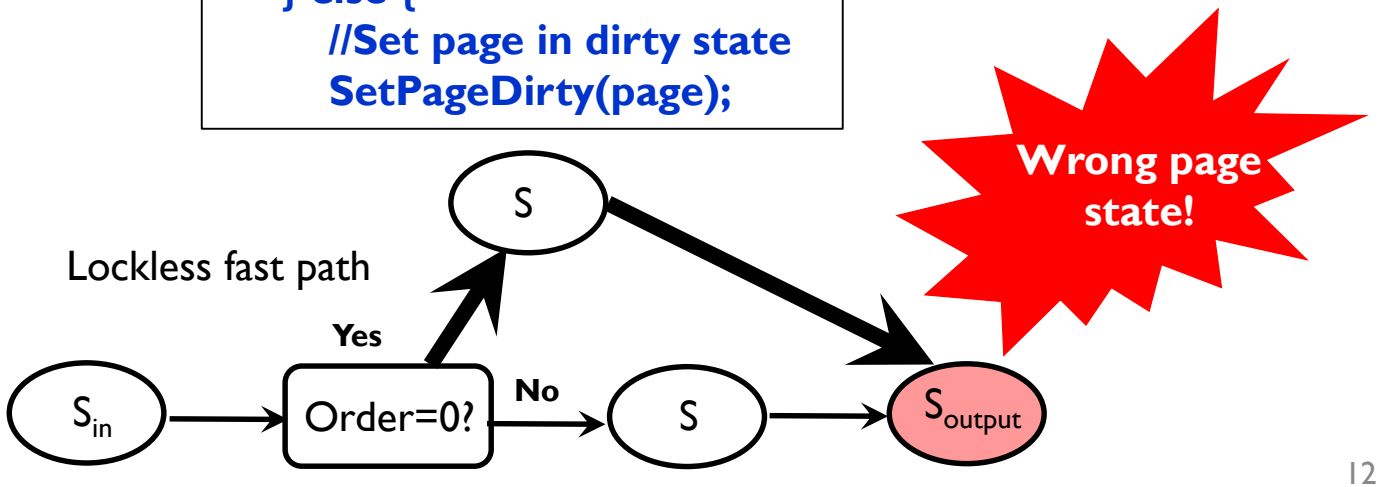
Patterns

Mismatching output

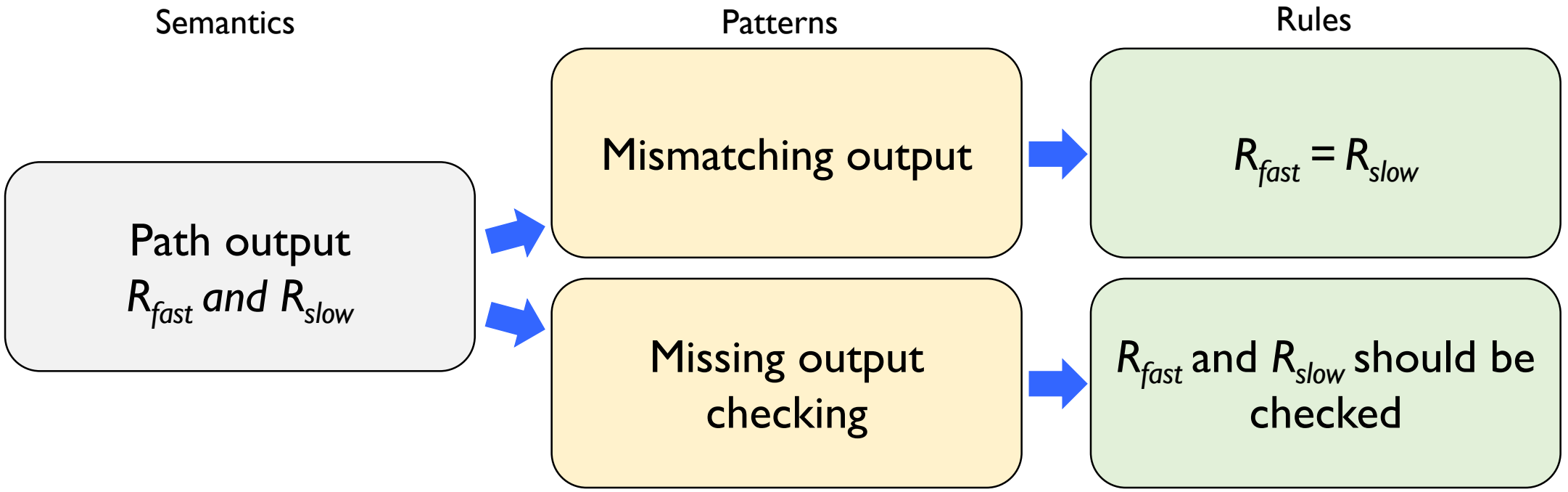
Rules

$R_{fast} = R_{slow}$

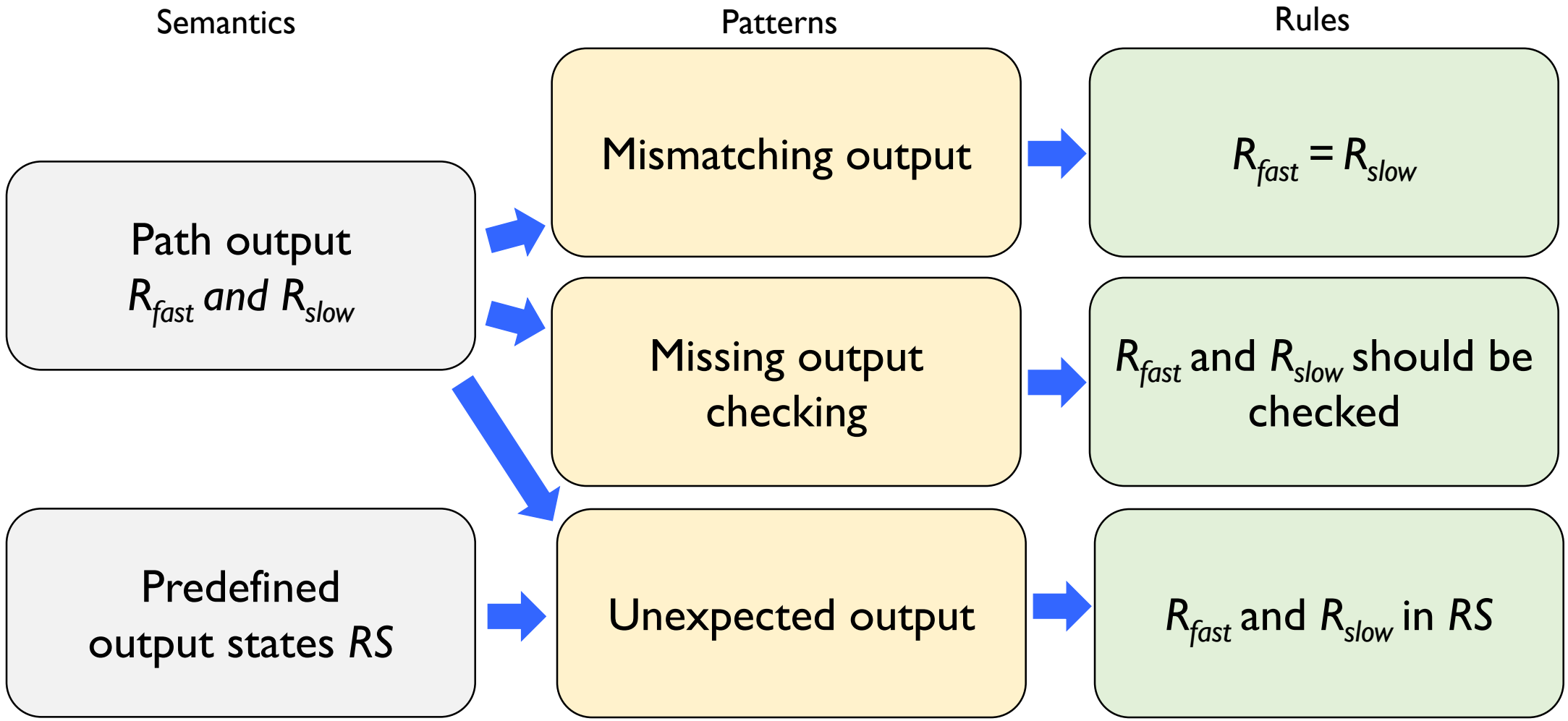
```
+  if (order == 0) {
+      …
+      SetPageDirty(page)
+      …
+  } else {
       //Set page in dirty state
       SetPageDirty(page);
```



