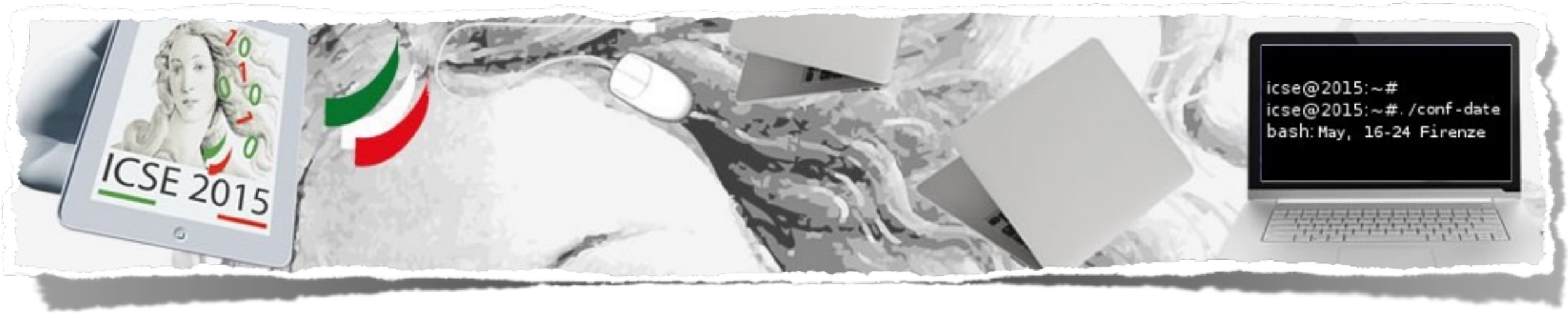


Regular Property Guided Dynamic Symbolic Execution

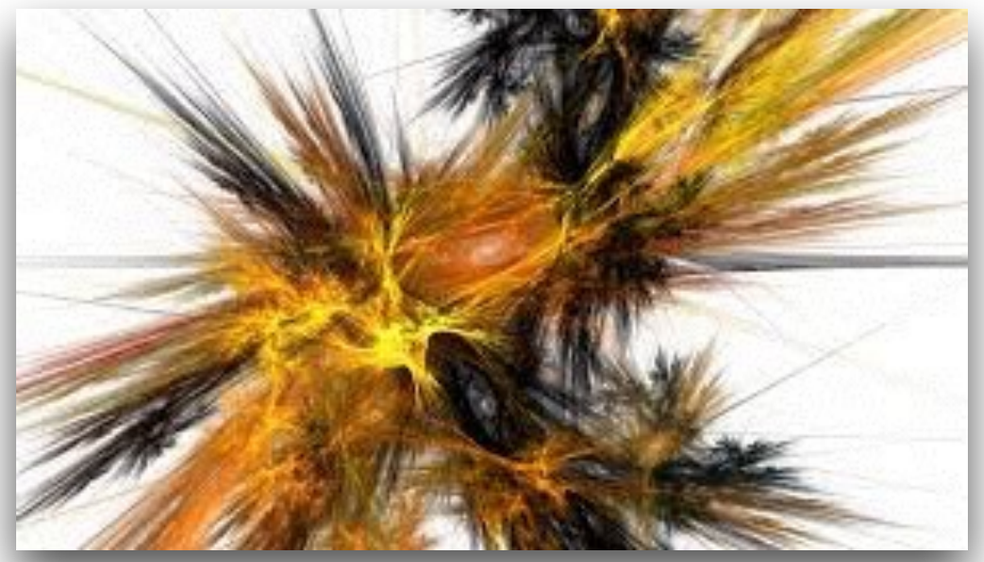
Zhenbang Chen
(zbchen@nudt.edu.cn)
joint work with Yufeng Zhang, Ji Wang, Wei Dong and Zhiming Liu

College of Computer, National University of Defense Technology, China
Centre for Software Engineering, Birmingham City University, UK

2015.05.22

Dynamic Symbolic Execution (DSE)

- Explore path spaces systematically
  - Test-case generation, bug-finding, bounded verification, …
- Path explosion problem



