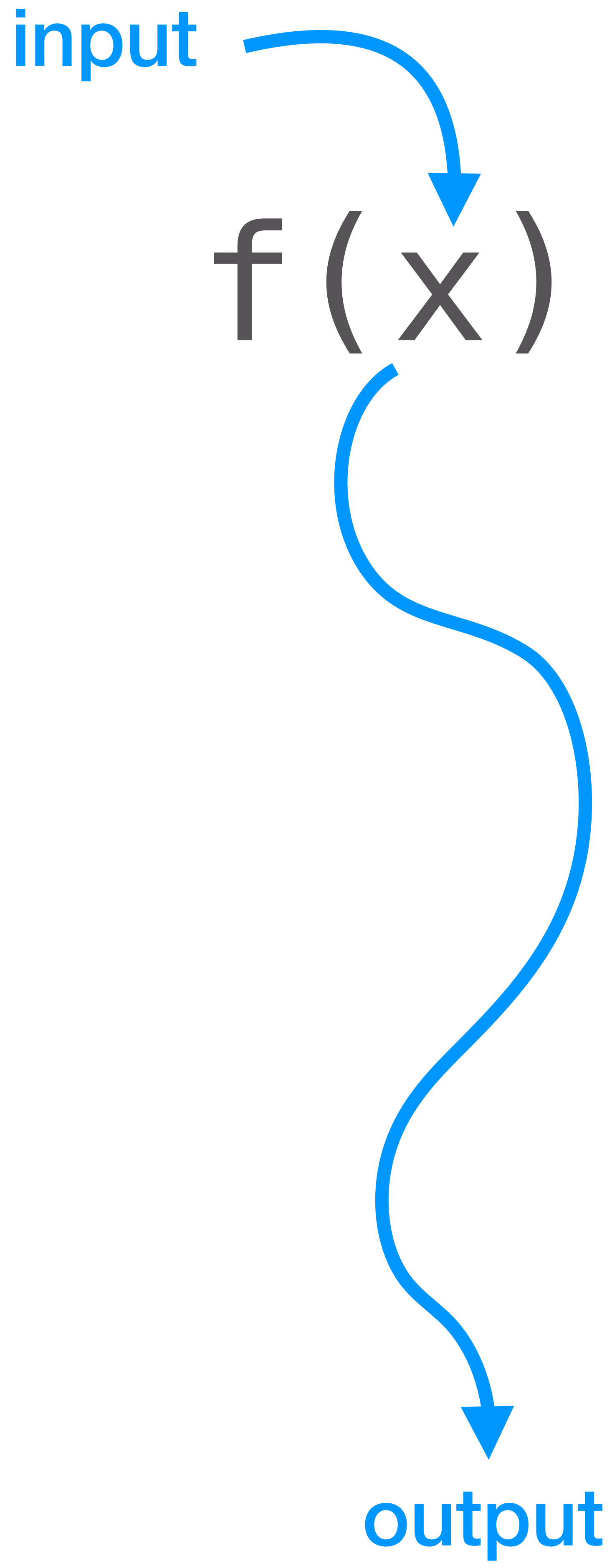
Purity Analysis for JavaScript

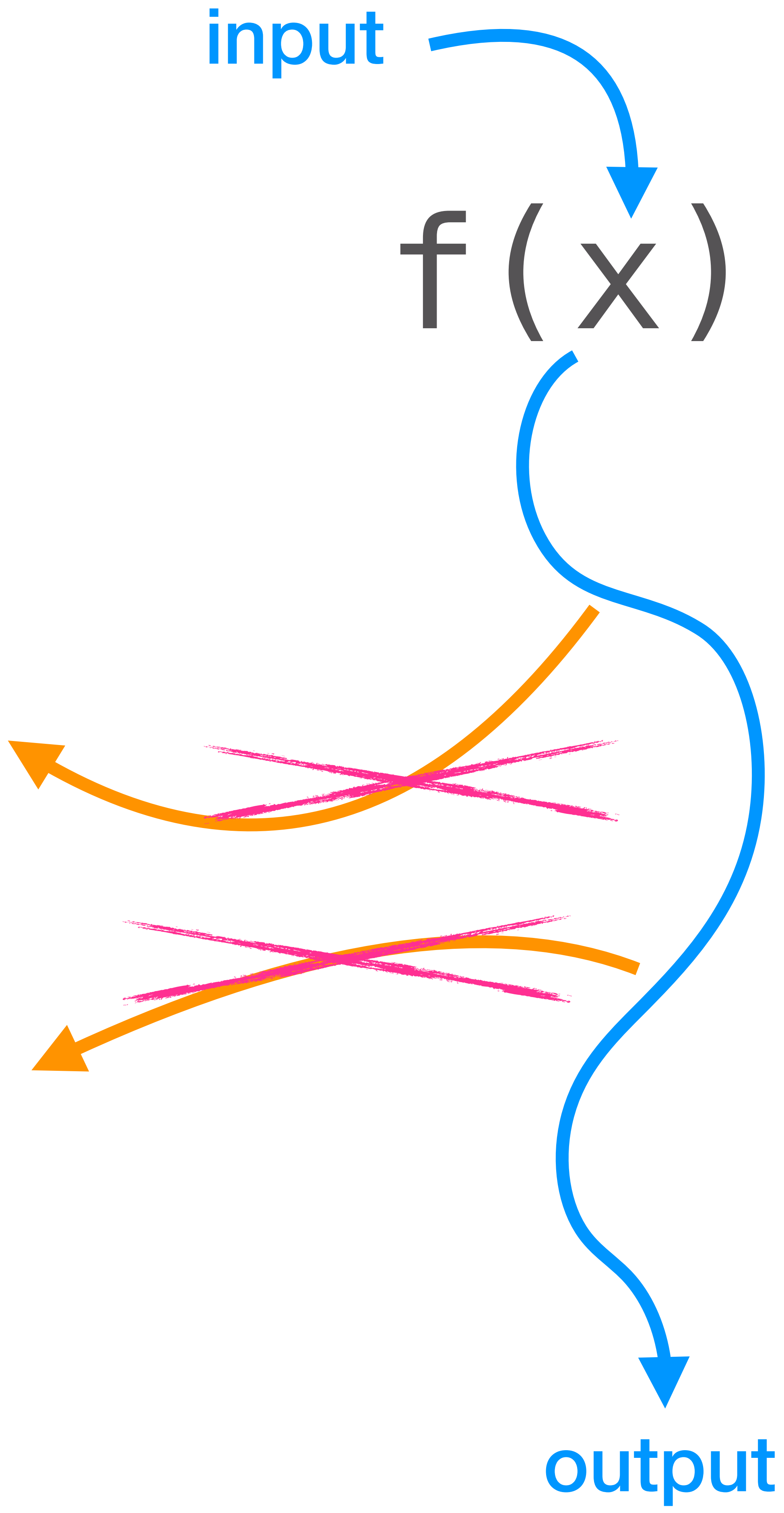
Jens Nicolay

Software Languages Lab

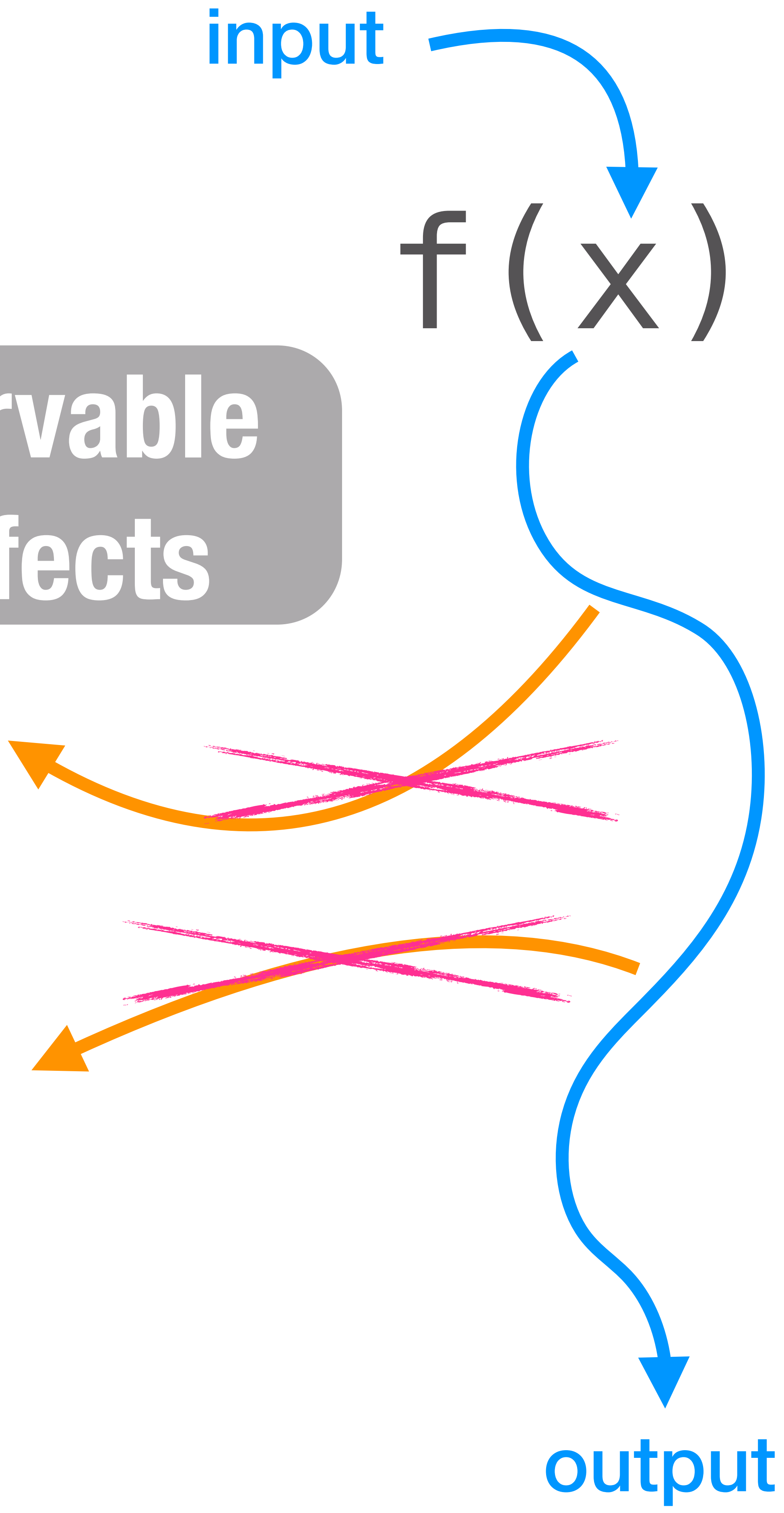
Vrije Universiteit Brussel

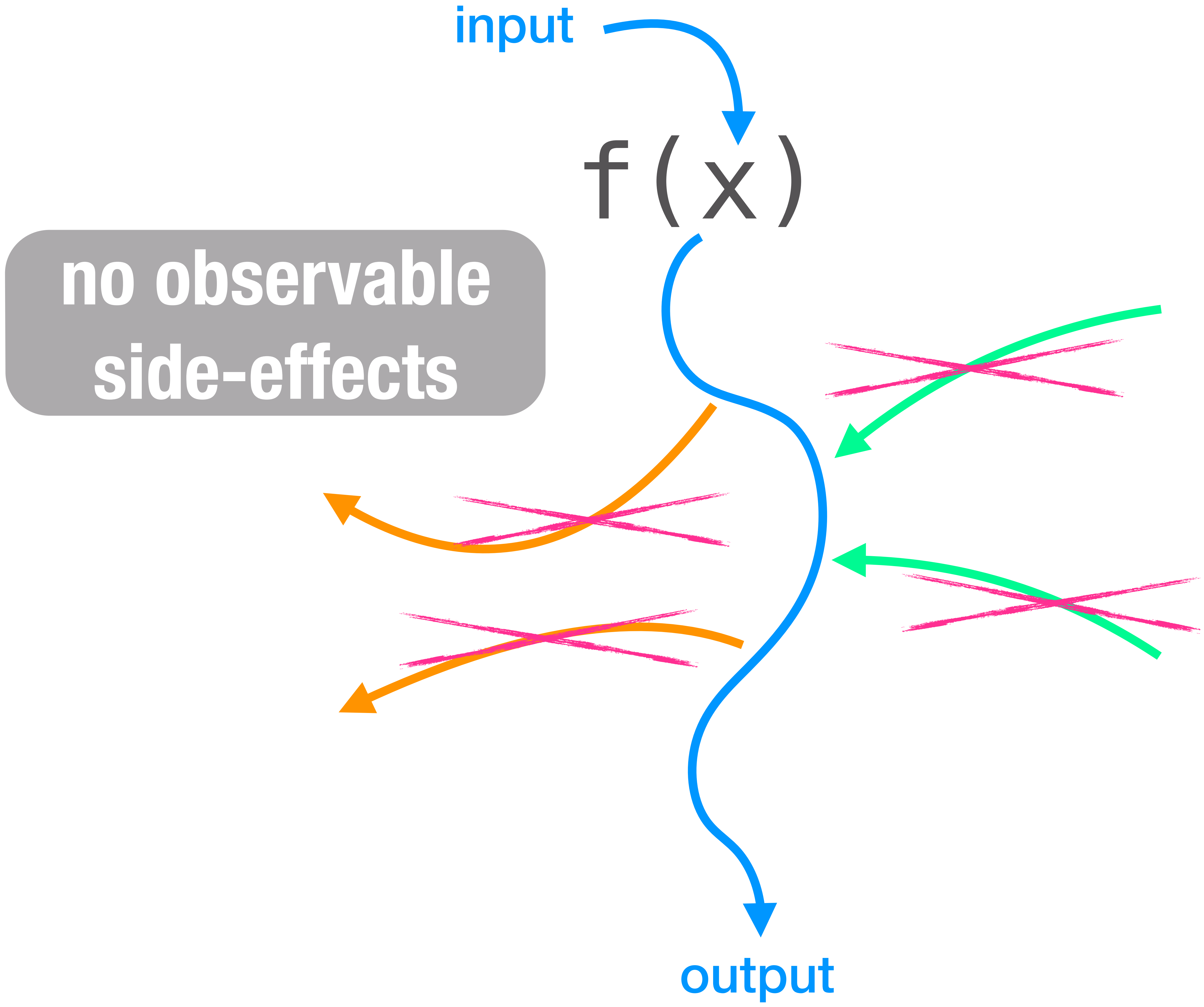
Purity Analysis for JavaScript

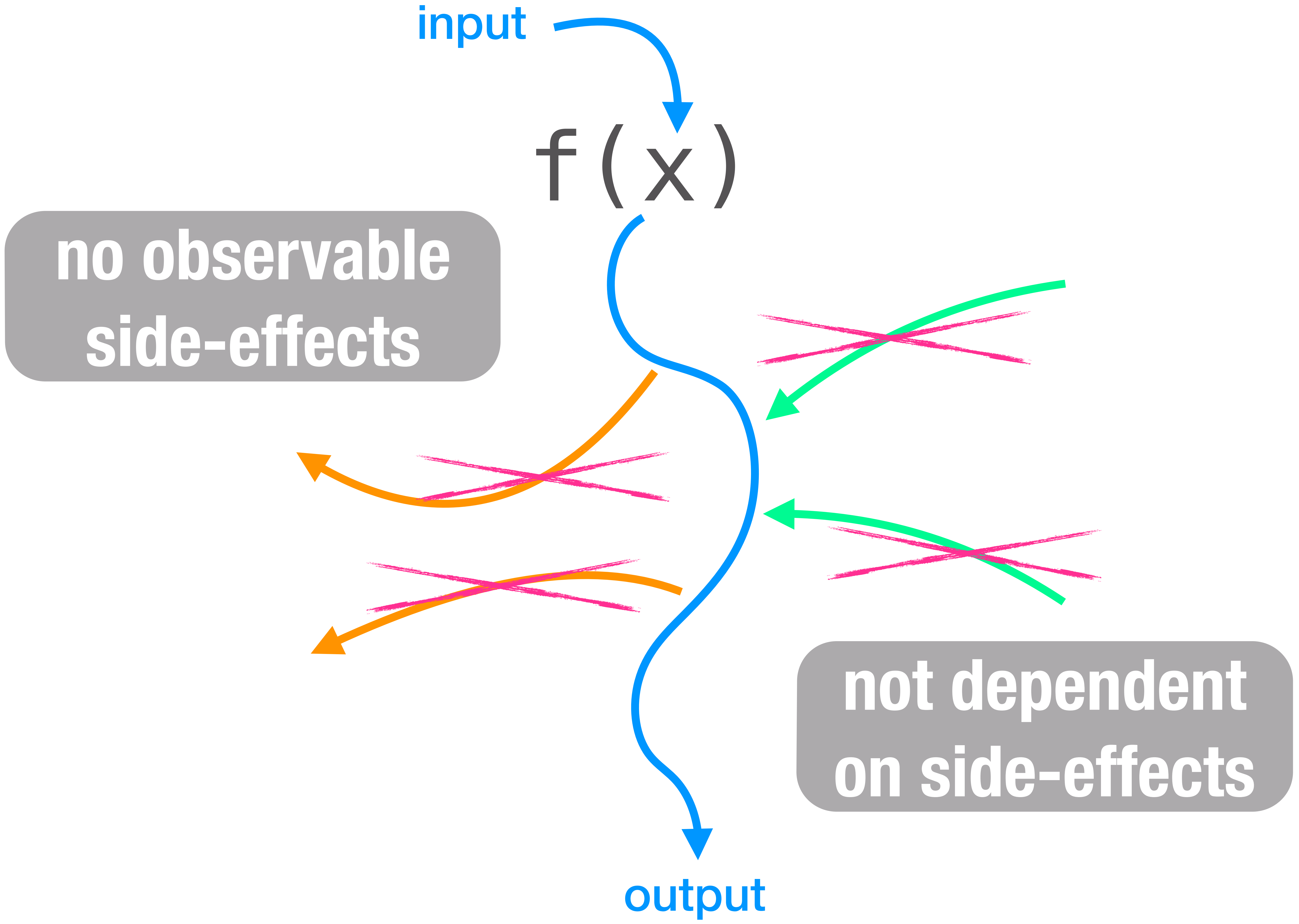

