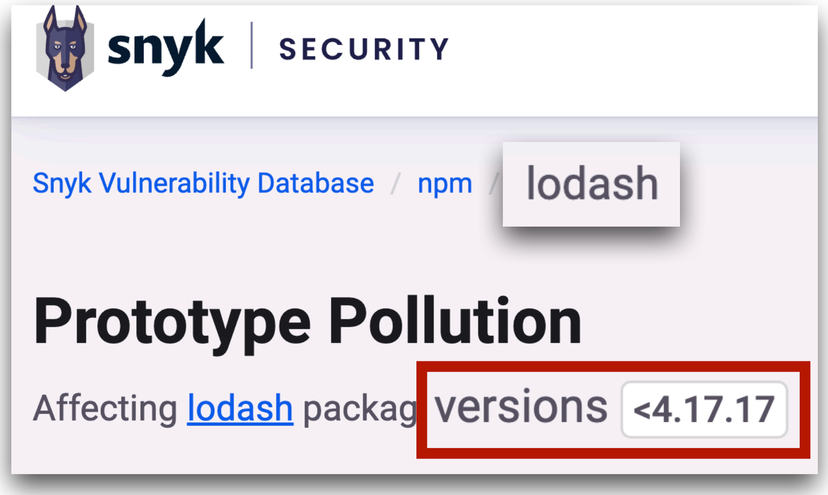
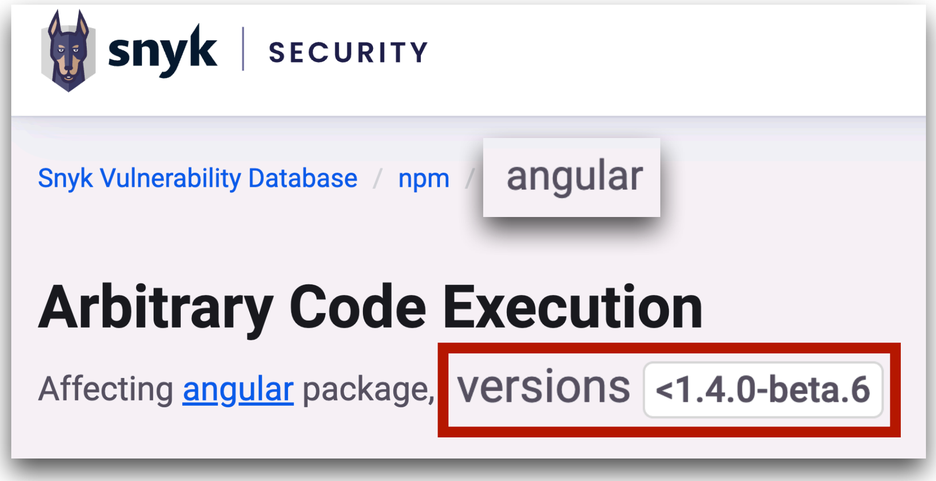
Debun: Detecting Bundled JavaScript Libraries on Web using Property-Order Graphs

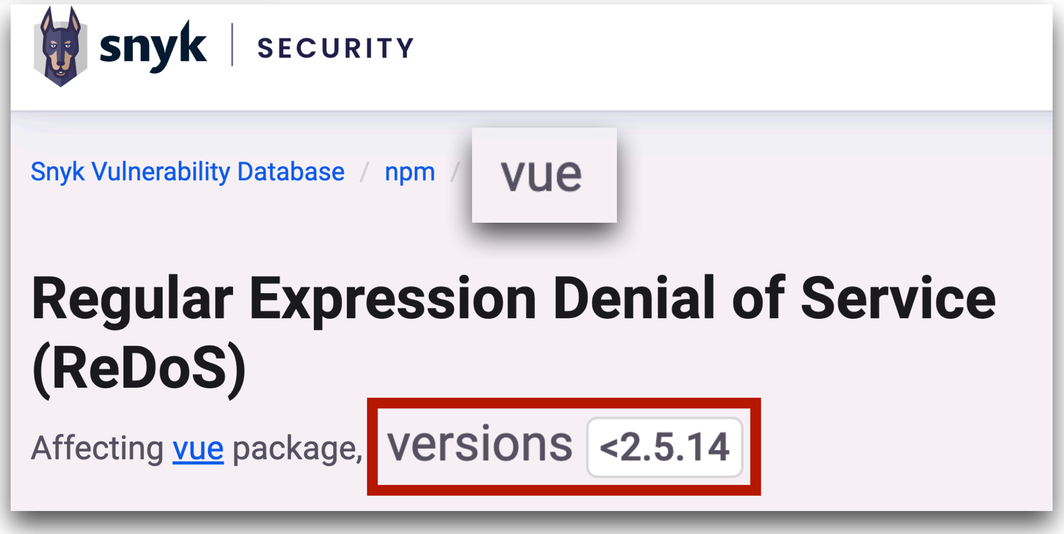
Seojin Kim*, **Sungmin Park***, and Jihyeok Park

KOREA UNIVERSITY

Vulnerabilities in JavaScript Library







PLRG

Deployment Process of Web Applications

- Bundlers ***modify*** and ***compress*** **user code** and **libraries** together using diverse **transpilers**
- It makes **difficult to detect libraries** in transpiled code in web applications

PLRG

Prior Work

- **LDC** (Library Detector for Chrome)
  - **Manually collected property patterns** at **runtime**
  - Library Detection

```
typeof (_ = window._) == ‘function'
typeof (chain = _ && _.chain) == ‘function’
```

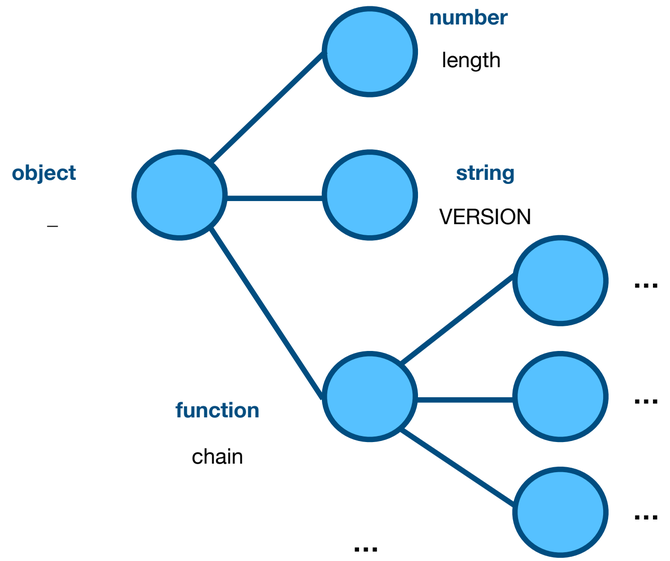
  - Version Detection

```
return { version: _.VERSION || UNKNOWN_VERSION };
```

Lodash v4.17.21

```
6   // Define properties
7   lodash.chain = function(value) {
8     var result = lodash(value);
9     result.__wrapped__ = value;
10    return result;
11  }
12  lodash.differenceBy = ...
13  ...
14  lodash.VERSION = '4.17.21'
15  // Export lodash
16  window._ = lodash;
17 }.call(this));
```

- **PTDetector** (ASE’23)
  - **Automatic extraction** of **property patterns** in tree form



- Prior work does **NOT utilize bundled code** to detect libraries

PLRG

Prior Work

- **LDC** (Library Detector for Chrome)
  - **Manually collected property patterns** at **runtime**
  - Library Detection

```
typeof (_ = window._) == 'function'
typeof (chain = _ && _.chain) == 'function'
```

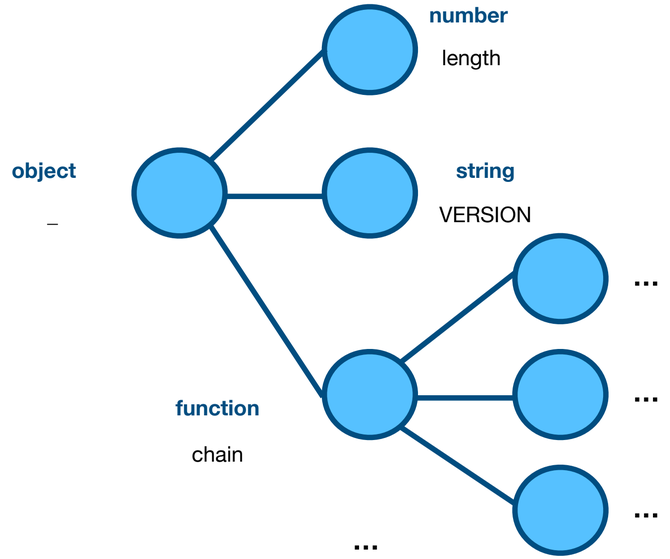
  - Version Detection

```
return { version: _.VERSION || UNKNOWN_VERSION };
```