DSE needs guiding

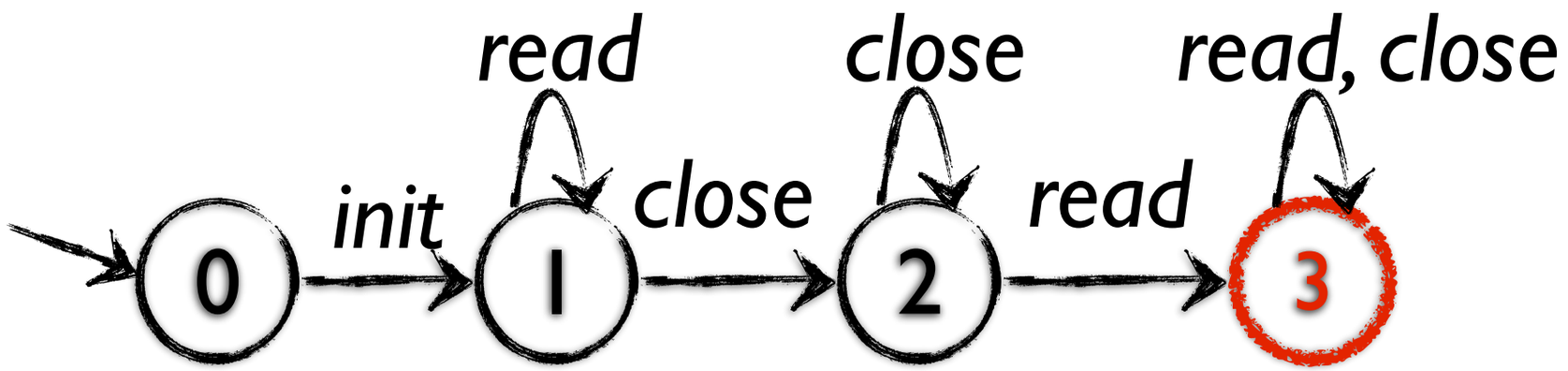
DSE needs guiding

Existing Work of Guiding Symbolic Execution

- Improving coverage
  - KLEE[OSDI'08], CREST[ASE'08], SGS[OOPSLA'13], CGS[FSE'14], …
- Reach program points
  - PEX[DSN'09], ESD[EuroSys'11], SDSE [SAS'11], BitBlaze[ISSTA'11], …
- Exploring the difference between programs
  - DiSE[PLDI'11], ZESTI [ICSE'12], KATCH[FSE'13], …

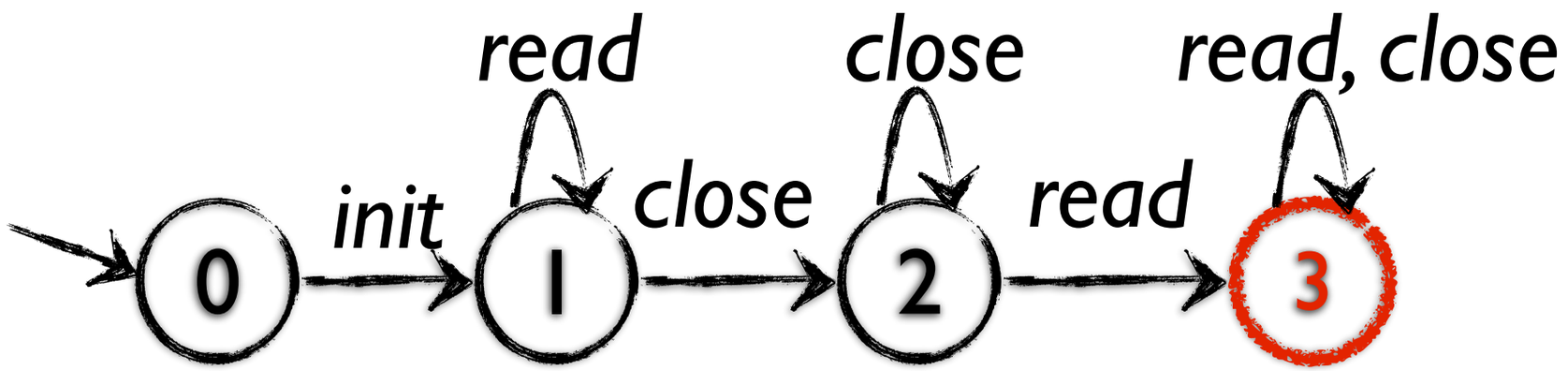
How about a Regular Property?

How about a Regular Property?



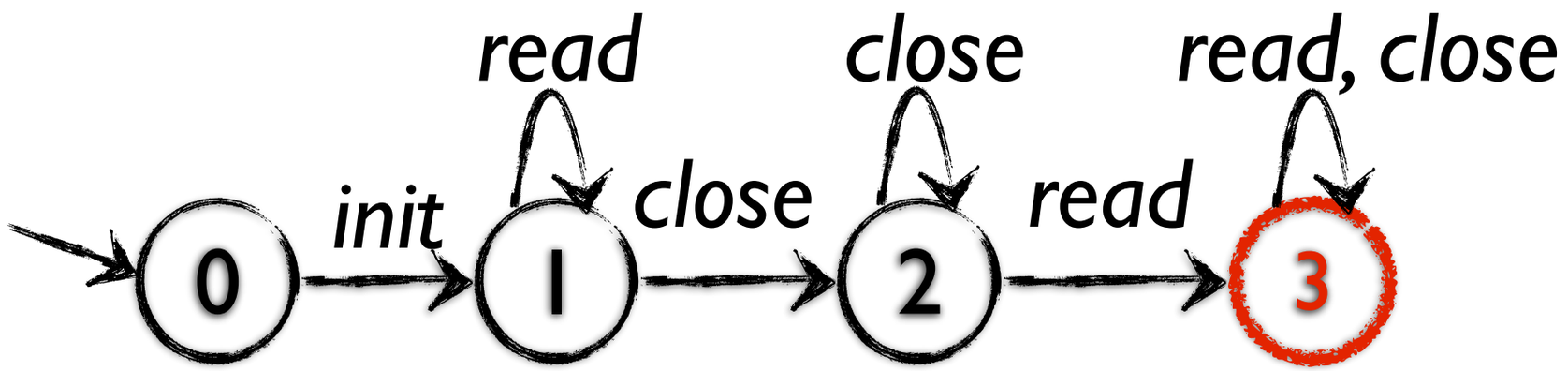
A bug property: a file is read after closed

How about a Regular Property?



A bug property: a file is read after closed

How about a Regular Property?