input
f(x)
output


input
f(x)
output

no observable side-effects




input
f(x)
no observable side-effects
output


input
f(x)
no observable side-effects
not dependent on side-effects
output

Purity

no observable side-effects

not dependent on side-effects

```
function compute(i, j)
{
    …
    let k = f(i) + x * f(j);
    …
    return Math.max(k, f(0));
}
```

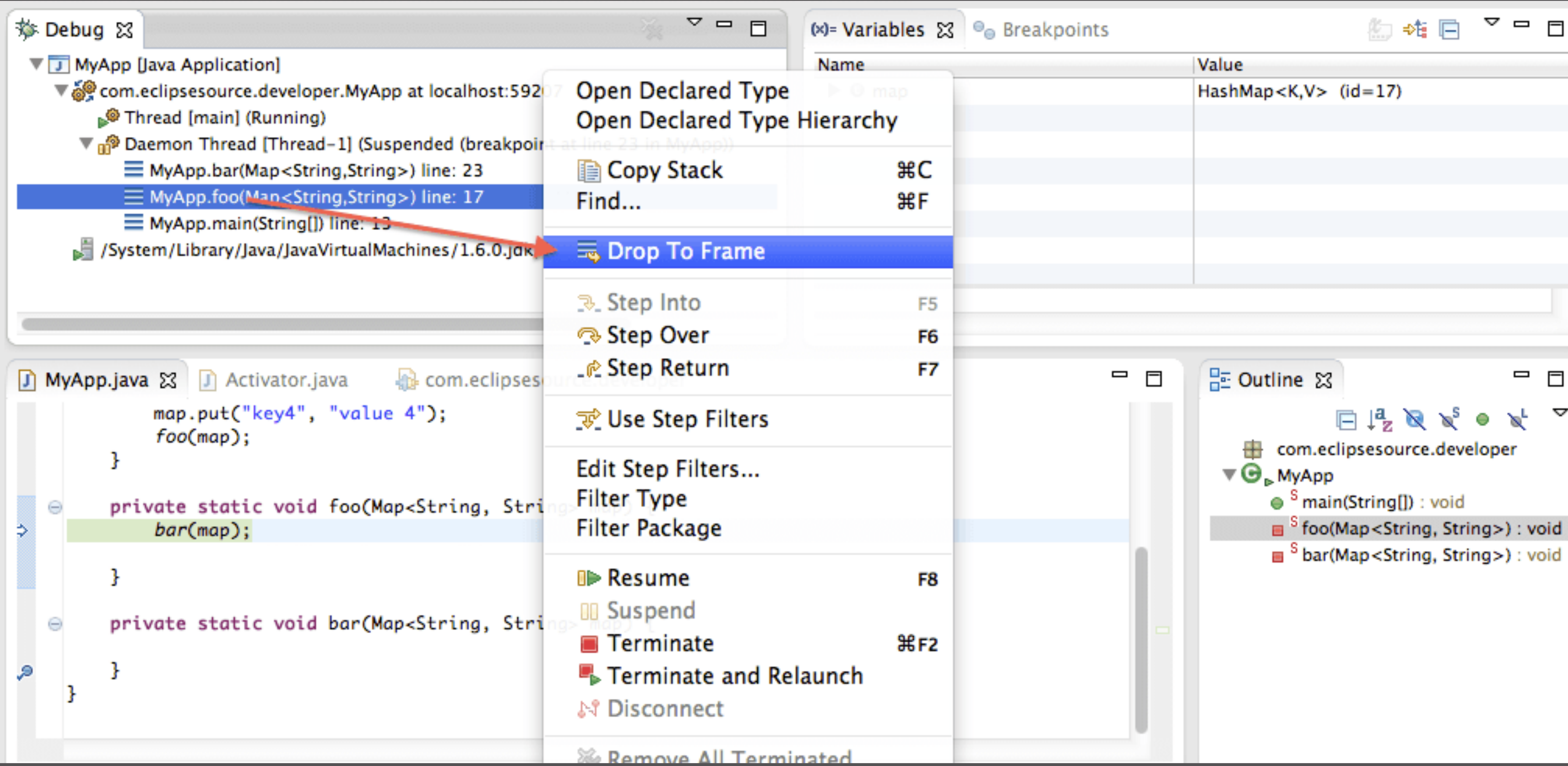
```
function compute(i, j)
{
    …      42
    let k = ~~f(i)~~ + x * ~~f(j);~~
    …
    return Math.max(k, ~~f(0)~~);
}
```

```
function compute(i, j)
{
   …
   let k = ~~f(i) + x * f(j);~~
   …
   return ~~Math.max(k, f(0));~~
}
```

```
function compute(i, j)
{
   …
   let k = f(i) + x * f(j);
   …
   return Math.max(k, f(0));
}
```

```
// @pre i++ > 0
function compute(i, j)
{
    …
    assertFalse(k = 0)
    …
}
```

```
// @pre ~~i++~~ > 0
function compute(i, j)
{
   …
   assertFalse(~~k = 0~~)
   …
}
```