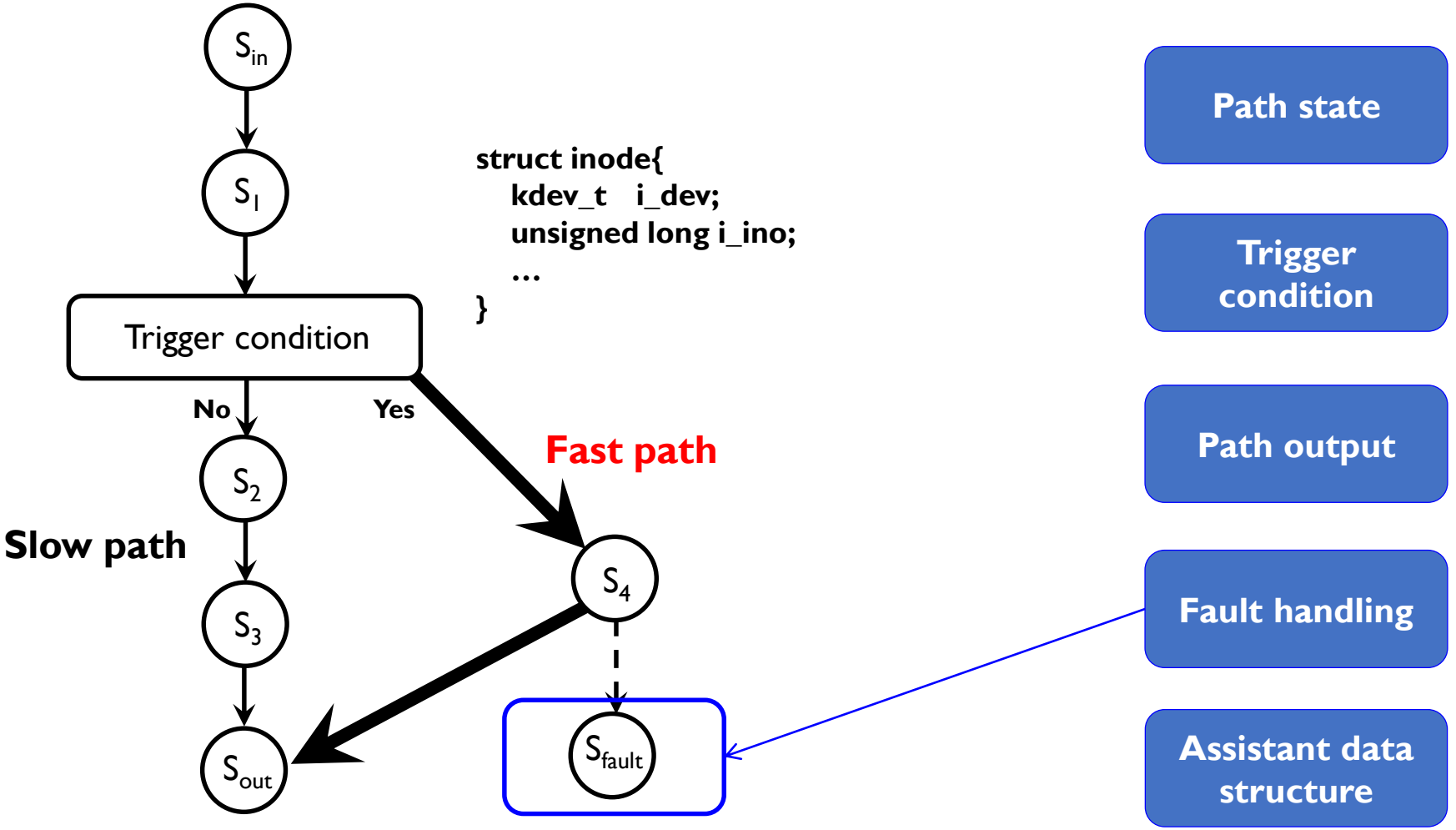
How does Path Output Cause Bugs?



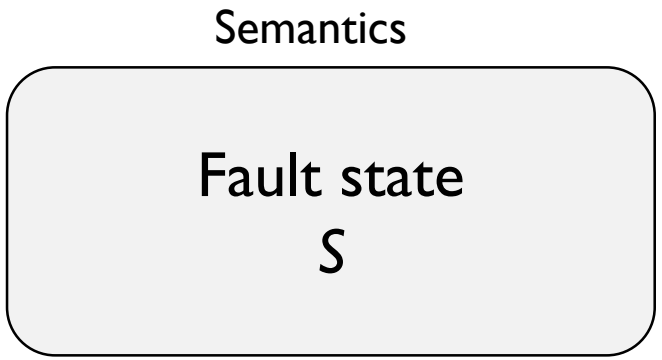
How does Path Output Cause Bugs?