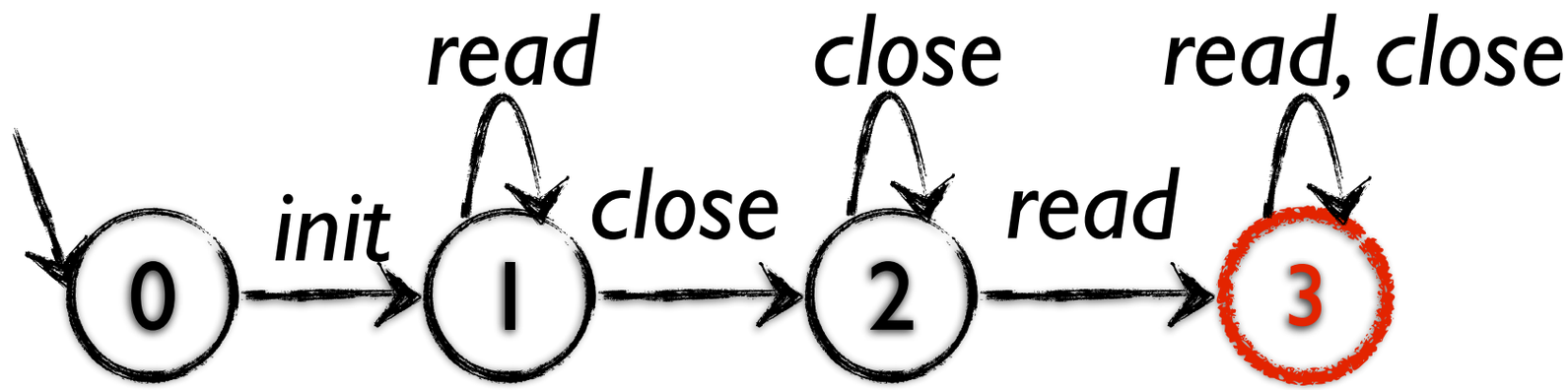


A bug property: a file is read after closed

How to guide DSE to find a program path satisfying P as soon as possible?

Observation and Insight

- Many irrelevant paths exist
- Even for relevant paths, only the ones with specific sequences can satisfy the regular property

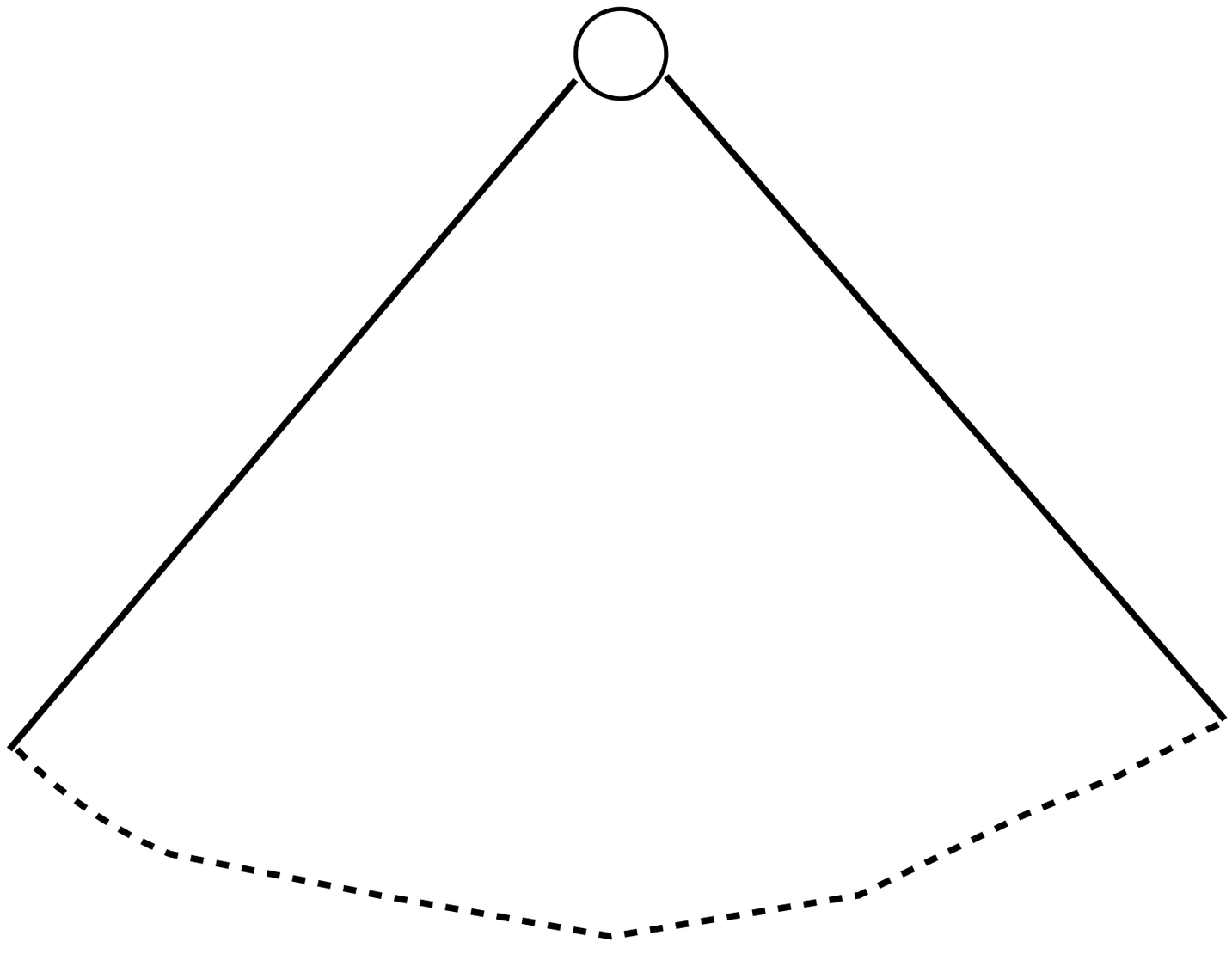


Observation and Insight

- Many irrelevant paths exist
- Even for relevant paths, only the ones with specific sequences can satisfy the regular property