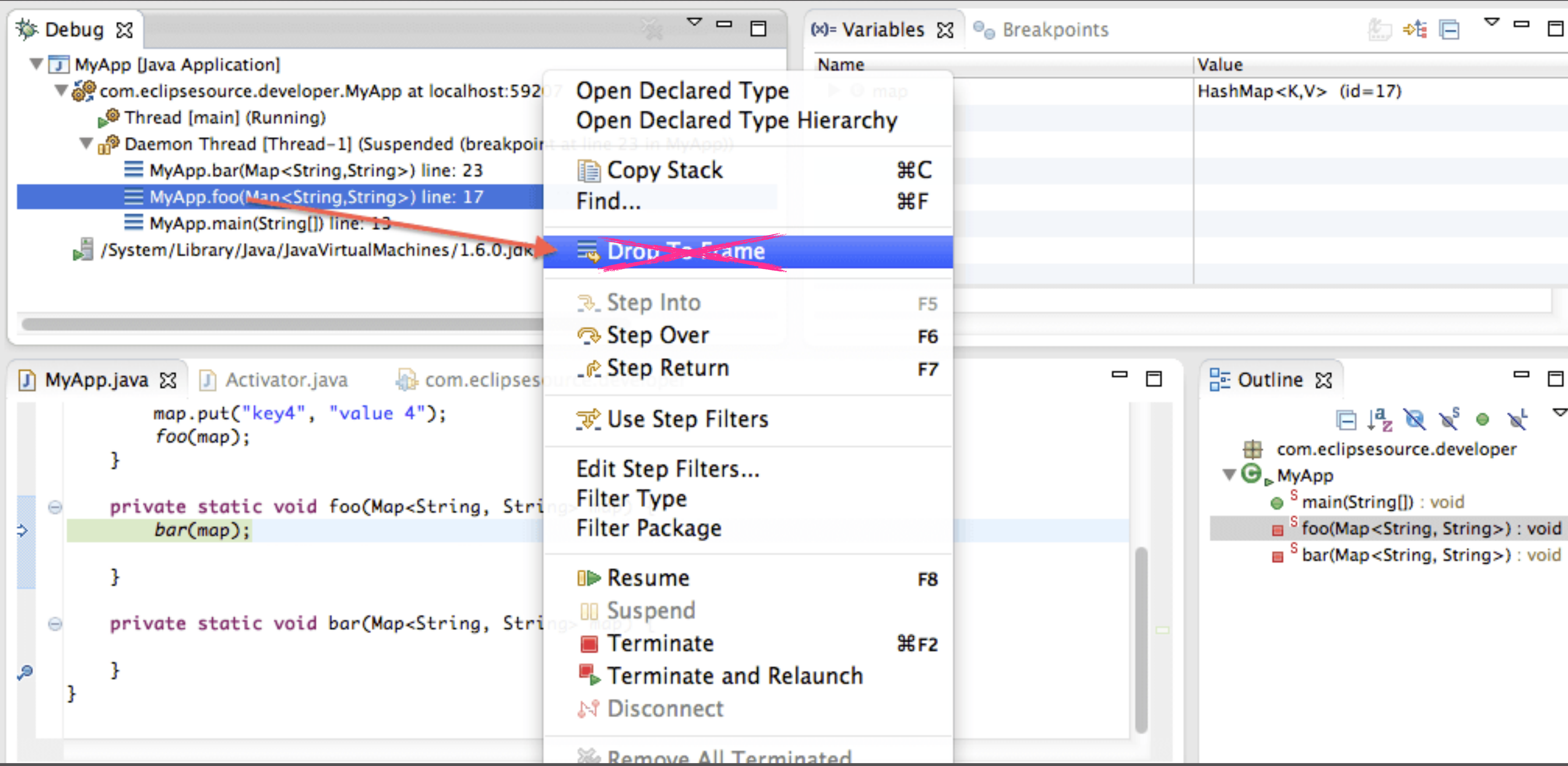


```
MyApp [Java Application]
  com.eclipsesource.developer.MyApp at localhost:592
    Thread [main] (Running)
    Daemon Thread [Thread-1] (Suspended (breakpoi
      MyApp.bar(Map<String,String>) line: 23
      MyApp.foo(Map<String,String>) line: 17
      MyApp.main(String[]) line: 13
  /System/Library/Java/JavaVirtualMachines/1.6.0.jdk
```

```
HashMap<K,V>  (id=17)
```

```
        map.put("key4", "value 4");
        foo(map);
    }

    private static void foo(Map<String, Stri
        bar(map);

    }

    private static void bar(Map<String, Stri

    }
}
```

```
com.eclipsesource.developer
MyApp
  main(String[]) : void
  foo(Map<String, String>) : void
  bar(Map<String, String>) : void
```