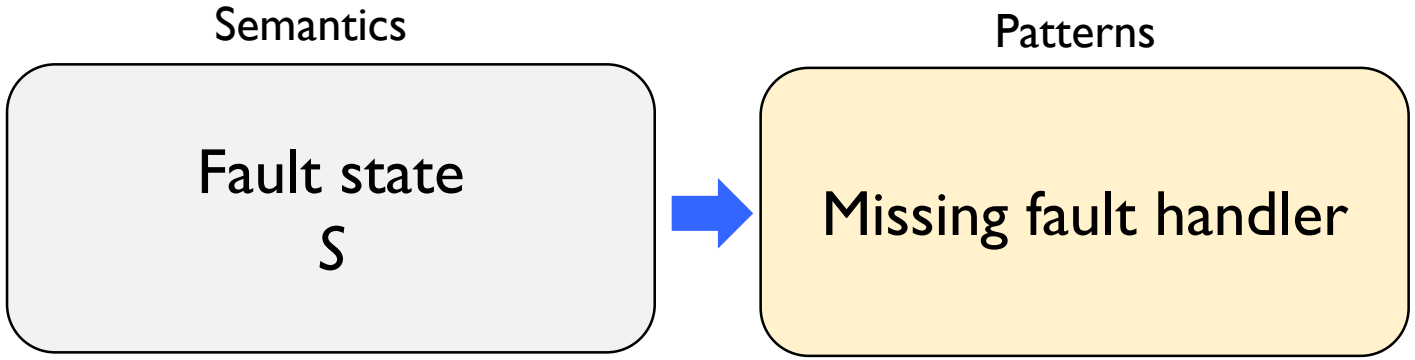
Fast-Path Bug Categorization



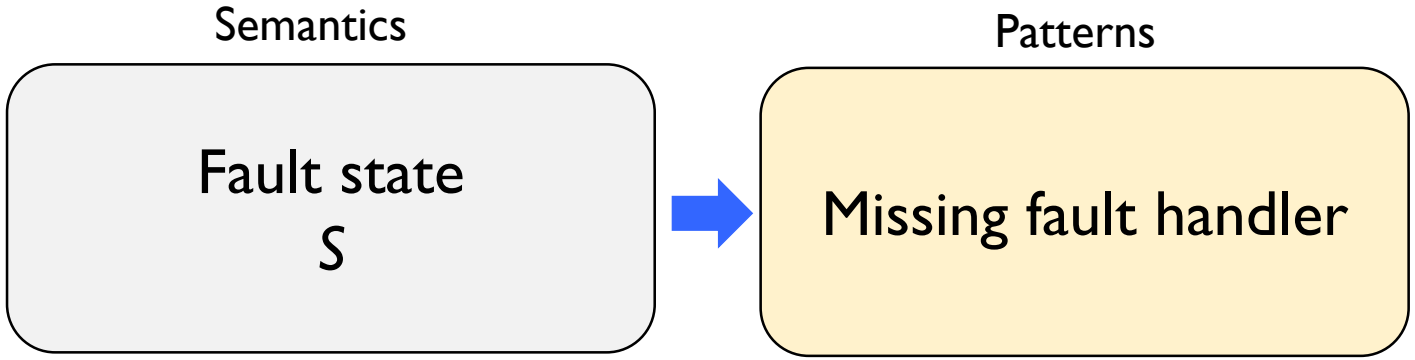
How does Fault Handler Cause Bugs?



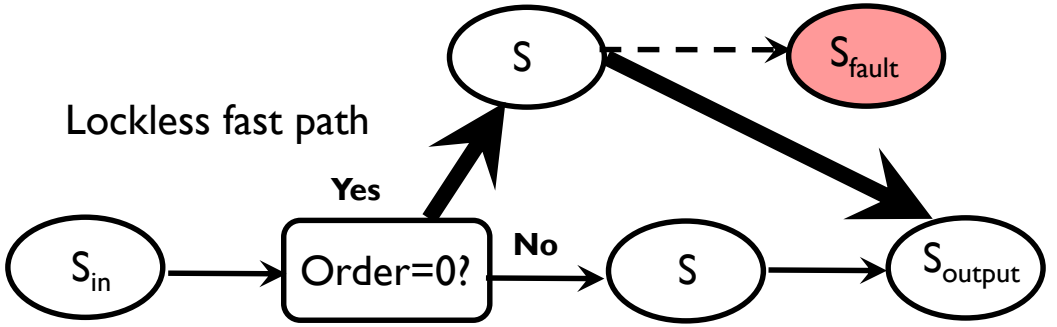
How does Fault Handler Cause Bugs?



How does Fault Handler Cause Bugs?




```
+   if (order == 0) {
+       page  = list_last_entry(list, struct page, lru);
+       if(!page) {
+           //Fault handler
+       }
+   } else {
```



How does Fault Handler Cause Bugs?