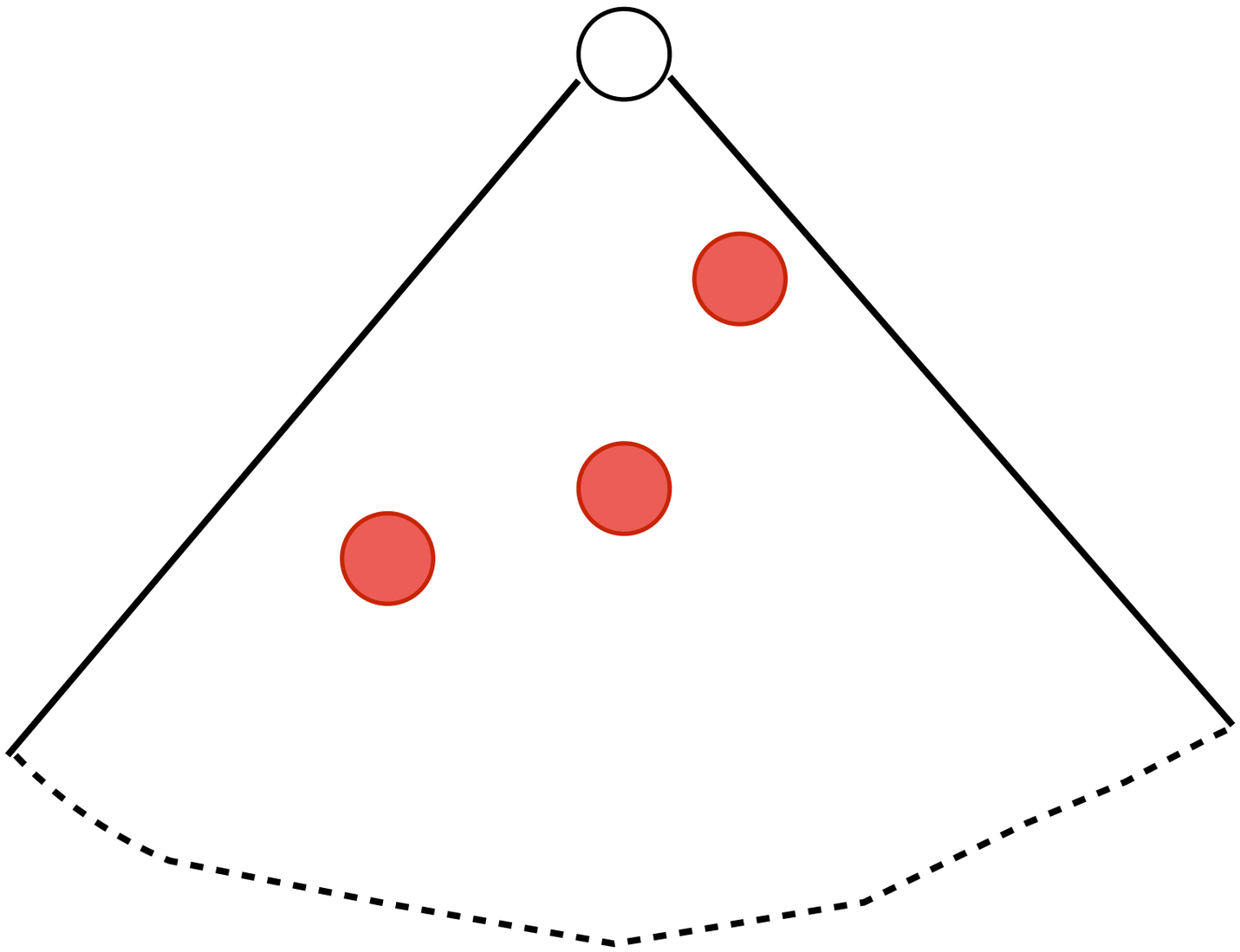
Evaluate the possibility of a branch to generate the paths satisfying the property