Lodash v4.17.21

```
6   // Define properties
7   lodash.chain = function(value) {
8     var result = lodash(value);
9     result.__wrapped__ = value;
10    return result;
11  }
12  lodash.differenceBy = ...
13  ...
14  lodash.VERSION = '4.17.21'
15  // Export lodash
16  window._ = lodash;
17 }.call(this));
```

- **PTDetector** (ASE'23)
  - **Automatic extraction** of **property patterns** in tree form



- Prior work does **NOT utilize bundled code** to detect libraries

PLRG

Prior Work

- **LDC** (Library Detector for Chrome)
  - **Manually collected property patterns** at **runtime**
  - Library Detection

```
typeof (_ = window._) == 'function'
typeof (chain = _ && _.chain) == 'function'
```

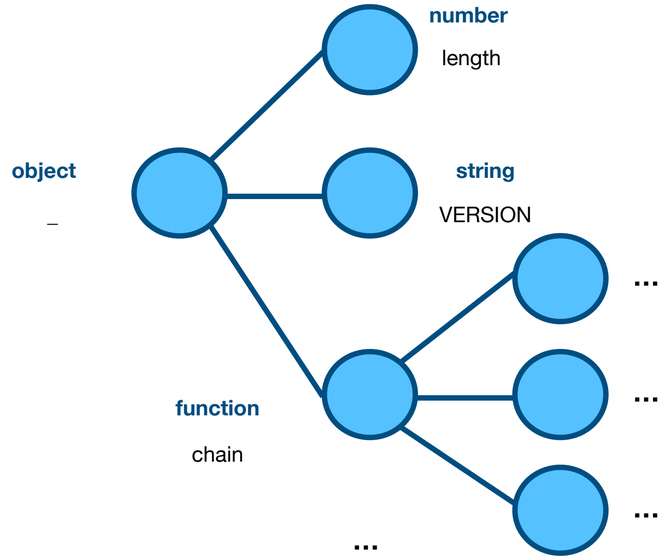
  - Version Detection

```
return { version: _.VERSION || UNKNOWN_VERSION };
```

Lodash v4.17.21

```
6   // Define properties
7   lodash.chain = function(value) {
8     var result = lodash(value);
9     result.__wrapped__ = value;
10    return result;
11  }
12  lodash.differenceBy = ...
13  ...
14  lodash.VERSION = '4.17.21'
15  // Export lodash
16  window._ = lodash;
17 }.call(this));
```

- **PTDetector** (ASE'23)
  - **Automatic extraction** of **property patterns** in tree form



- Prior work does **NOT utilize bundled code** to detect libraries

PLRG

Prior Work

- **LDC** (Library Detector for Chrome)
  - **Manually collected property patterns** at **runtime**
  - Library Detection

```
typeof (_ = window._) == 'function'
typeof (chain = _ && _.chain) == 'function'
```

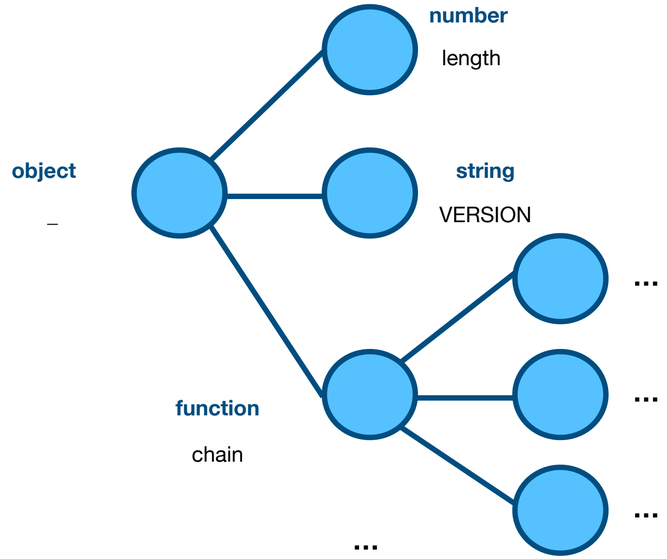
  - Version Detection

```
return { version: _.VERSION || UNKNOWN_VERSION };
```

Lodash v4.17.21

```
6   // Define properties
7   lodash.chain = function(value) {
8     var result = lodash(value);
9     result.__wrapped__ = value;
10    return result;
11  }
12  lodash.differenceBy = ...
13  ...
14  lodash.VERSION = '4.17.21'
15  // Export lodash
16  window._ = lodash;
17 }.call(this));
```

- **PTDetector** (ASE'23)
  - **Automatic extraction** of **property patterns** in tree form



- Prior work does **NOT utilize bundled code** to detect libraries

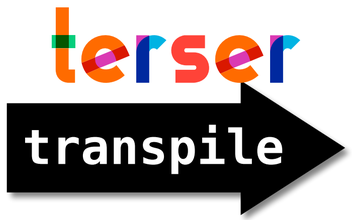
PLRG

Challenges of Library Detection in Transpiled Code

```
function remove(array, predicate) {
  var result = [];
  if (!(array && array.length)) {
    return result;
  }
  var index = -1,
    indexes = [],
    length = array.length;

  predicate = getIteratee(predicate, 3);
  while (++index < length) {
    var value = array[index];
    if (predicate(value, index, array)) {
      result.push(value);
      indexes.push(index);
    }
  }
  basePullAt(array, indexes);
  return result;
}
```

"remove" function in **Lodash.js v4.17.21**




```
function f(e,r){var t=[];if(!e||!e.length)

return t;var n=-1,u=[],a=e.length;

for(r=an(r,3);++n<a;){var h=e[n];

r(h,n,e)&&(t.push(h),u.push(n))}

return bf(e,u),t}
```

Transpiled Code

PLRG

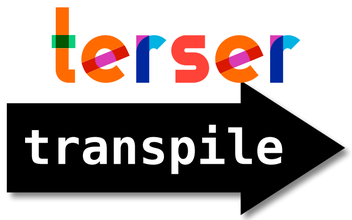
Challenges of Library Detection in Transpiled Code

```
function remove(array, predicate) {
  var result = [];
  if (!(array && array.length)) {
    return result;
  }
  var index = -1,
    indexes = [],
    length = array.length;

  predicate = getIteratee(predicate, 3);
  while (++index < length) {
    var value = array[index];
    if (predicate(value, index, array)) {
      result.push(value);
      indexes.push(index);
    }
  }
  basePullAt(array, indexes);
  return result;
}
```

"remove" function in **Lodash.js v4.17.21**

1. Mangled Names of Variables/Functions




```
function f(e,r){var t=[];if(!e||!e.length)
return t;var n=-1,u=[],a=e.length;
for(r=an(r,3);++n<a;){var h=e[n];
r(h,n,e)&&(t.push(h),u.push(n))}
return bf(e,u),t}
```

Transpiled Code

PLRG

Challenges of Library Detection in Transpiled Code

```
function remove(array, predicate) {
  var result = [];
  if (!(array && array.length)) {
    return result;
  }
  var index = -1,
    indexes = [],
    length = array.length;

  predicate = getIteratee(predicate, 3);
  while (++index < length) {
    var value = array[index];
    if (predicate(value, index, array)) {
      result.push(value);
      indexes.push(index);
    }
  }
  basePullAt(array, indexes);
  return result;
}
```

"remove" function in **Lodash.js v4.17.21**

1. **Mangled Names** of Variables/Functions
2. **Changed Control Statements**

terser

transpile

```
function f(e,r){var t=[];if(!e||!e.length)
return t;var n=-1,u=[],a=e.length;
for(r=an(r,3);++n<a;){var h=e[n];
r(h,n,e)&&(t.push(h),u.push(n))}
return bf(e,u),t}
```

Transpiled Code

PLRG

What is **Preserved** after **Transpilation**?