Debug
MyApp [Java Application]
com.eclipsesource.developer.MyApp at localhost:59207
Thread [main] (Running)
Daemon Thread [Thread-1] (Suspended (breakpoint at line 23 in MyApp))
MyApp.bar(Map<String,String>) line: 23
MyApp.foo(Map<String,String>) line: 17
MyApp.main(String[]) line: 13
/System/Library/Java/JavaVirtualMachines/1.6.0.jdk
Variables
Breakpoints
Name
Value
map
HashMap<K,V> (id=17)
Open Declared Type
Open Declared Type Hierarchy
Copy Stack ⌘C
Find... ⌘F
~~Drop To Frame~~
Step Into F5
Step Over F6
Step Return F7
Use Step Filters
Edit Step Filters...
Filter Type
Filter Package
Resume F8
Suspend
Terminate ⌘F2
Terminate and Relaunch
Disconnect
Remove All Terminated
MyApp.java
Activator.java
com.eclipsesource.developer
Outline
com.eclipsesource.developer
MyApp
main(String[]) : void
foo(Map<String, String>) : void
bar(Map<String, String>) : void

```
        map.put("key4", "value 4");
        foo(map);
    }

    private static void foo(Map<String, Stri
        bar(map);

    }

    private static void bar(Map<String, Stri

    }
}
```

eclipsesource.com

- complexity reduction
- optimization
- reproducibility
- safety

Purity Analysis for JavaScript

- functional, object-oriented, side-effecting language
- JavaScript quirks

- functional, object-oriented, side-effecting language
- JavaScript quirks

```
function Foo(bar)
{
  this.bar = bar;
}

function f(foo)
{
  foo.bar = !foo.bar;
}

var foo = new Foo(true);
f(foo)
```

```
function Foo(bar)
{
  this.bar = bar;
}

function f(foo)
{
  foo.bar = !foo.bar;
}

var foo = new Foo(true);
f(foo)
```

not pure

```
function Foo(bar)
{
  this.bar = bar;
}

function f()
{
  var foo2 = new Foo(false);
  foo2.bar = true;
  return foo2.bar;
}

f()
```

```
function Foo(bar)
{
  this.bar = bar;
}

function f()
{
  var foo2 = new Foo(false);
  foo2.bar = true;
  return foo2.bar;
}

f()
```

pure

```
var z = false;

function f()
{
  return z;
}

f();
z=true;
f()
```

```
var z = false;

function f()
{
  return z;
}

f();
z=true;
f()
```

not pure

```
function f()
{
   var x=0;
   function g() {x=x+1};
   g()
}

f()
```

```
function f()
{
  var x=0;
  function g() {x=x+1};
  g()
}

f()
```

pure

```
function f()
{
  return g();
}
```

?

```
function f(g)
{
  return g();
}
```

?

```
function f()
{
   return o.x;
}
```

?

- function purity does not depend on function alone
- assignment allowed
- free variables allowed
- calling impure functions allowed