Key Idea



Evaluate a branch based on its history and future behaviors

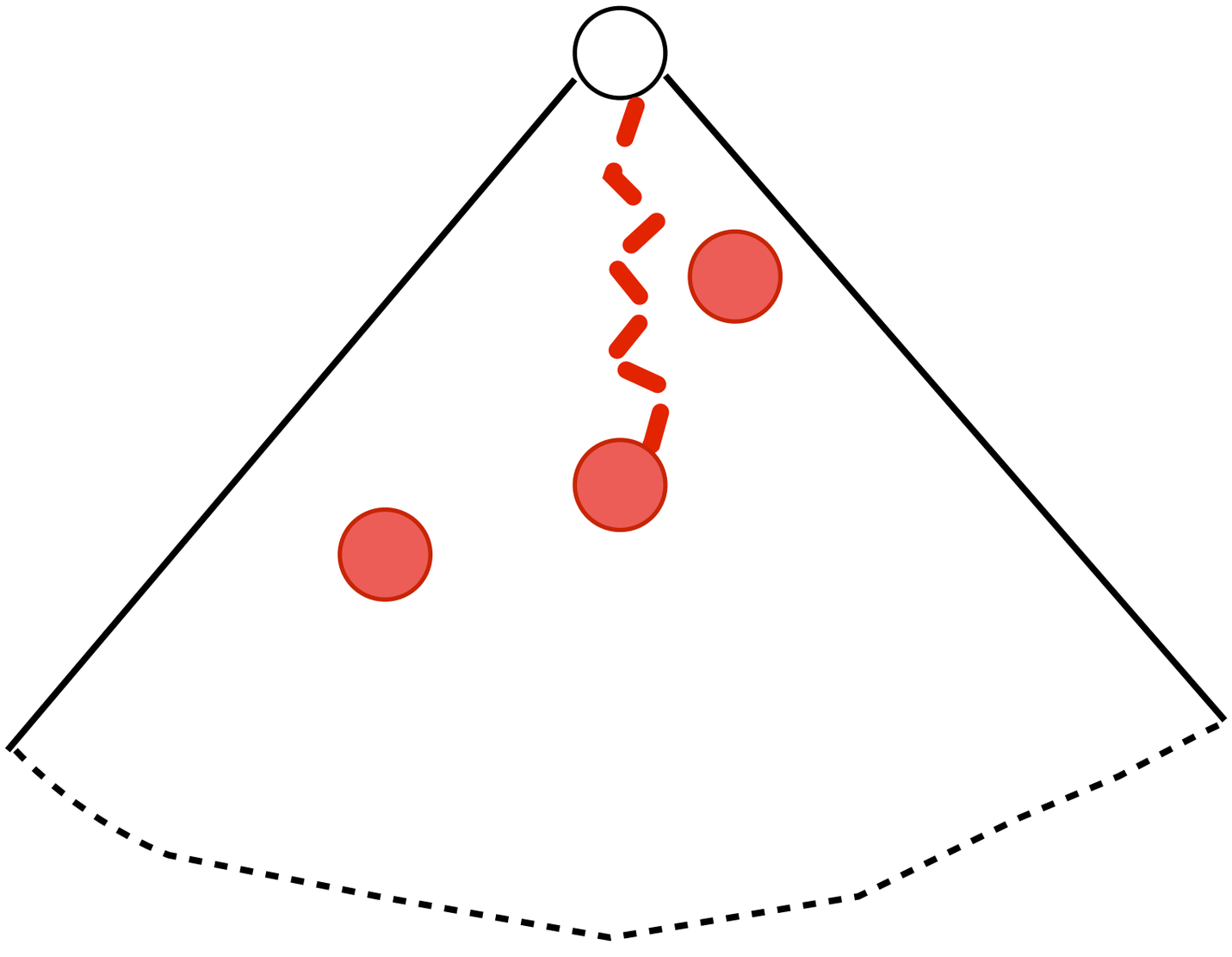
Key Idea



Evaluate a branch based on its history and future behaviors

Key Idea

history