- **Property names** are *preserved* to support JavaScript's dynamic property access

```
array['len' + 'gth']  // array.length
```

- **Execution order** **between property reads/writes** is *preserved* to maintain side effects

```
obj = {
  get p() { console.log(1); }
  set q() { console.log(2); }
}
```

```
obj.p;        // print 1
obj.q = 42;   // print 2
```

≠

```
obj.q = 42;   // print 2
obj.p;        // print 1
```

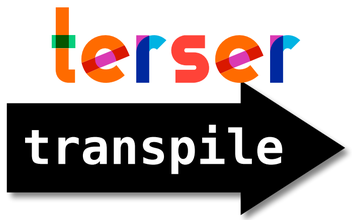
PLRG

What is **Preserved** after **Transpilation**?

```
function remove(array, predicate) {
  var result = [];
  if (!(array && array.length)) {
    return result;
  }
  var index = -1,
    indexes = [],
    length = array.length;

  predicate = getIteratee(predicate, 3);
  while (++index < length) {
    var value = array[index];
    if (predicate(value, index, array)) {
      result.push(value);
      indexes.push(index);
    }
  }
  basePullAt(array, indexes);
  return result;
}
```

"remove" function in **Lodash.js v4.17.21**




```
function f(e,r){var t=[];if(!e||!e.length)
return t;var n=-1,u=[],a=e.length;
for(r=an(r,3);++n<a;){var h=e[n];
r(h,n,e)&&(t.push(h),u.push(n))}
return bf(e,u),t}
```

Transpiled Code

PLRG

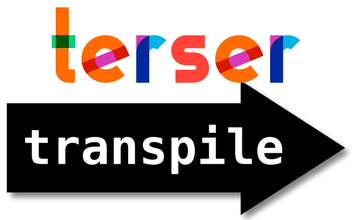
What is **Preserved** after **Transpilation**?

```
function remove(array, predicate) {
  var result = [];
  if (!(array && array.length)) {
    return result;
  }
  var index = -1,
    indexes = [],
    length = array.length;

  predicate = getIteratee(predicate, 3);
  while (++index < length) {
    var value = array[index];
    if (predicate(value, index, array)) {
      result.push(value);
      indexes.push(index);
    }
  }
  basePullAt(array, indexes);
  return result;
}
```

"remove" function in **Lodash.js v4.17.21**

1. Property Names




```
function f(e,r){var t=[];if(!e||!e.length)
return t;var n=-1,u=[],a=e.length;
for(r=an(r,3);++n<a;){var h=e[n];
r(h,n,e)&&(t.push(h),u.push(n))}
return bf(e,u),t}
```

Transpiled Code

PLRG

What is **Preserved** after **Transpilation**?

```
function remove(array, predicate) {
  var result = [];
  if (!(array && array.length)) {
    return result;
  }
  var index = -1,
    indexes = [],
    length = array.length;

  predicate = getIteratee(predicate, 3);
  while (++index < length) {
    var value = array[index];
    if (predicate(value, index, array)) {
      result.push(value);
      indexes.push(index);
    }
  }
  basePullAt(array, indexes);
  return result;
}
```

"remove" function in **Lodash.js v4.17.21**

1. **Property Names**
2. **Execution Order** between **Property Reads/Writes**

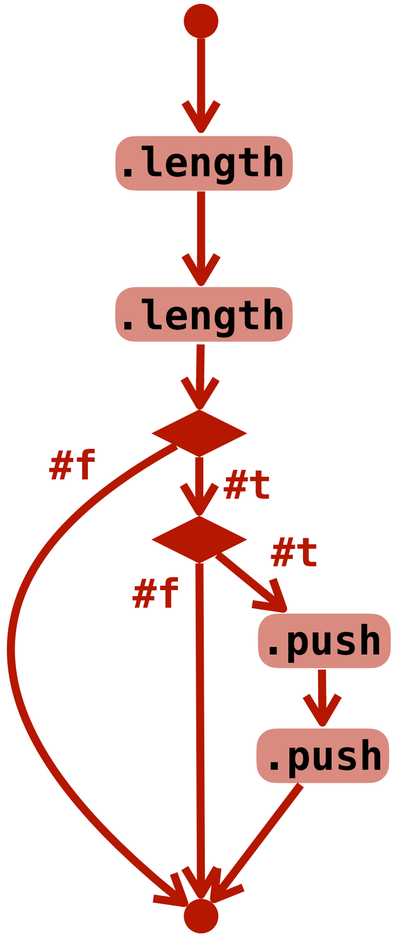
terser transpile

```
function f(e,r){var t=[];if(!e||!e.length)
return t;var n=-1,u=[],a=e.length;
for(r=an(r,3);++n<a;){var h=e[n];
r(h,n,e)&&(t.push(h),u.push(n))}
return bf(e,u),t}
```

Transpiled Code

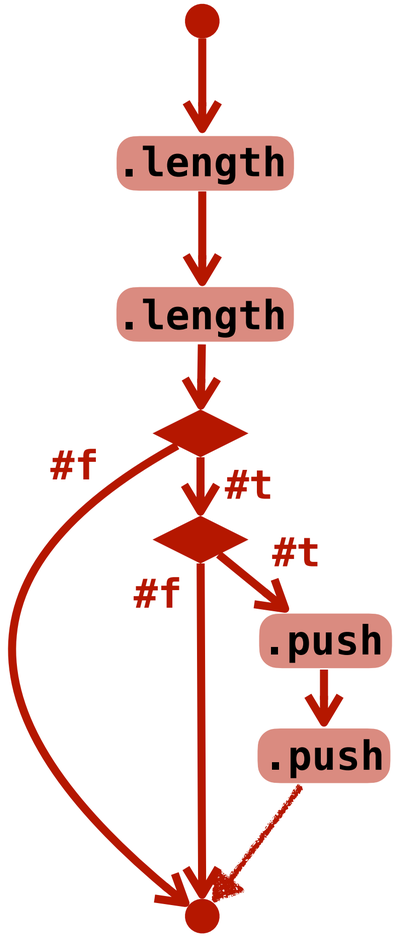
PLRG

Property-Order Graph (POG)



"remove" function in **Lodash.js v4.17.21**

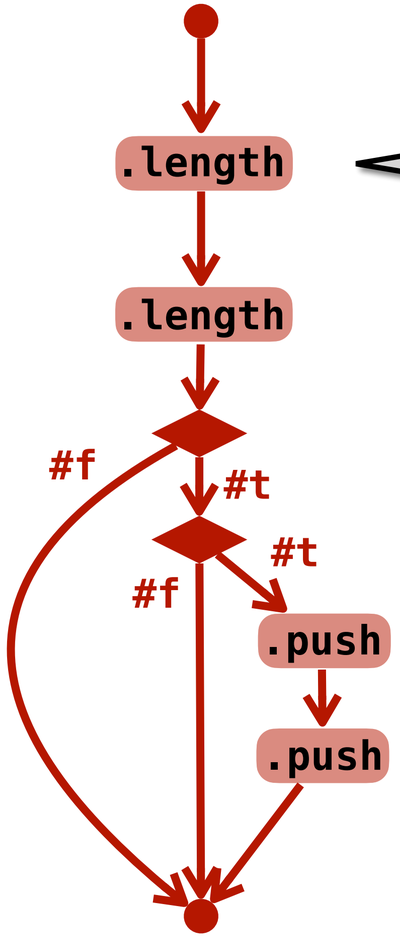
terser
transpile



Transpiled Code

PLRG

Property-Order Graph (POG)



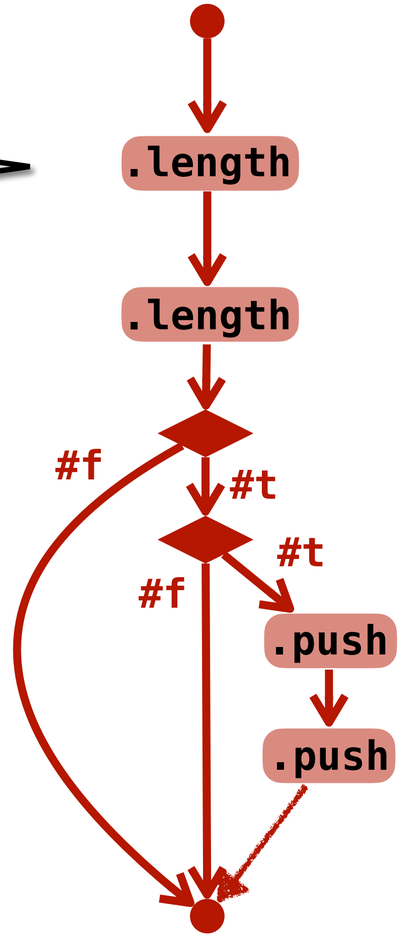
fingerprint:
0x3FA345B1

=

fingerprint:
0x3FA345B1

terser

transpile



"**remove**" function in **Lodash.js v4.17.21**

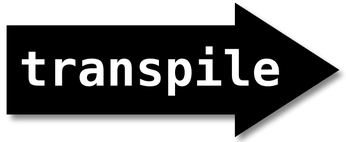
Transpiled Code

PLRG

Problem - POGs are NOT always Consistent

1. Flipped branch

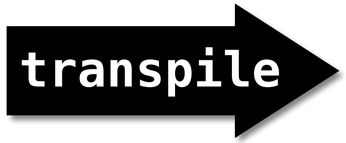
```
if (!y) x.p; else x.q;
```




```
if (y) x.q; else x.p;
```