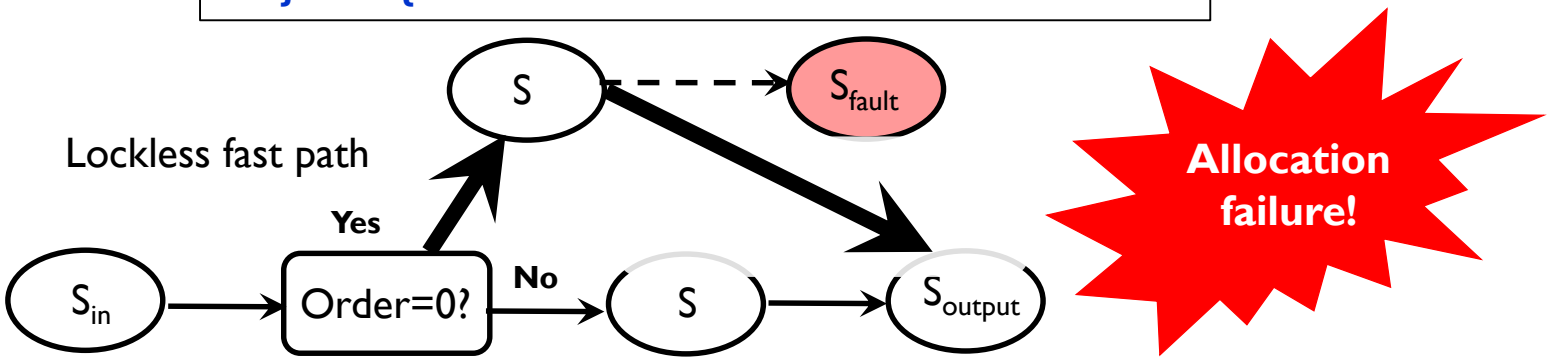
Semantics

Fault state
S

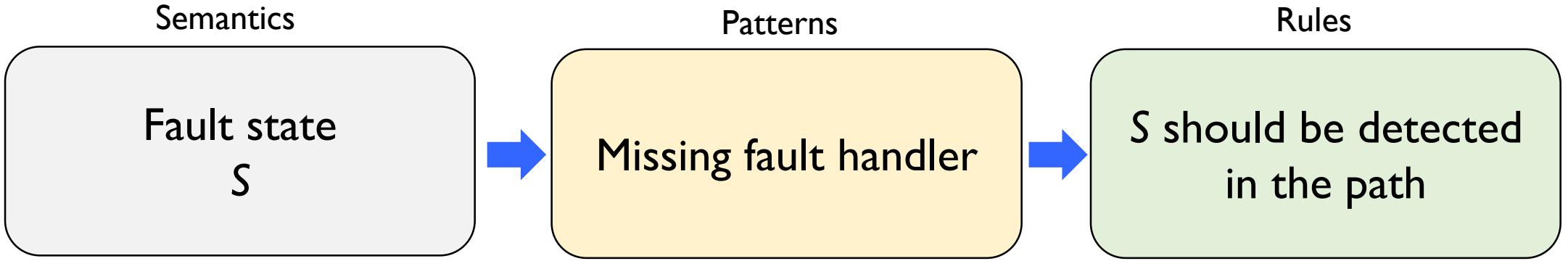
Patterns

Missing fault handler

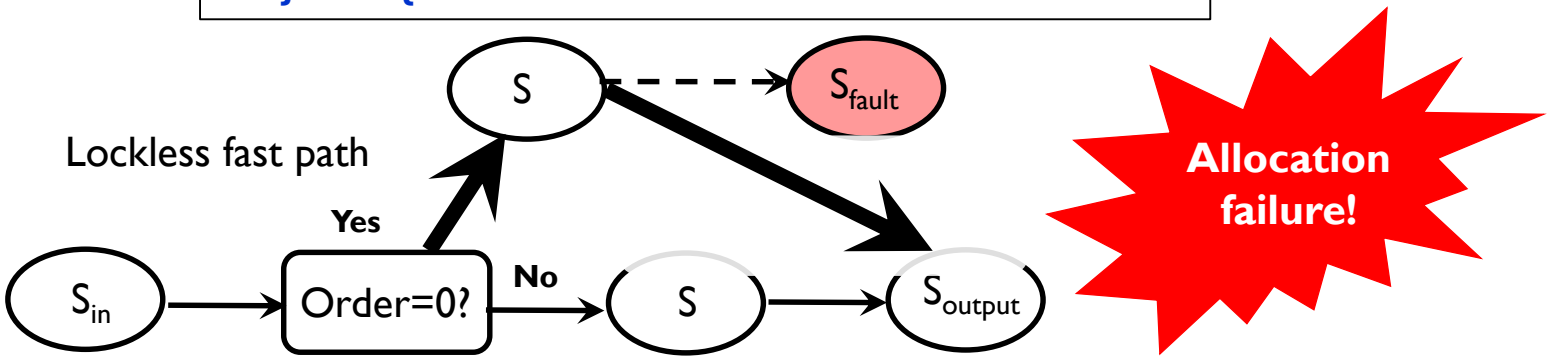
```
+   if (order == 0) {
+       page  = list_last_entry(list, struct page, lru);
+       if(!page) {
+           //Fault handler
+       }
+   } else {
```



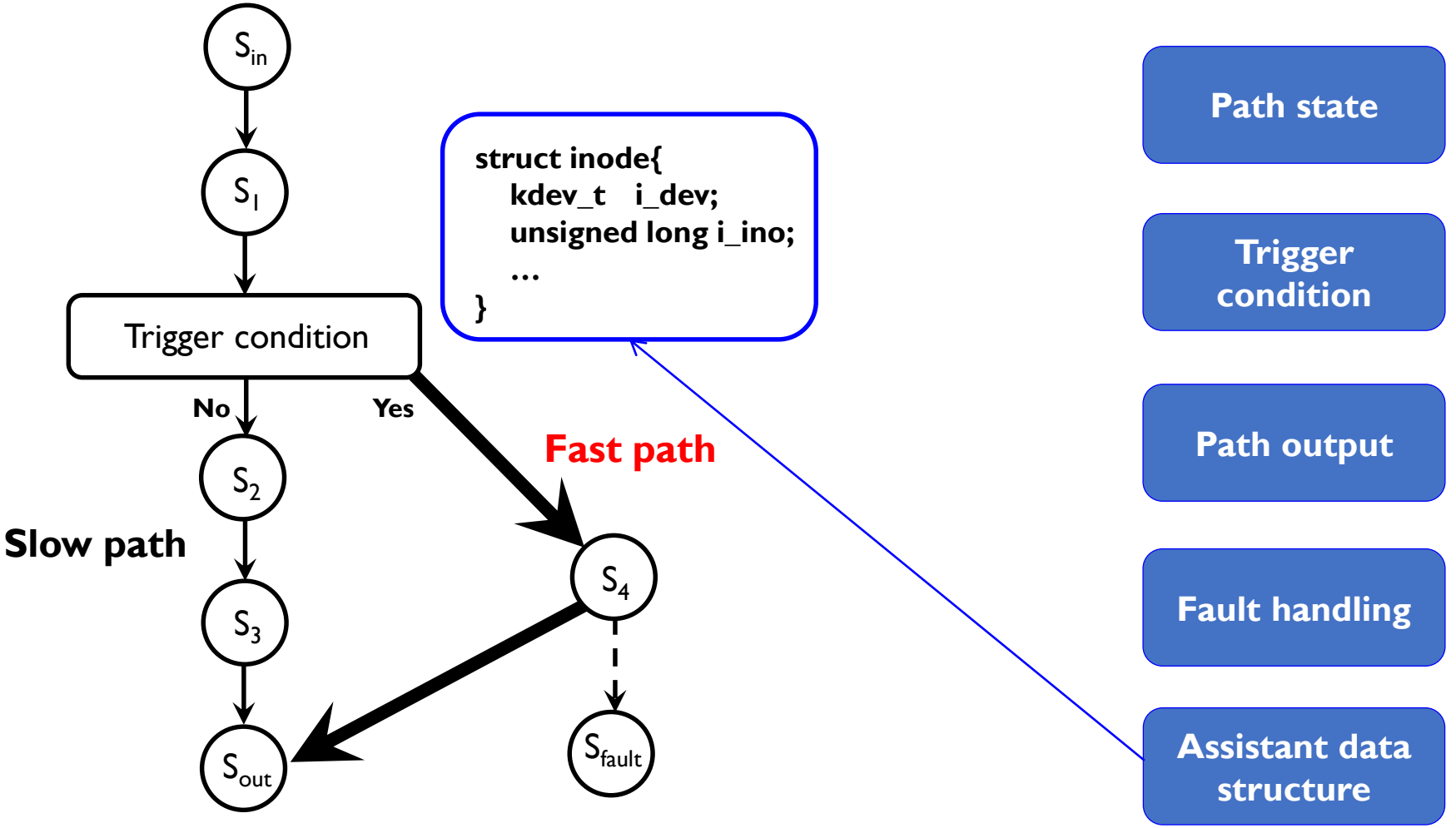
How does Fault Handler Cause Bugs?




```
+   if (order == 0) {
+       page  = list_last_entry(list, struct page, lru);
+       if(!page) {
+           //Fault handler
+       }
+   } else {
```



Fast-Path Bug Categorization

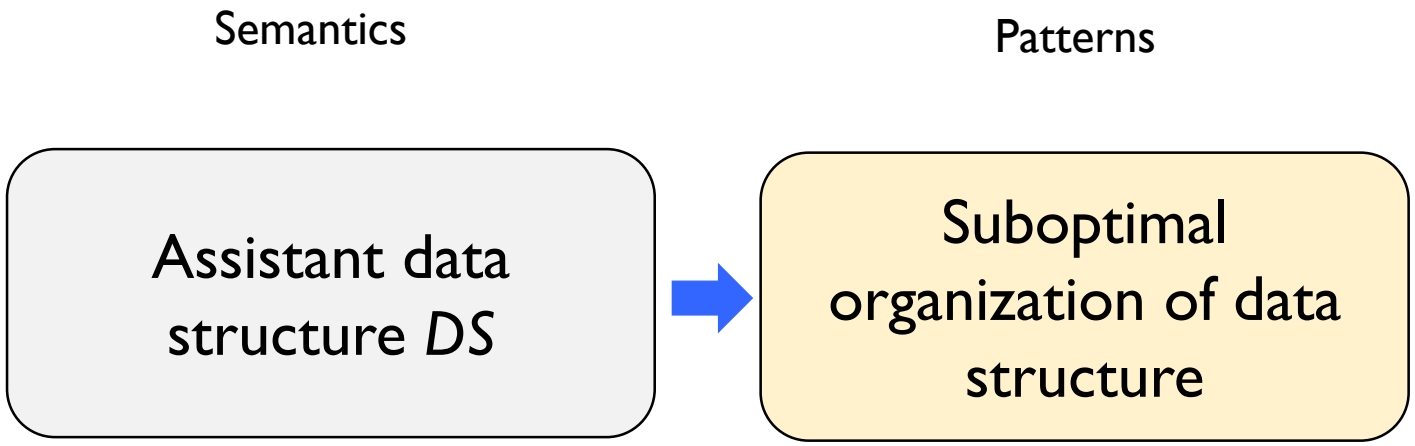


How does Assistant Data Structure Cause Bugs?