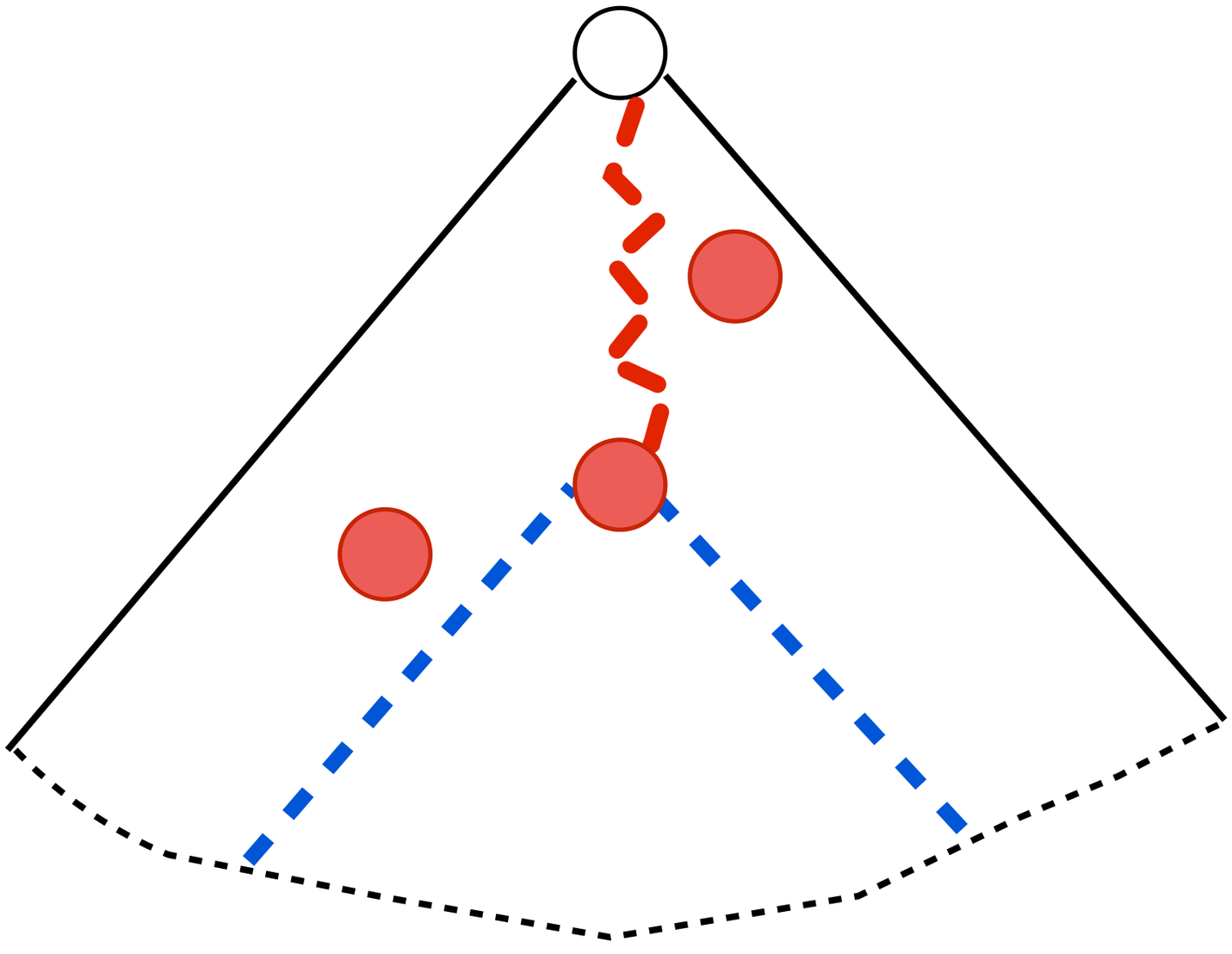


Evaluate a branch based on its history and future behaviors

Key Idea

history

future



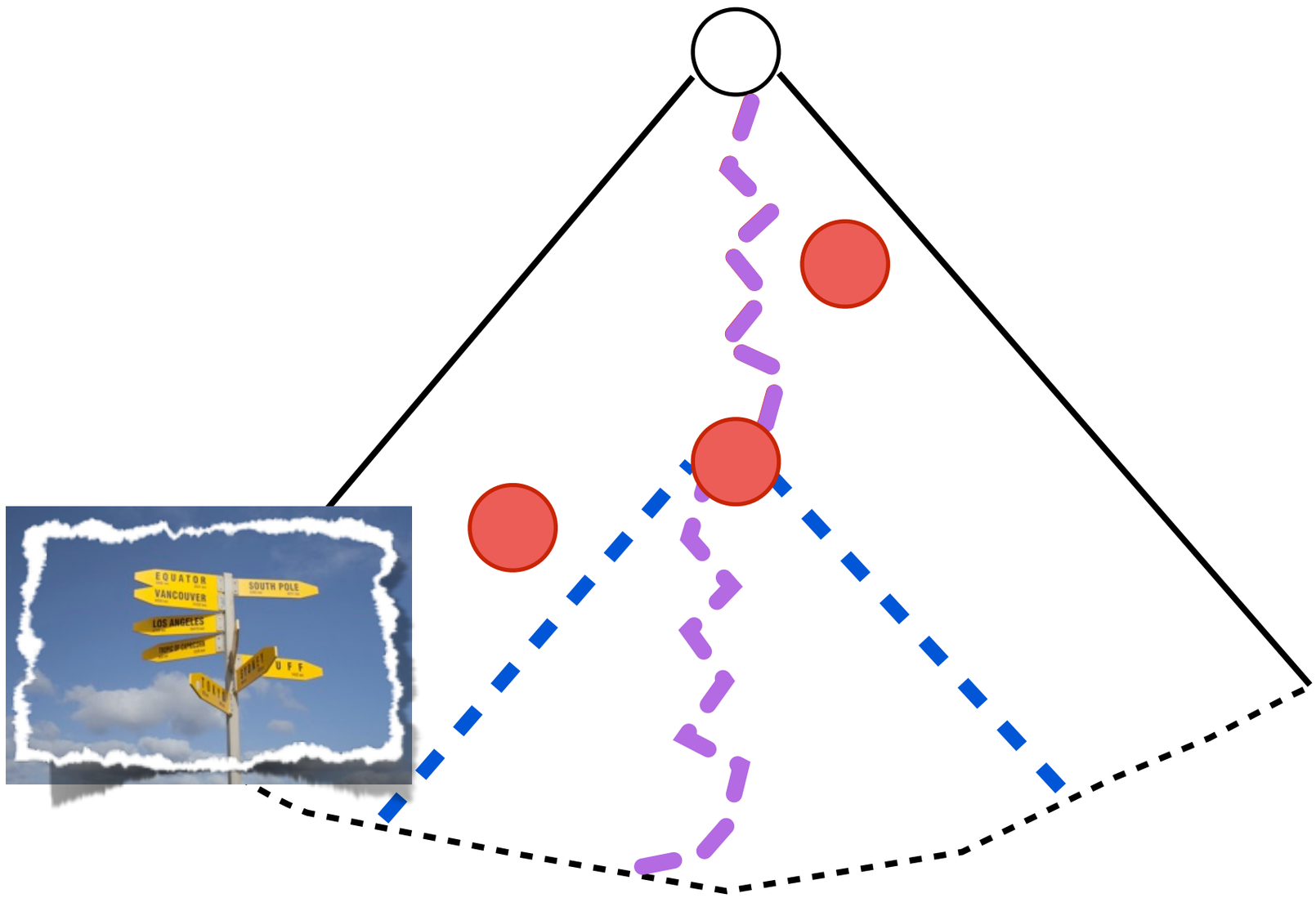
Evaluate a branch based on its history and future behaviors