2. Merged branch

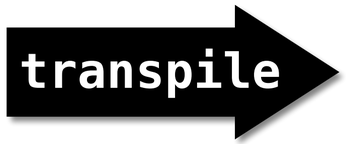
```
while (x.p) {
  if (x.q) break;
}
```




```
for (; x.p && !x.q; );
```

3. Combined duplicate code

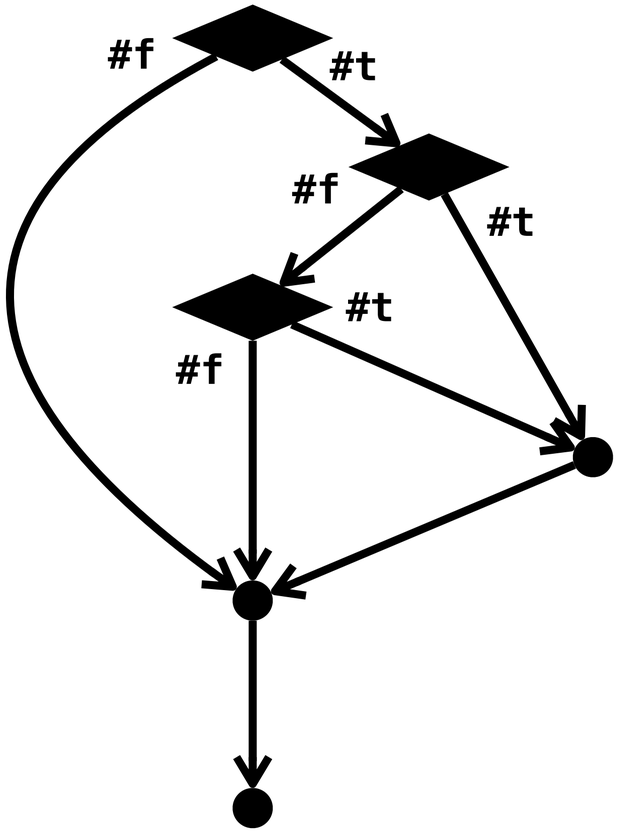
```
if (y) x.p = x.q = e1;
else   x.p = x.q = e2;
```




```
x.p = x.q = y ? e1 : e2;
```

PLRG

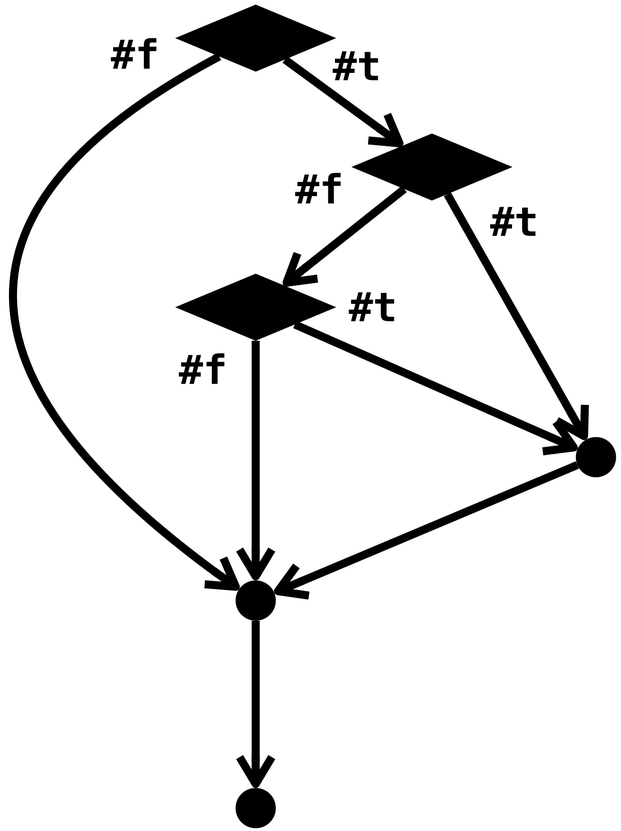
Solution - Refine POGs using Path-Sensitive Truthy Analysis



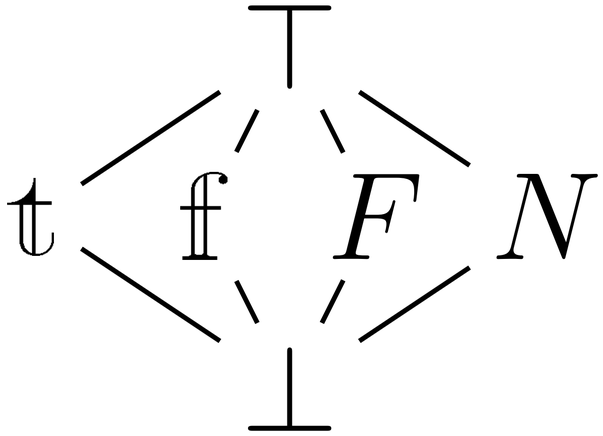
Track truthiness of variables along execution paths!
(Path = Control flow from each branch)

PLRG

Solution - Refine POGs using Path-Sensitive Truthy Analysis



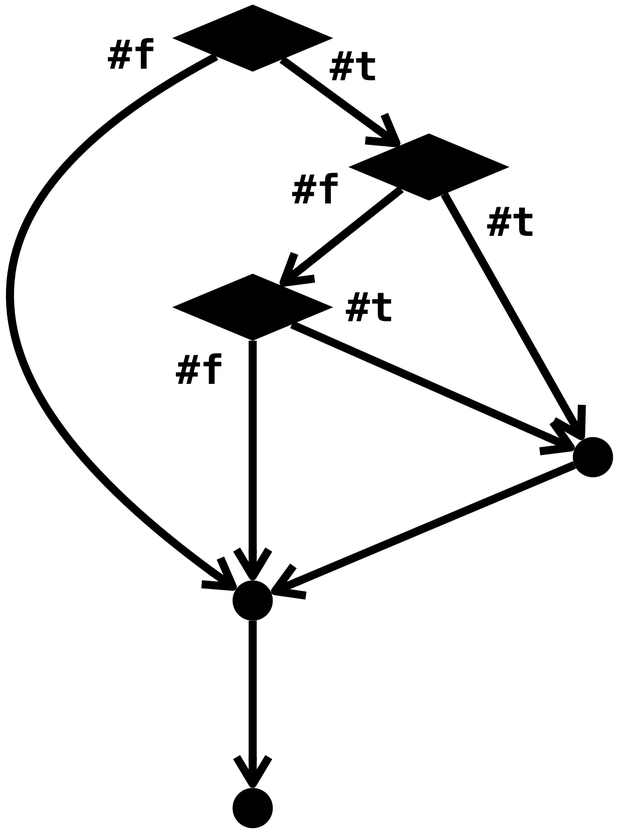
Track truthiness of variables along execution paths!
(Path = Control flow from each branch)