Purity Analysis for JavaScript

- control flow
- value flow
- read/write effects

JIPDA

JavaScript Introspective Pushdown Analysis

A Tool-Builder’s Tool

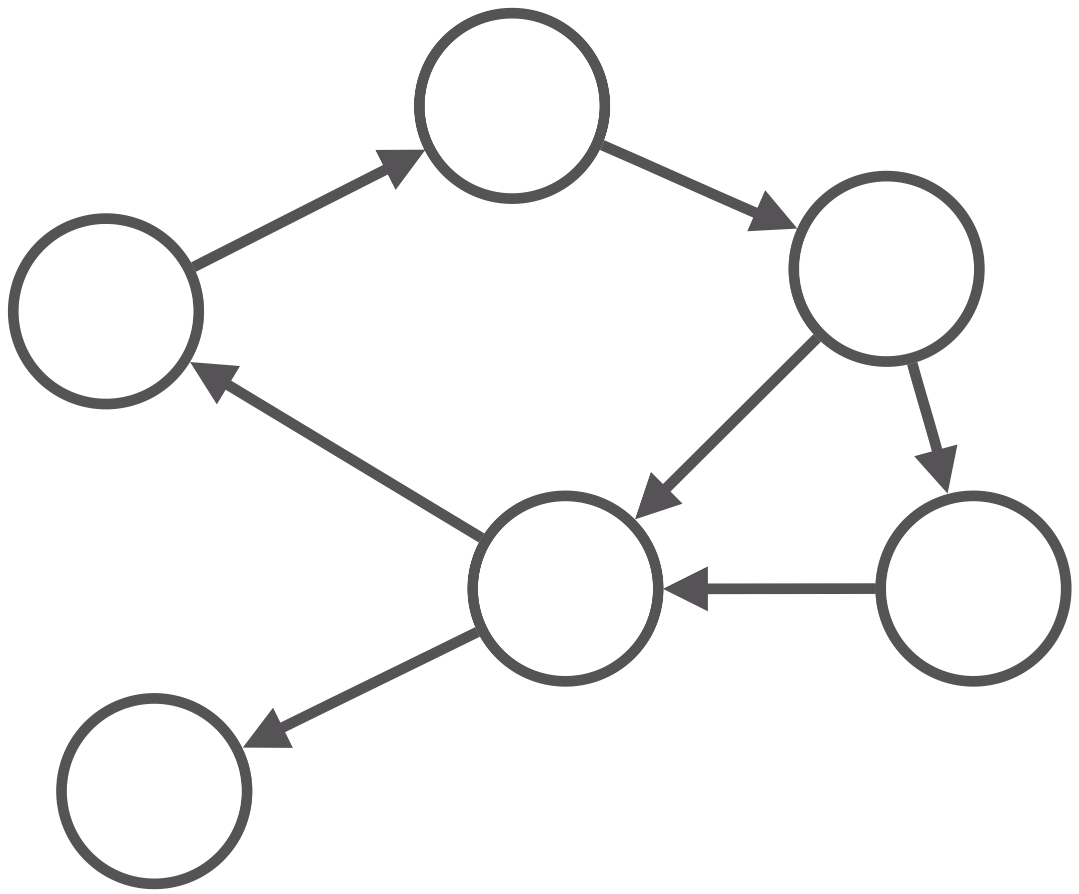
```
Pushdown.pathsBwTo =
  function (s, target, etg)
  {
    var todo = [s];
    var visited = ArraySet.empty();
    var paths = ArraySet.empty();
    while (todo.length > 0)
    {
      var q = todo.shift();
      if (q.equals(target) || visited.contains(q))
      {
        continue;
      }
      visited = visited.add(q);
      var incoming = etg.incoming(q)
      paths = paths.addAll(incoming);
      var qs = incoming.map(Edge.source);
      todo = todo.concat(qs);
    }
    return paths.values();
  }
```

program

+

abstract machine

$$\begin{aligned}
\mathbf{ev}([\![s\,.\,x]\!], \hat{a}, \hat{\sigma}, \hat{\phi} : \hat{\kappa}) &\rightsquigarrow_{\Xi} \mathbf{ko}(\hat{\phi}, \hat{d}, \hat{\sigma}, \hat{\kappa}) \\
\text{where } (\hat{a}', \Xi') &\in \widehat{evalSimple}(s, \hat{a}, \hat{\sigma}) \\
(\hat{d}, \Xi'') &\in \widehat{lookupProp}(x, \hat{a}', \hat{\sigma}) \\
\Xi &= \Xi' \cup \Xi''
\end{aligned}$$



state graph

graph queries

(stack properties, trace properties)