Key Idea

history $\cap$ *future* $\neq \varnothing$

history

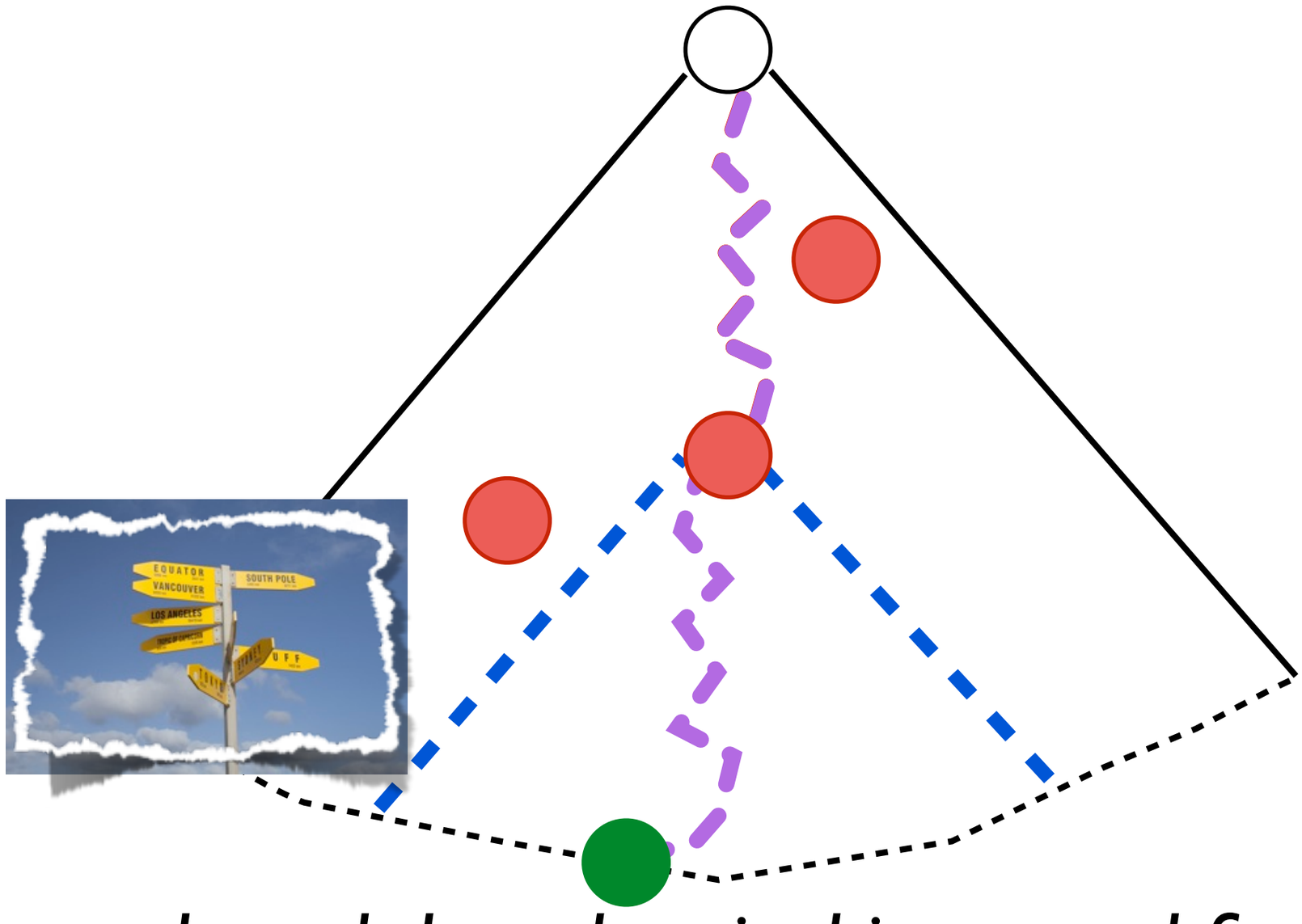
future



Evaluate a branch based on its history and future behaviors

Key Idea

history *history* ∩ *future* ≠ ∅ *future*

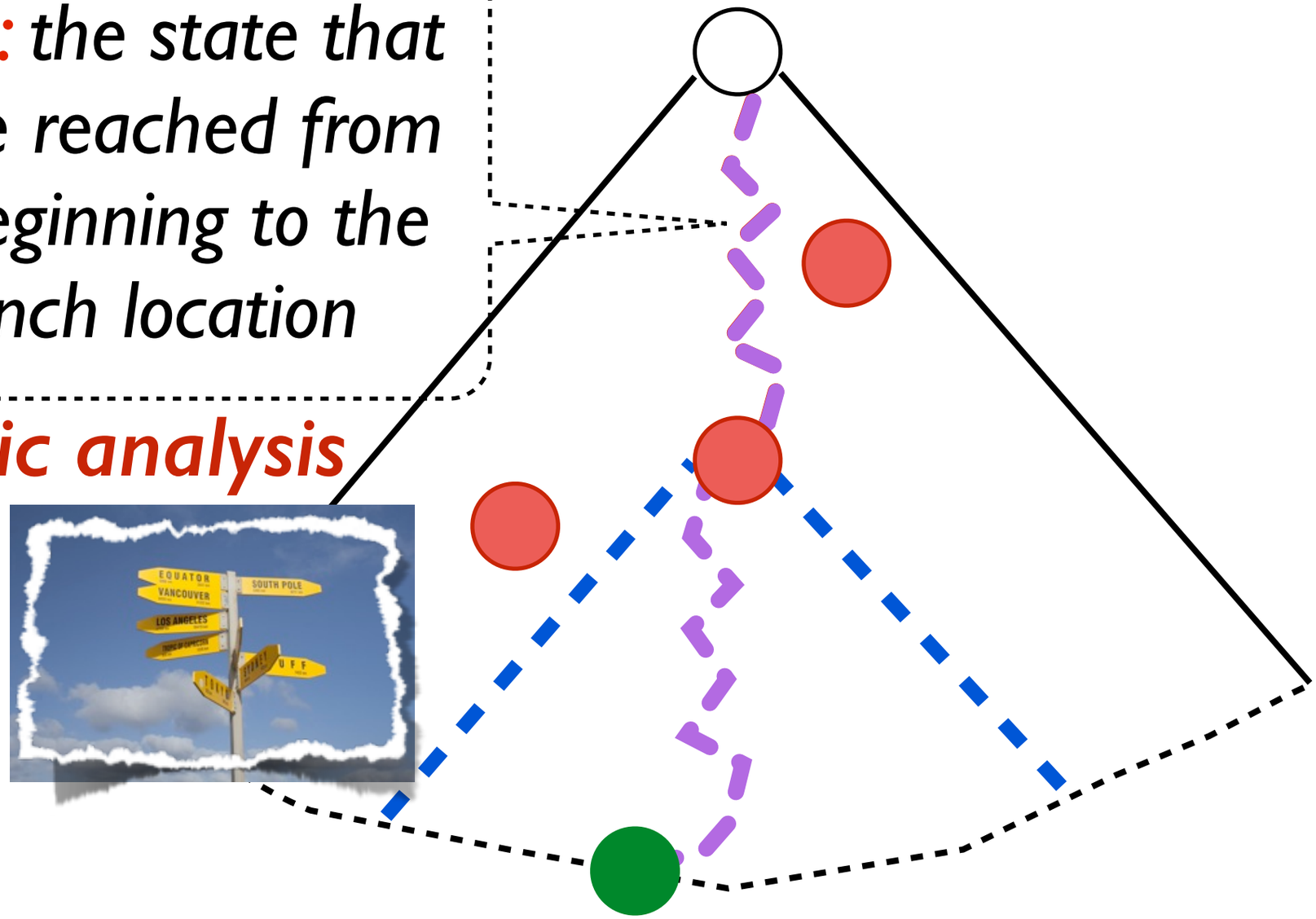


Evaluate a branch based on its history and future behaviors

Key Idea

history *history* ∩ *future* ≠ ∅ *future*

Preset: the state that can be reached from the beginning to the branch location



Dynamic analysis