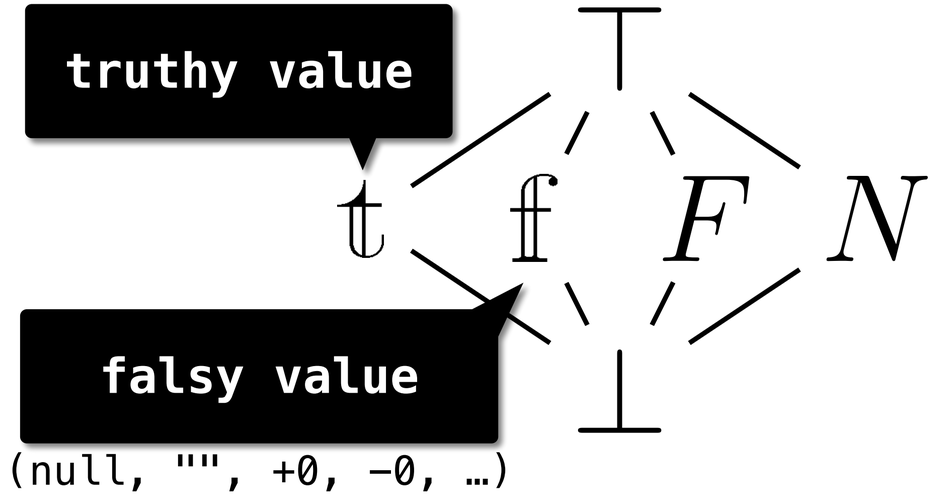
Abstract domain

PLRG

Solution - Refine POGs using Path-Sensitive Truthy Analysis



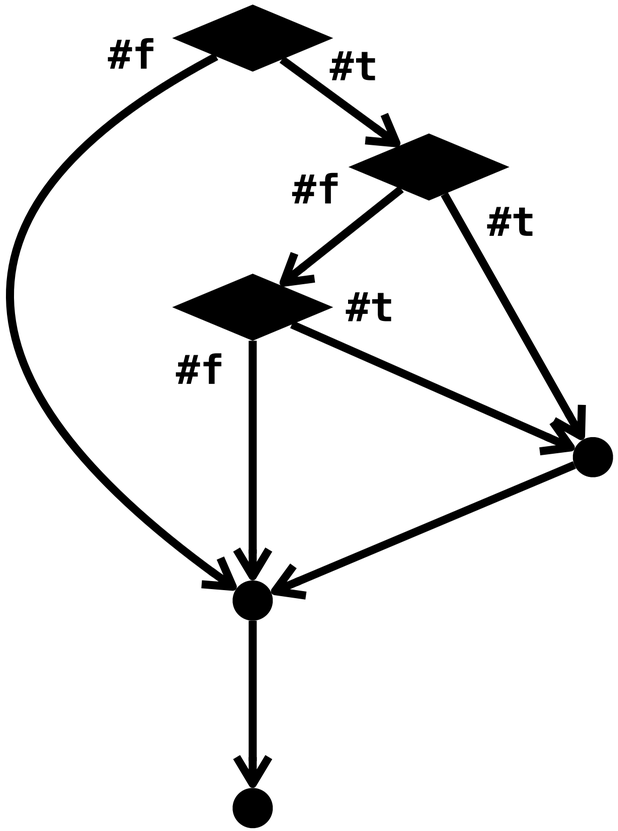
Track truthiness of variables along execution paths!
(Path = Control flow from each branch)



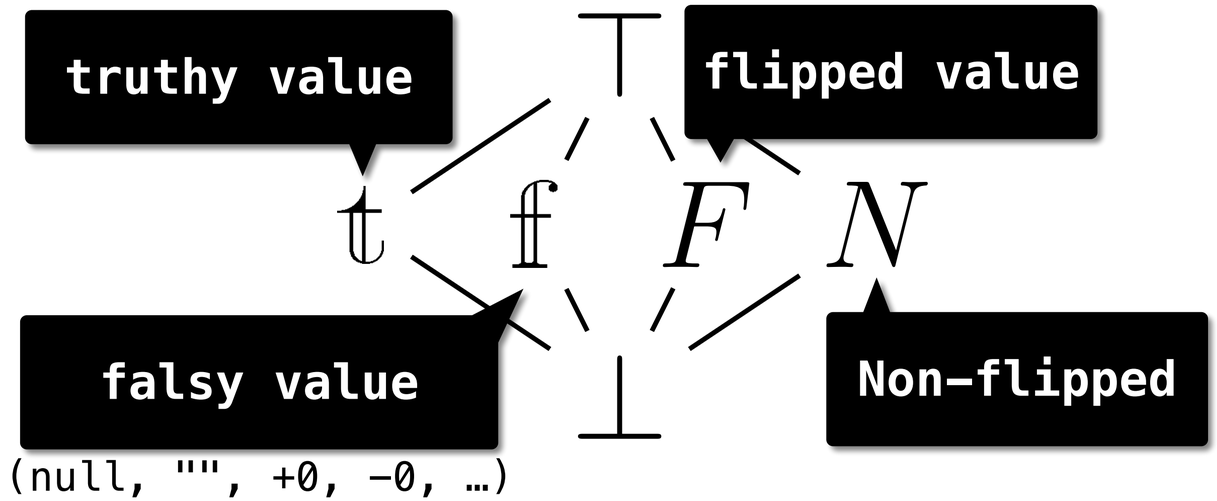
Abstract domain

PLRG

Solution - Refine POGs using Path-Sensitive Truthy Analysis



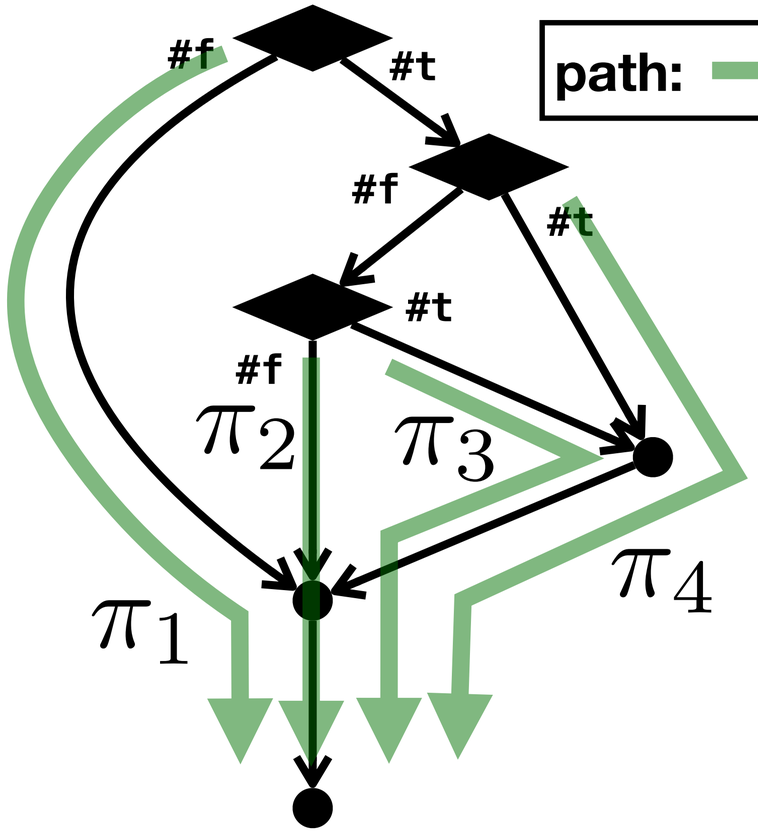
Track truthiness of variables along execution paths!
(Path = Control flow from each branch)



Abstract domain

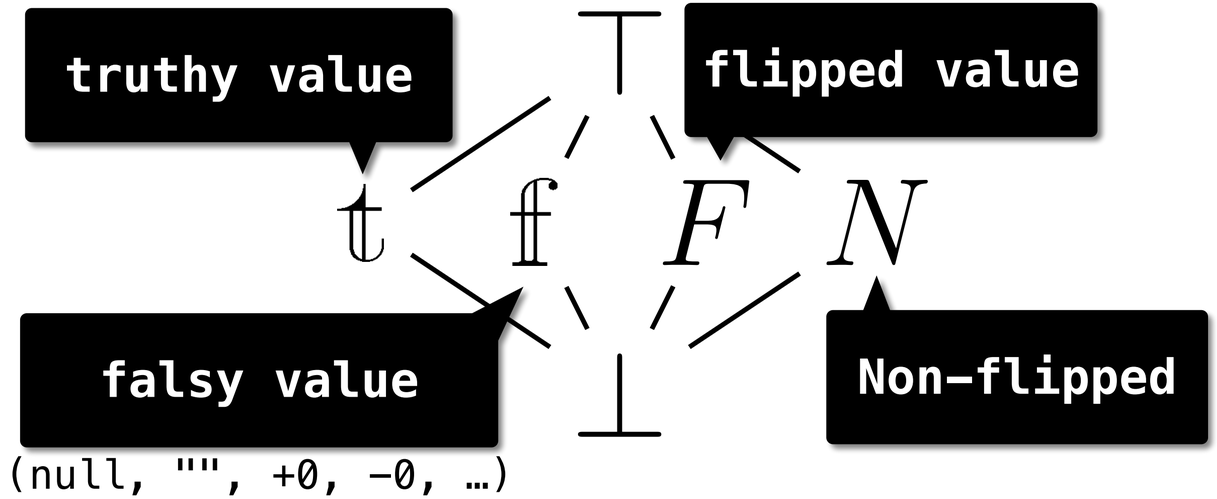
PLRG

Solution - Refine POGs using Path-Sensitive Truthy Analysis



path:

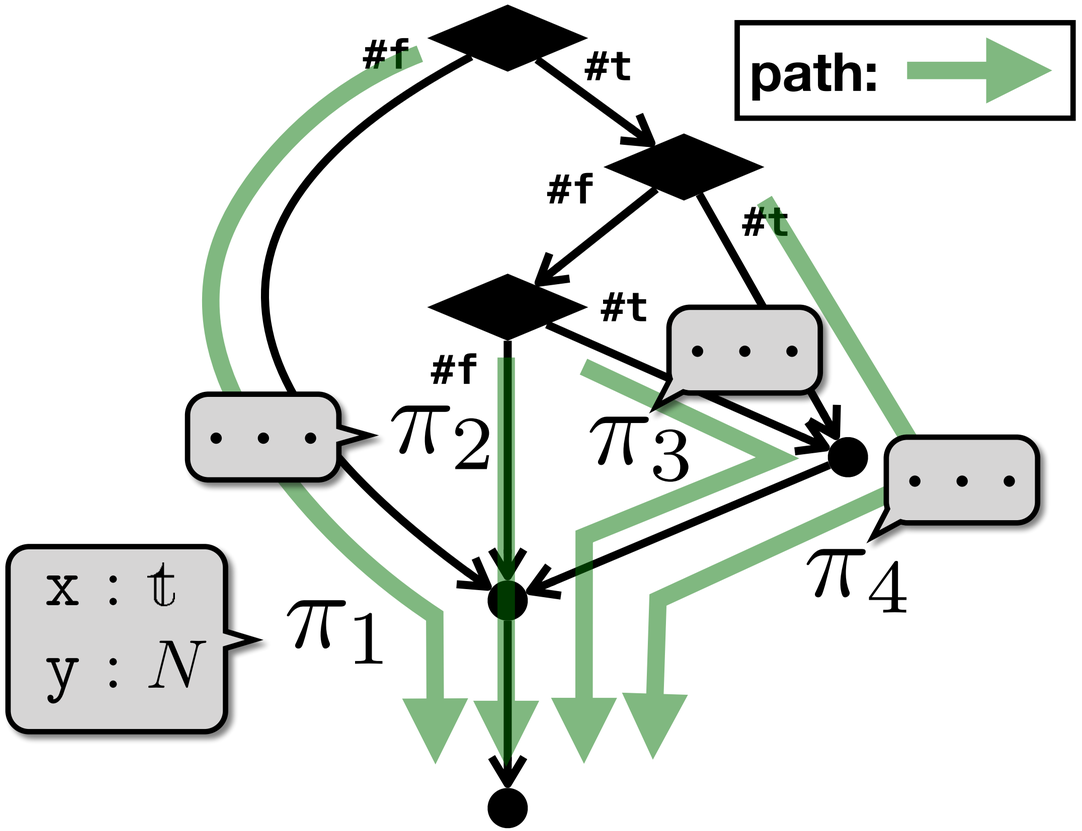
Track truthiness of variables along execution paths!
(Path = Control flow from each branch)



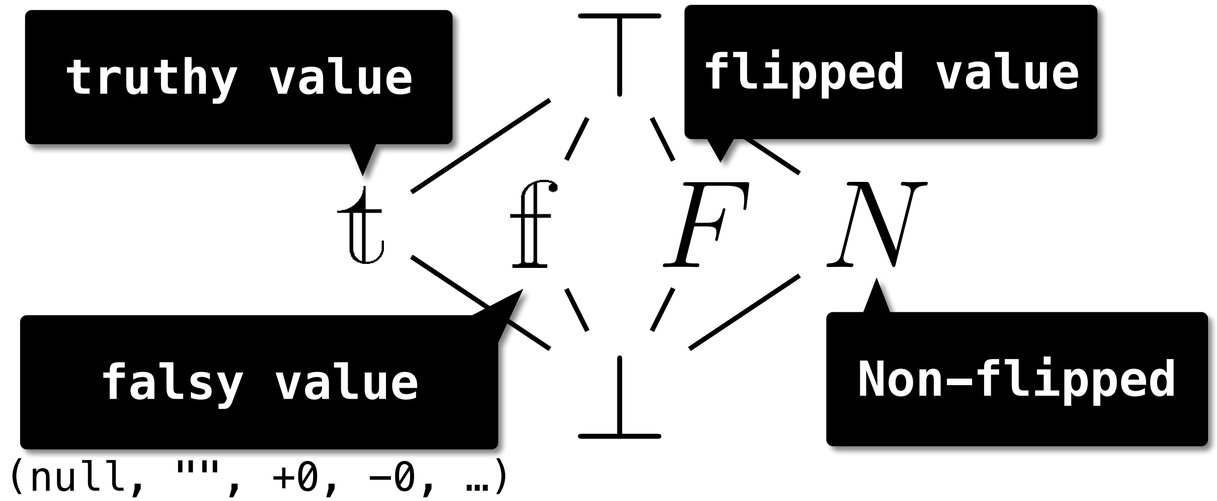
Abstract domain

PLRG

Solution - Refine POGs using Path-Sensitive Truthy Analysis



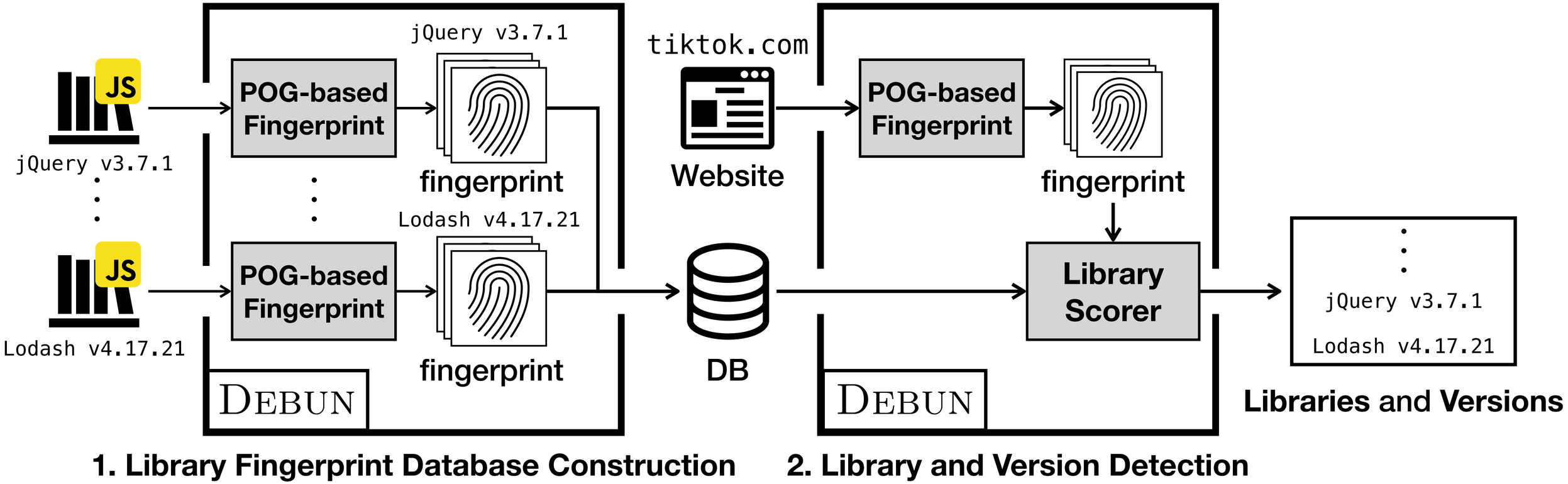
Track truthiness of variables along execution paths!
(Path = Control flow from each branch)



Abstract domain

PLRG

Debun - A POG-based Library Detector



PLRG

Evaluation

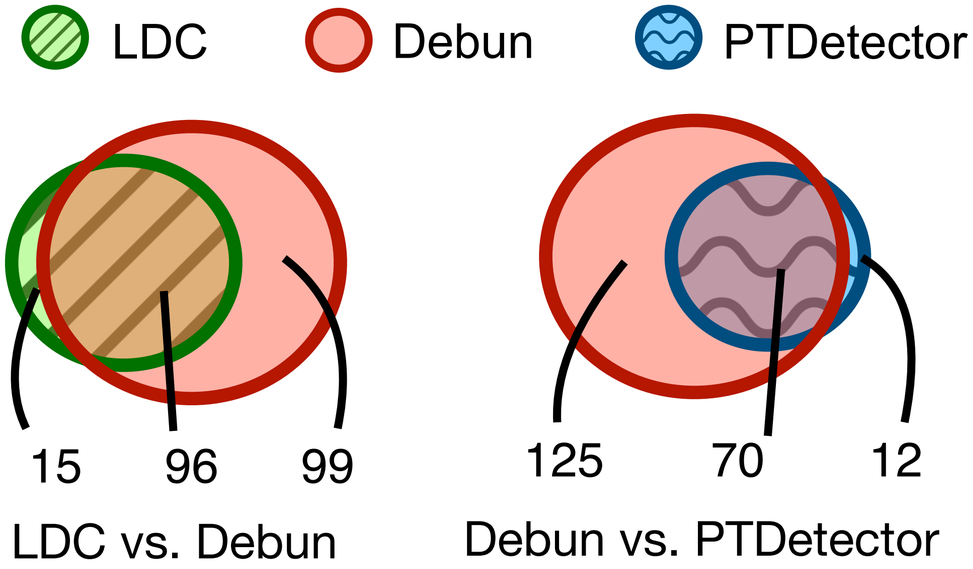
- Constructed fingerprint database from **8,256 versions** of **78 libraries** registered on `cdnjs`
- Evaluated on **68 crawlable websites** from the top 100 high-traffic websites
- **RQ1) Library detection** performance compared to existing techniques
- **RQ2) Library version detection** performance compared to existing techniques
- **RQ3) Ablation study** - effectiveness of analysis-based graph refinement