Semantics

Assistant data structure *DS*

How does Assistant Data Structure Cause Bugs?



How does Assistant Data Structure Cause Bugs?

Semantics

Patterns

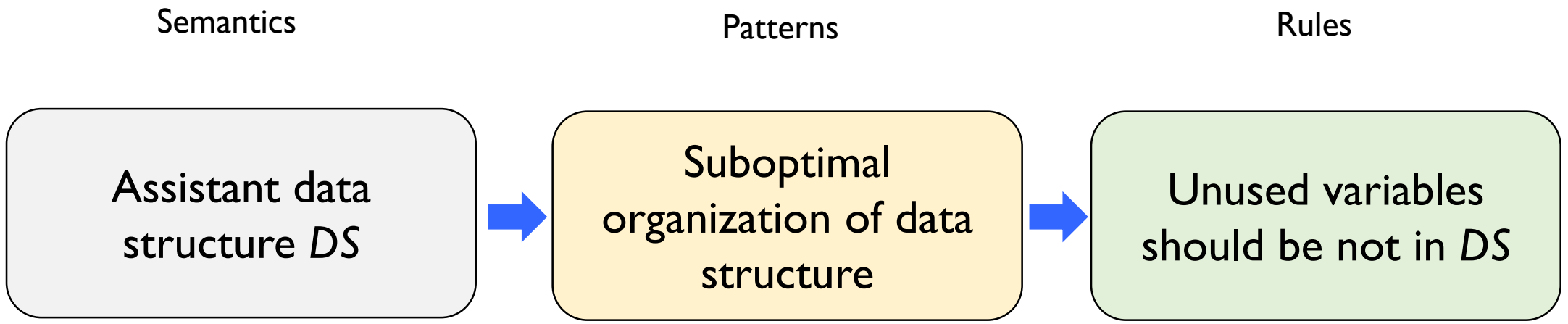
Assistant data structure *DS*

Suboptimal organization of data structure

```
struct inode{
    kdev_t          i_dev;
    unsigned long   i_ino;
    int             i_cindex;
    …
}
```


Never used in fast path

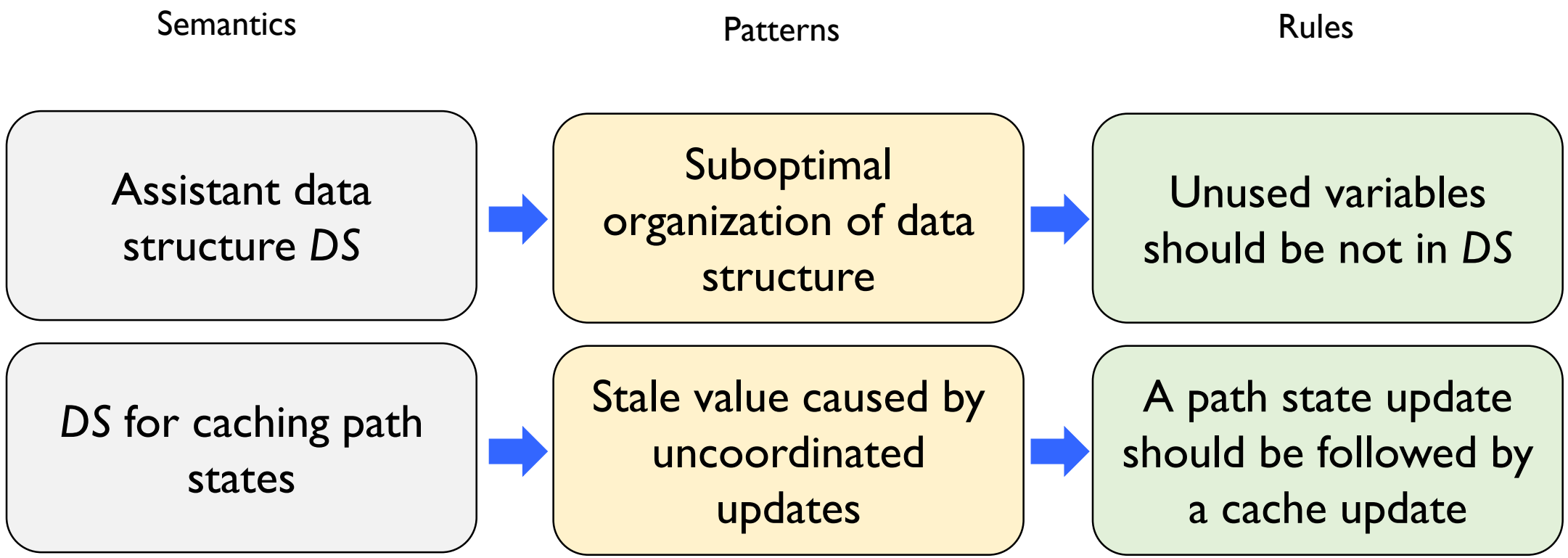
How does Assistant Data Structure Cause Bugs?



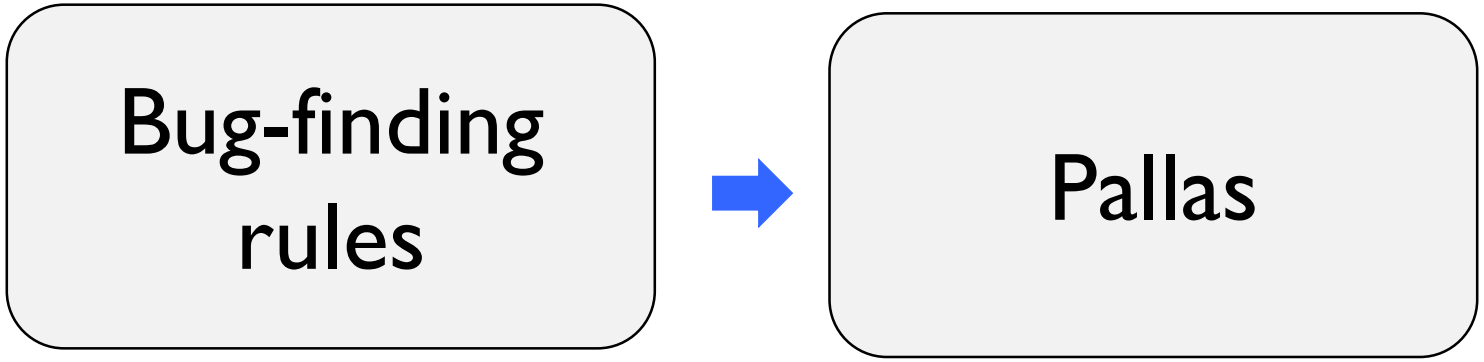

```
struct inode{
    kdev_t          i_dev;
    unsigned long   i_ino;
    int             i_cindex;
    …
}
```