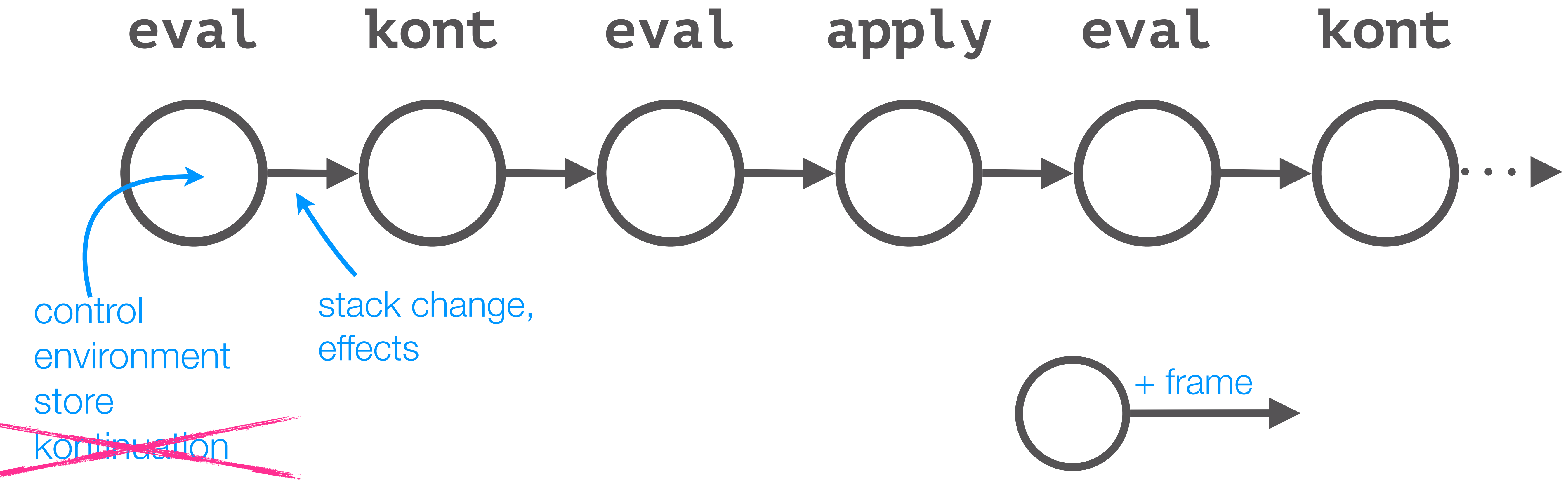
→ liveness analysis

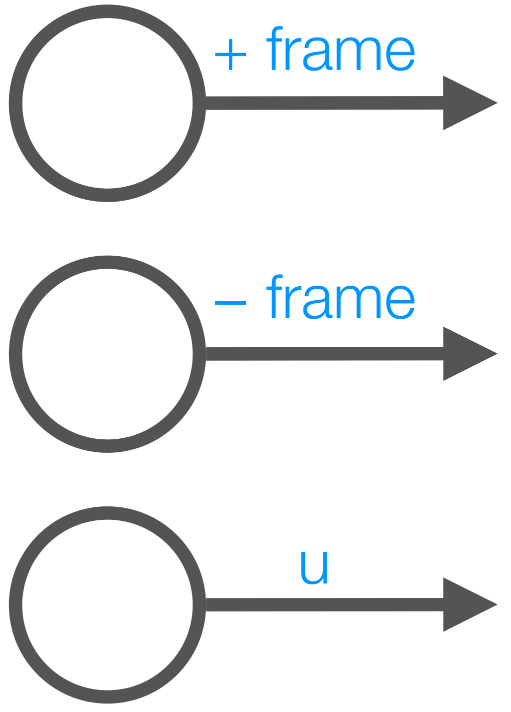
→ purity analysis

→ dependence analysis

→ taint analysis

...

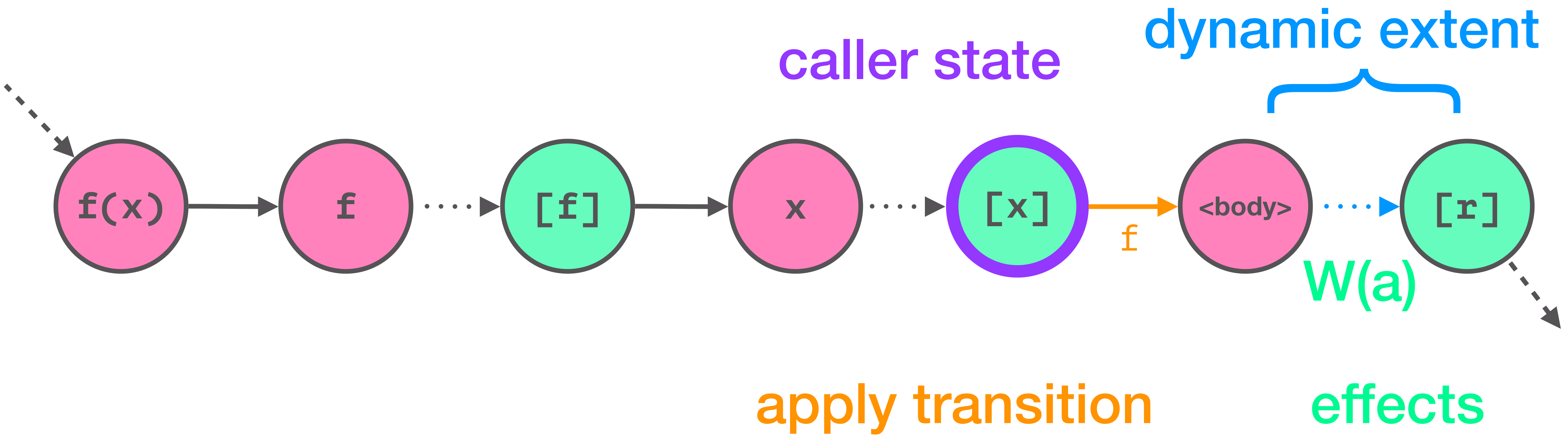




+ pushdown driver

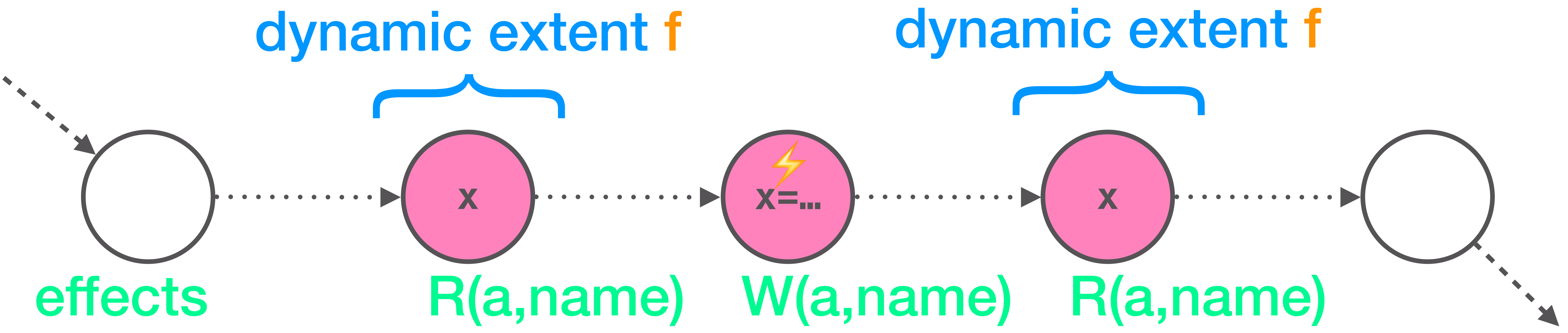
Pushdown control-flow analysis of higher-order programs, Earl et al., Scheme2010 “no observable side-effects”

“not dependent on side-effects”



“no observable side-effects”

“not dependent on side-effects”



Purity is a
useful property

Detecting purity is
difficult in JavaScript

JIPDA is a static analysis
capable of detecting purity

Purity Analysis for JavaScript

https://github.com/jensnicolay/jipda

Jens Nicolay • Quentin Stievenart
Carlos Noguera • Coen De Roover
Wolfgang De Meuter

Software Languages Lab
Vrije Universiteit Brussel