PLRG

RQ1) Library Detection

- Average detection time: **1,009 ms per website**

Metric	LDC	PTDETECTOR	DEBUN
TP	111	82	195
FP	3	9	7
FN	112	141	28
Precision	97.37%	90.11%	96.53%
Recall	49.78%	36.77%	87.44%
F1-score	65.88%	52.23%	91.76%



LDC vs. Debun

Debun vs. PTDetector

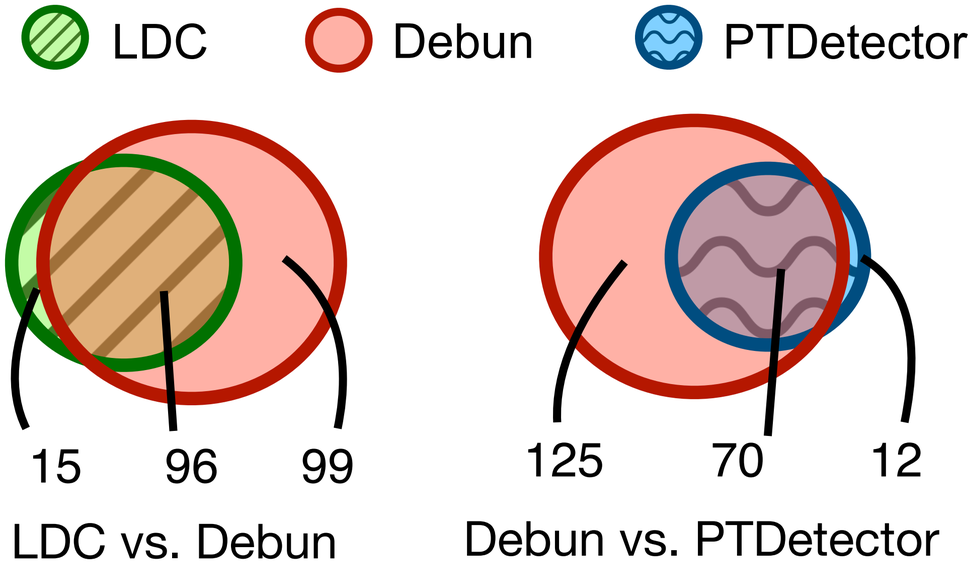
PLRG

RQ1) Library Detection

- Average detection time: **1,009 ms per website**

Metric	LDC	PTDetector	Debun
TP	111	82	195
FP	3	9	7
FN	112	141	28
Precision	97.37%	90.11%	96.53%
Recall	49.78%	36.77%	87.44%
F1-score	65.88%	52.23%	91.76%

+25.88%p



PLRG

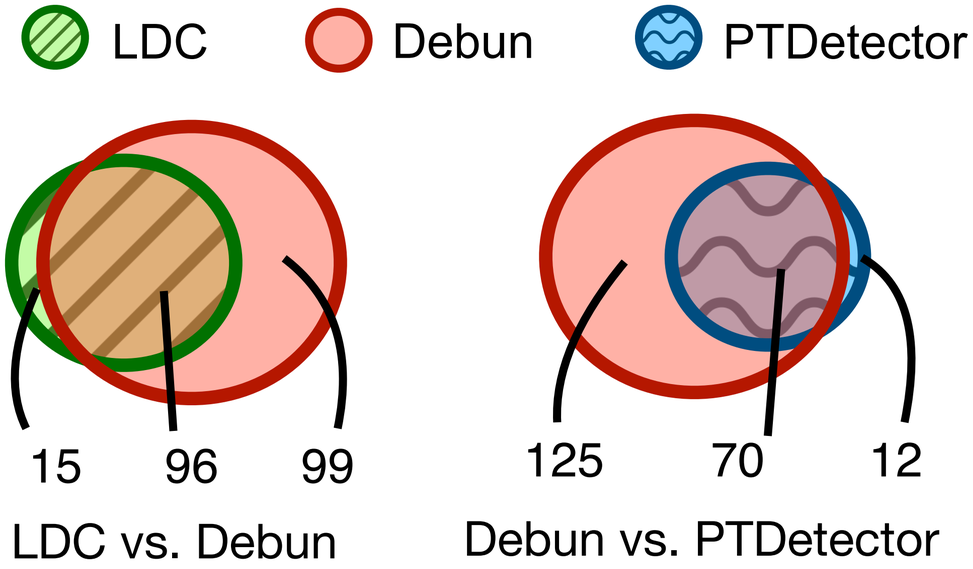
RQ1) Library Detection

- Average detection time: **1,009 ms per website**

Metric	LDC	PTDetector	Debun
TP	111	82	195
FP	3	9	7
FN	112	141	28
Precision	97.37%	90.11%	96.53%
Recall	49.78%	36.77%	87.44%
F1-score	65.88%	52.23%	91.76%

+39.53%p

+25.88%p



PLRG

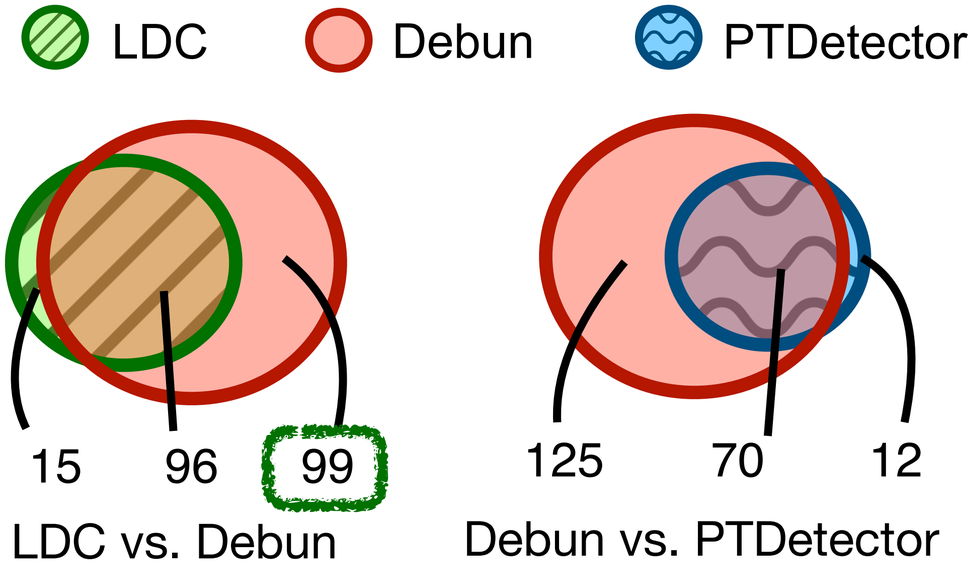
RQ1) Library Detection

- Average detection time: **1,009 ms per website**

Metric	LDC	PTDetector	Debun
TP	111	82	195
FP	3	9	7
FN	112	141	28
Precision	97.37%	90.11%	96.53%
Recall	49.78%	36.77%	87.44%
F1-score	65.88%	52.23%	91.76%

+39.53%p

+25.88%p



PLRG

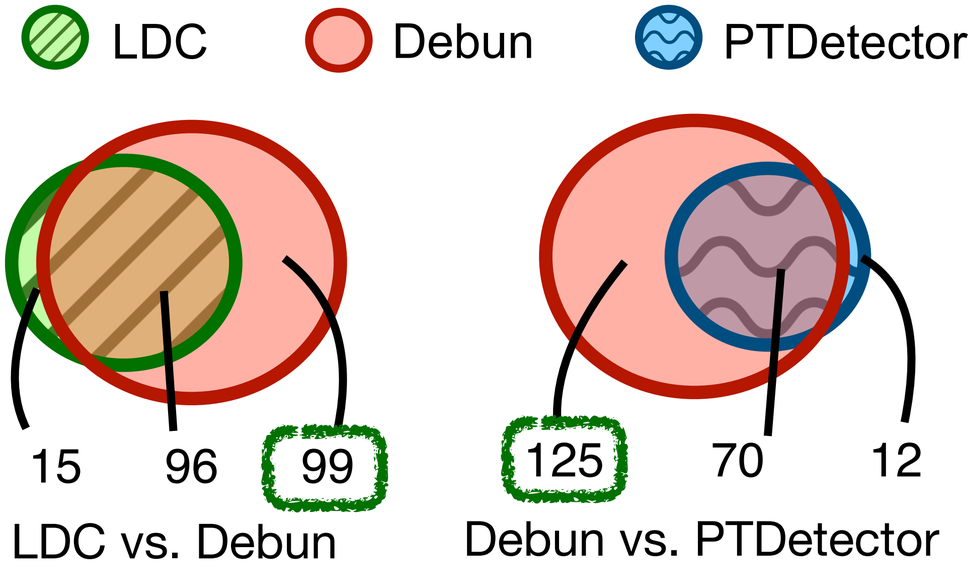
RQ1) Library Detection

- Average detection time: **1,009 ms per website**

Metric	LDC	PTDetector	Debun
TP	111	82	195
FP	3	9	7
FN	112	141	28
Precision	97.37%	90.11%	96.53%
Recall	49.78%	36.77%	87.44%
F1-score	65.88%	52.23%	91.76%