Never used in fast path

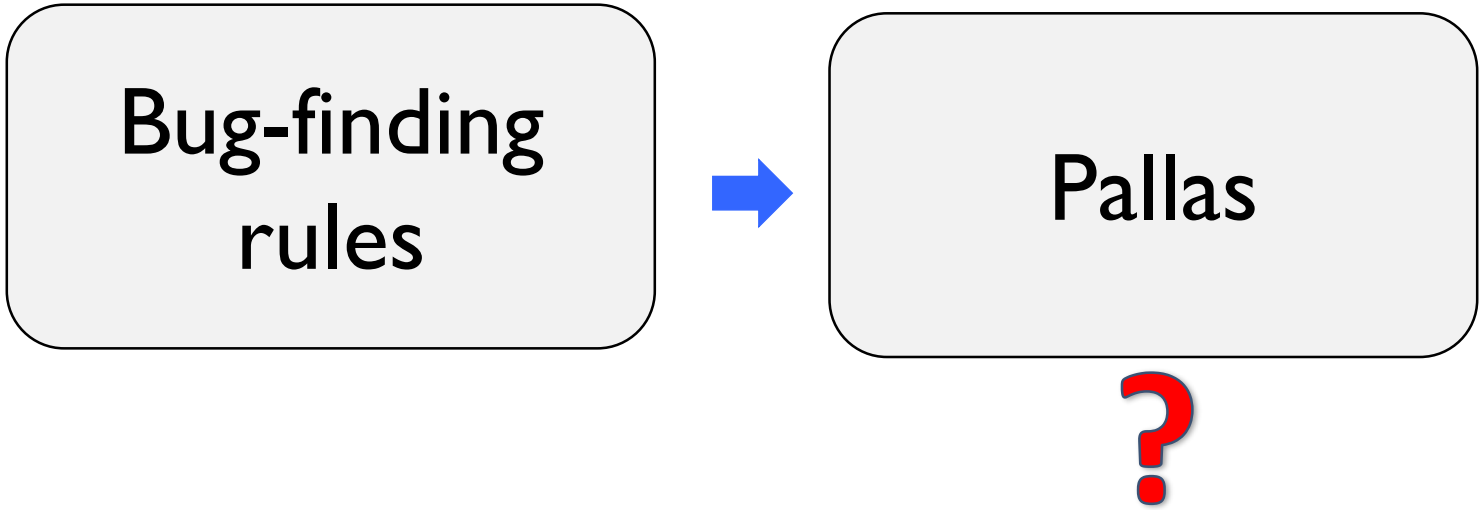
How does Assistant Data Structure Cause Bugs?



Pallas: A Semantic-Aware Static Checking Tool

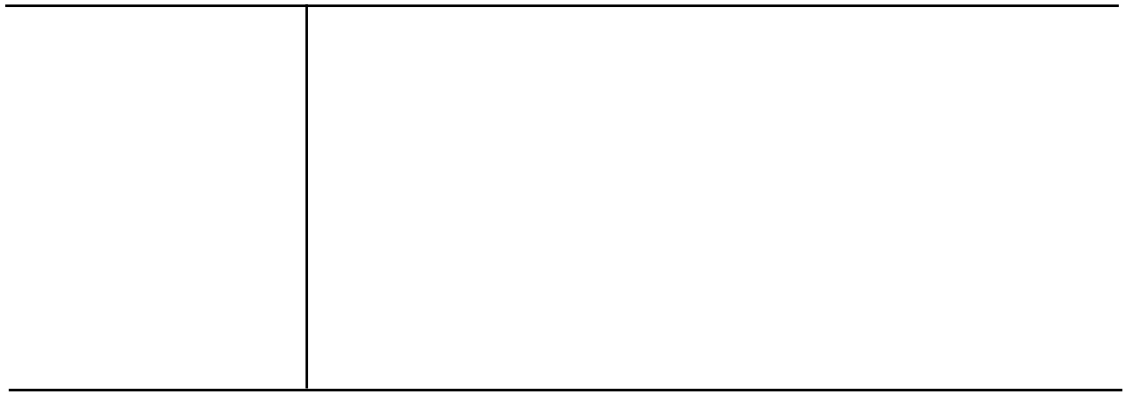


Pallas: A Semantic-Aware Static Checking Tool



How to Generate Path Information?

```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
     return SUCCESS;
}
```

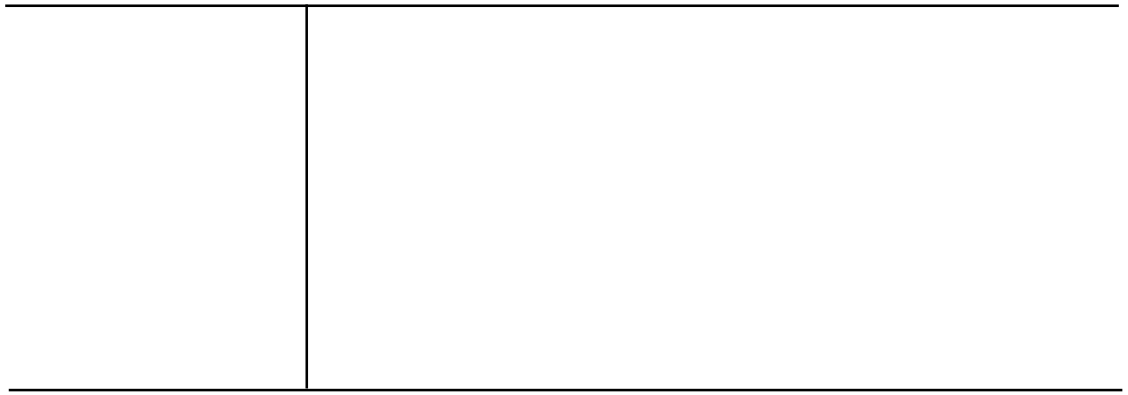


A fast-path patch

Path information

How to Generate Path Information?

```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
     return SUCCESS;
}
```



A fast-path patch

Path information

How to Generate Path Information?

```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
     return SUCCESS;
}
```

Signature	foo_get_page(gfp_mask)

A fast-path patch

Path information

How to Generate Path Information?

```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
     return SUCCESS;
}
```

Signature	foo_get_page(gfp_mask)

A fast-path patch

Path information

How to Generate Path Information?

```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
     return SUCCESS;
}
```

A fast-path patch

Signature	foo_get_page(gfp_mask)
Condition	order == 0

Path information

How to Generate Path Information?

```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
     return SUCCESS;
}
```

A fast-path patch

Signature	foo_get_page(gfp_mask)
Condition	order == 0

Path information

How to Generate Path Information?

```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
     return SUCCESS;
}
```

A fast-path patch

Signature	foo_get_page(gfp_mask)
Condition	order == 0
State	gfp_mask=noio_flag(gfp_mask)

Path information

How to Generate Path Information?

```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
    return SUCCESS;
}
```

A fast-path patch

Signature	foo_get_page(gfp_mask)
Condition	order == 0
State	gfp_mask=noio_flag(gfp_mask)

Path information

How to Generate Path Information?

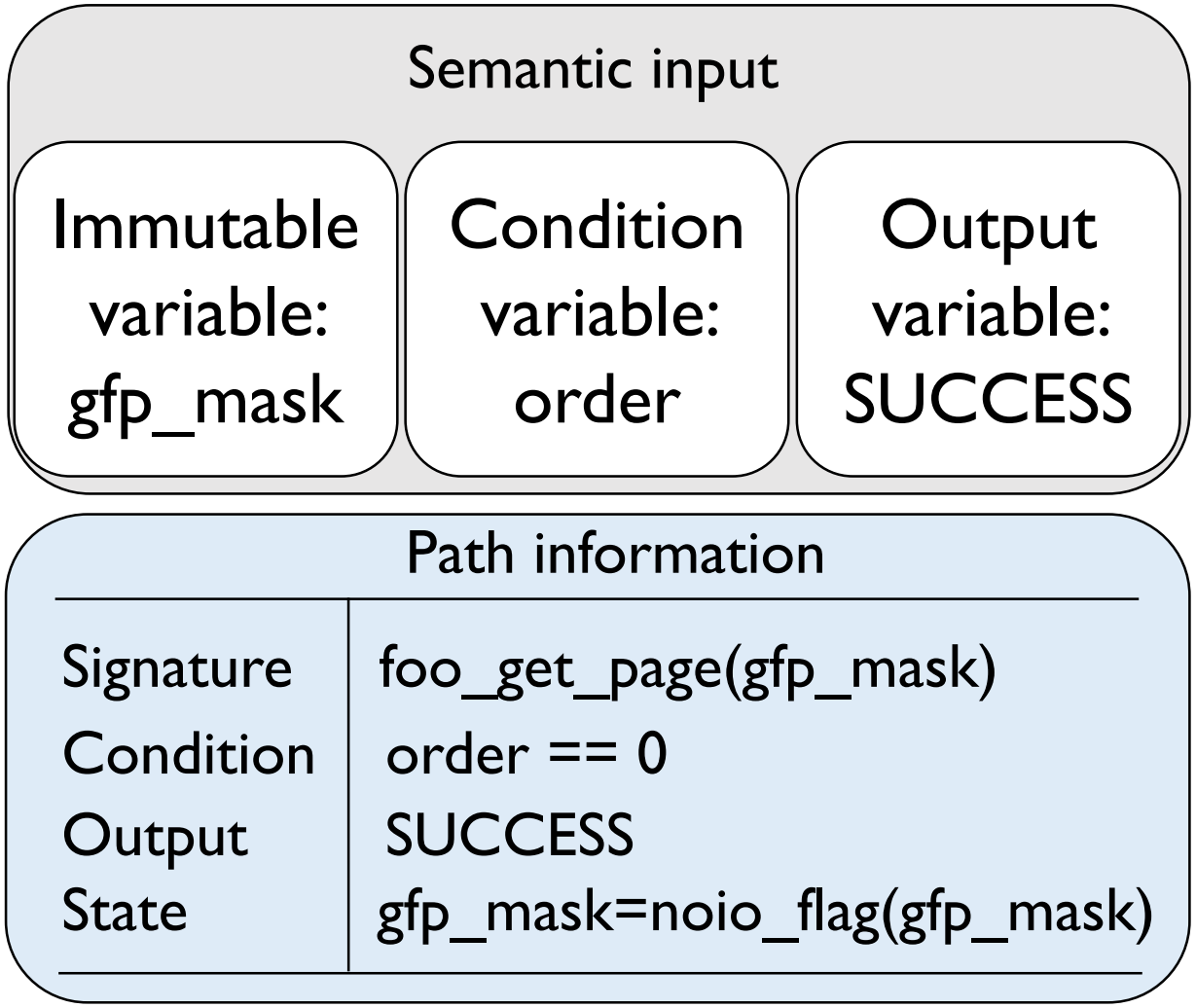
```
int * foo_get_page(gfp_mask)
{
+   //lockless fast path
+   if (order == 0) {
+       gfp_mask = noio_flag(gfp_mask);
+       }
+    }
    return SUCCESS;
}
```

A fast-path patch

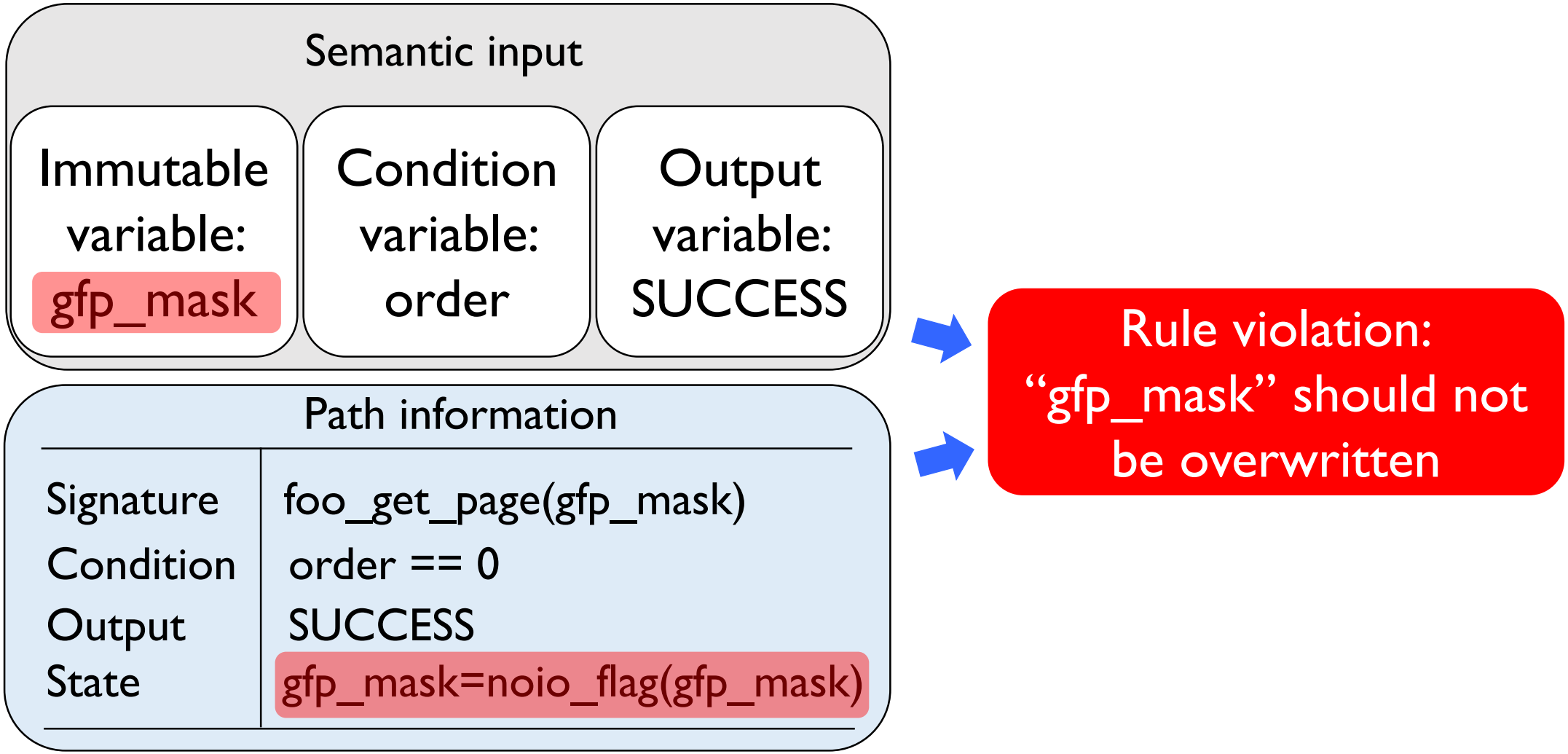
Signature	foo_get_page(gfp_mask)
Condition	order == 0
State	gfp_mask=noio_flag(gfp_mask)
Output	SUCCESS

Path information

How does Pallas Check Paths?