+39.53%p

+25.88%p

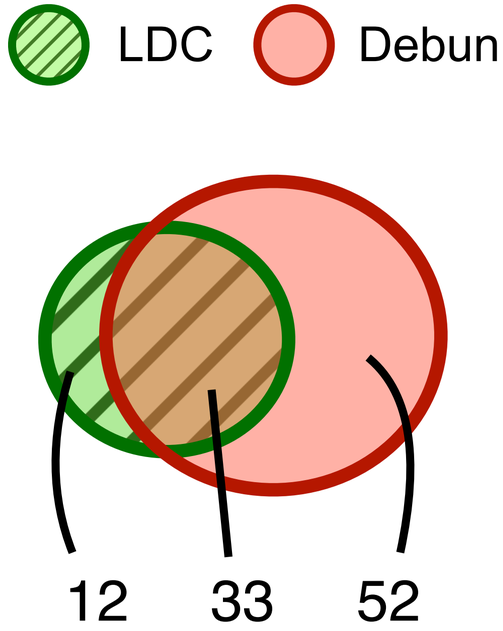


PLRG

RQ2) Library Version Detection

- Average detection time: **1,009 ms per website**

Metric	LDC	DEBUN
TP	45	85
FP	0	16
FN	60	20
Precision	100.00%	84.16%
Recall	42.86%	80.95%
F1 score	60.00%	82.52%



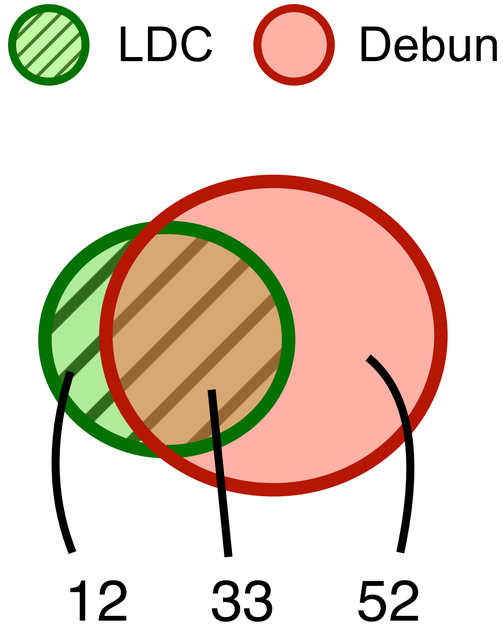
PLRG

RQ2) Library Version Detection

- Average detection time: **1,009 ms per website**

Metric	LDC	DEBUN
TP	45	85
FP	0	16
FN	60	20
Precision	100.00%	84.16%
Recall	42.86%	80.95%
F1 score	60.00%	82.52%

+22.52%p



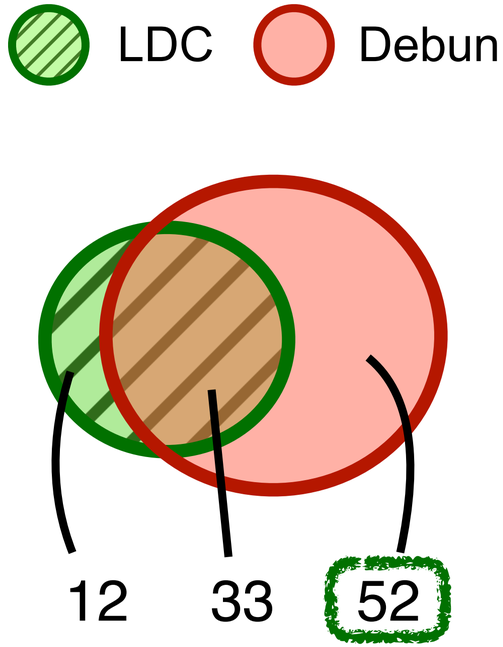
PLRG

RQ2) Library Version Detection

- Average detection time: **1,009 ms per website**

Metric	LDC	DEBUN
TP	45	85
FP	0	16
FN	60	20
Precision	100.00%	84.16%
Recall	42.86%	80.95%
F1 score	60.00%	82.52%

+22.52%p



PLRG

RQ3) Ablation study - Analysis-based Refinement

Consistency = (Ratio of **Preserved Fingerprints** after Transpiration)

Metric	Count	POG	POG+F	POG+FB	POG+FBC
# Consistent	47,385	35,370	43,358	45,404	45,522
# Functions	54,368	54,368	54,368	54,368	54,368
Consistency	87.16%	65.06%	79.75%	83.51%	83.73%

Count: w/o execution order

F: Branch Flipping
B: Branch Bypassing
C: Path Cloning

Accuracy = (Degree of **Fingerprint Uniqueness**)

Metric	Count	POG	POG+F	POG+FB	POG+FBC
# Functions	55,518	55,518	55,518	55,518	55,518
# Duplicated	171,5034	274,252	273,252	273,678	273,684
Accuracy	3.28%	20.24%	20.32%	20.29%	20.29%

PLRG

RQ3) Ablation study - Analysis-based Refinement

Consistency = (Ratio of **Preserved Fingerprints** after Transpiration)

Metric	Count	POG	POG+F	POG+FB	POG+FBC
# Consistent	47,385	35,370	43,358	45,404	45,522
# Functions	54,368	54,368	54,368	54,368	54,368
Consistency	87.16%	65.06%	79.75%	83.51%	83.73%

Count: w/o execution order

F: Branch Flipping
B: Branch Bypassing
C: Path Cloning

+18.67%p

Accuracy = (Degree of **Fingerprint Uniqueness**)

Metric	Count	POG	POG+F	POG+FB	POG+FBC
# Functions	55,518	55,518	55,518	55,518	55,518
# Duplicated	171,5034	274,252	273,252	273,678	273,684
Accuracy	3.28%	20.24%	20.32%	20.29%	20.29%

PLRG

RQ3) Ablation study - Analysis-based Refinement

Consistency = (Ratio of Preserved Fingerprints after Transpiration)

Metric	Count	POG	POG+F	POG+FB	POG+FBC
# Consistent	47,385	35,370	43,358	45,404	45,522
# Functions	54,368	54,368	54,368	54,368	54,368
Consistency	87.16%	65.06%	79.75%	83.51%	83.73%

Count: w/o execution order

F: Branch Flipping
B: Branch Bypassing
C: Path Cloning

+18.67%p

Accuracy = (Degree of Fingerprint Uniqueness)

Metric	Count	POG	POG+F	POG+FB	POG+FBC
# Functions	55,518	55,518	55,518	55,518	55,518
# Duplicated	171,5034	274,252	273,252	273,678	273,684
Accuracy	3.28%	20.24%	20.32%	20.29%	20.29%

Low Accuracy

PLRG

RQ3) Ablation study - Analysis-based Refinement

Consistency = (Ratio of **Preserved Fingerprints** after Transpiration)

Metric	Count	POG	POG+F	POG+FB	POG+FBC
# Consistent	47,385	35,370	43,358	45,404	45,522
# Functions	54,368	54,368	54,368	54,368	54,368
Consistency	87.16%	65.06%	79.75%	83.51%	83.73%

Count: w/o execution order

F: Branch Flipping
B: Branch Bypassing
C: Path Cloning

+18.67%p

Accuracy = (Degree of **Fingerprint Uniqueness**)

Metric	Count	POG	POG+F	POG+FB	POG+FBC
# Functions	55,518	55,518	55,518	55,518	55,518
# Duplicated	171,5034	274,252	273,252	273,678	273,684
Accuracy	3.28%	20.24%	20.32%	20.29%	20.29%

Low Accuracy

Mostly Preserved

PLRG

KOREA UNIVERSITY

PLRG

Debun (Artifact)

Video

FSE ASE 2025