How does Pallas Check Paths?



How does Pallas Check Paths?

Semantic input

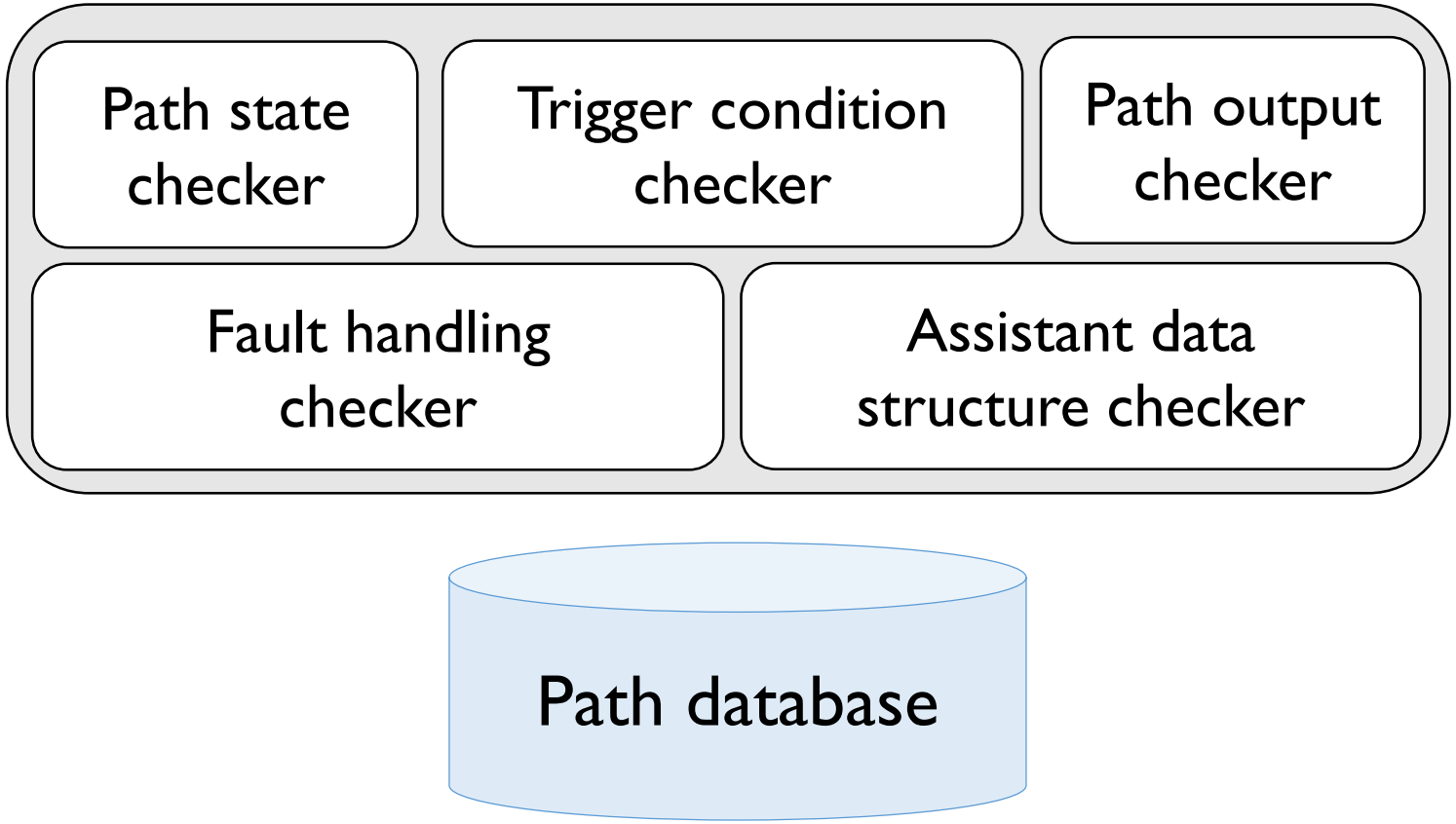
Immutable variable:	Condition variable:	Output variable:
gfp_mask	order	SUCCESS

Path information

Signature	foo_get_page(gfp_mask)
Condition	order == 0
Output	SUCCESS
State	gfp_mask=noio_flag(gfp_mask)

Rule violation: "gfp_mask" should not be overwritten

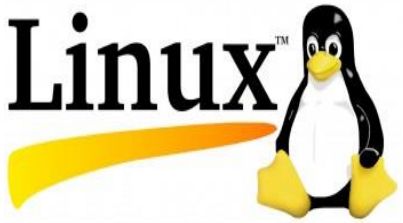
Pallas Implementation



Evaluation with Real-World Systems

Chromium

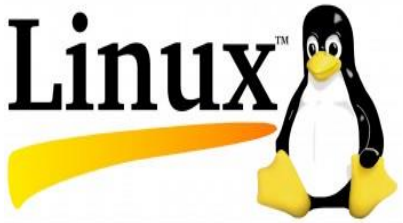
Evaluation with Real-World Systems



Chromium

Checkers	Bugs	False positive
Path state	29	18
Trigger condition	41	13
Path output	35	16
Fault handling	27	10
Assistant DS	23	12
Total	155	69

Evaluation with Real-World Systems



Chromium

Checkers	Bugs	False positive
Path state	29	18
Trigger condition	41	13
Path output	35	16
Fault handling	27	10
Assistant DS	23	12
Total	155	69

The bugs already exist for 3.1 years on average

An Example of New Bugs in Chromium

```
Void PPBNaClPrivate::DownloadNexe(…)
{
   // Try the fast path for retrieving the file
   PP_FileHandle handle = OpenNaClExecutable(instance, url, &out_file_info)
   if(handle != PP_kInvalidFileHandle) {
      DownloadNexeCompletion(request, out_file_info, FileDownloader::SUCCESS);
      return;
   }

   // The fast path didn't work, try slow path
   …
}
```

An Example of New Bugs in Chromium

```
Void PPBNaClPrivate::DownloadNexe(…)
{
   // Try the fast path for retrieving the file
   PP_FileHandle handle = OpenNaClExecutable(instance, url, &out_file_info)
   if(handle != PP_kInvalidFileHandle) {
      DownloadNexeCompletion(request, out_file_info, FileDownloader::SUCCESS);
      return;
   }

   // The fast path didn't work, try slow path
   …
}
```

Conclusions

- Fast path introduces semantic bugs
- General bug patterns exist in fast-path bugs
- Pallas: a semantic-aware static checking tool

*We will release bug database soon

Thanks!

Jian Huang†
jian.huang@gatech.edu

Michael Allen-Bond

Xuechen Zhang

† Georgia Tech

Washington State University

Q&A