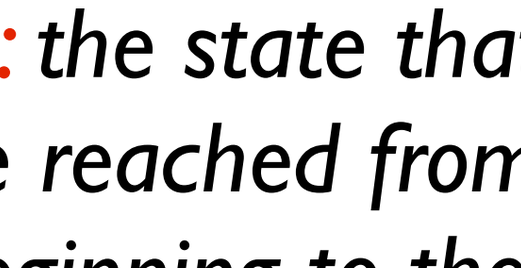
Evaluate a branch based on its history and future behaviors

Key Idea

history *history* ∩ *future* ≠ ∅ *future*

Preset: the state that can be reached from the beginning to the branch location

Postset: the states from which it can reach a final state after executing the rest program after the branch location



Dynamic analysis

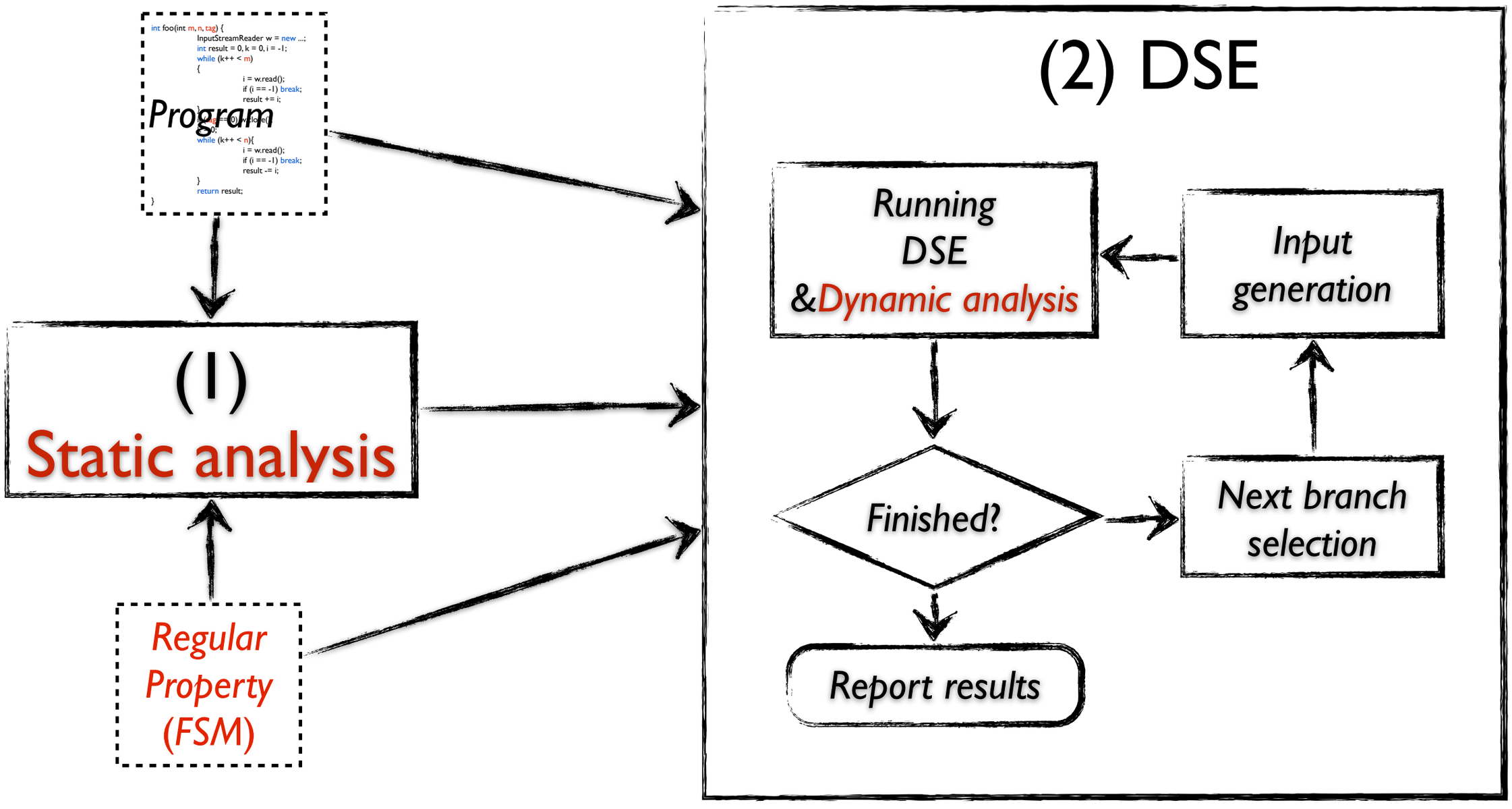
Static analysis

Evaluate a branch based on its history and future behaviors

Sneak Preview of Results

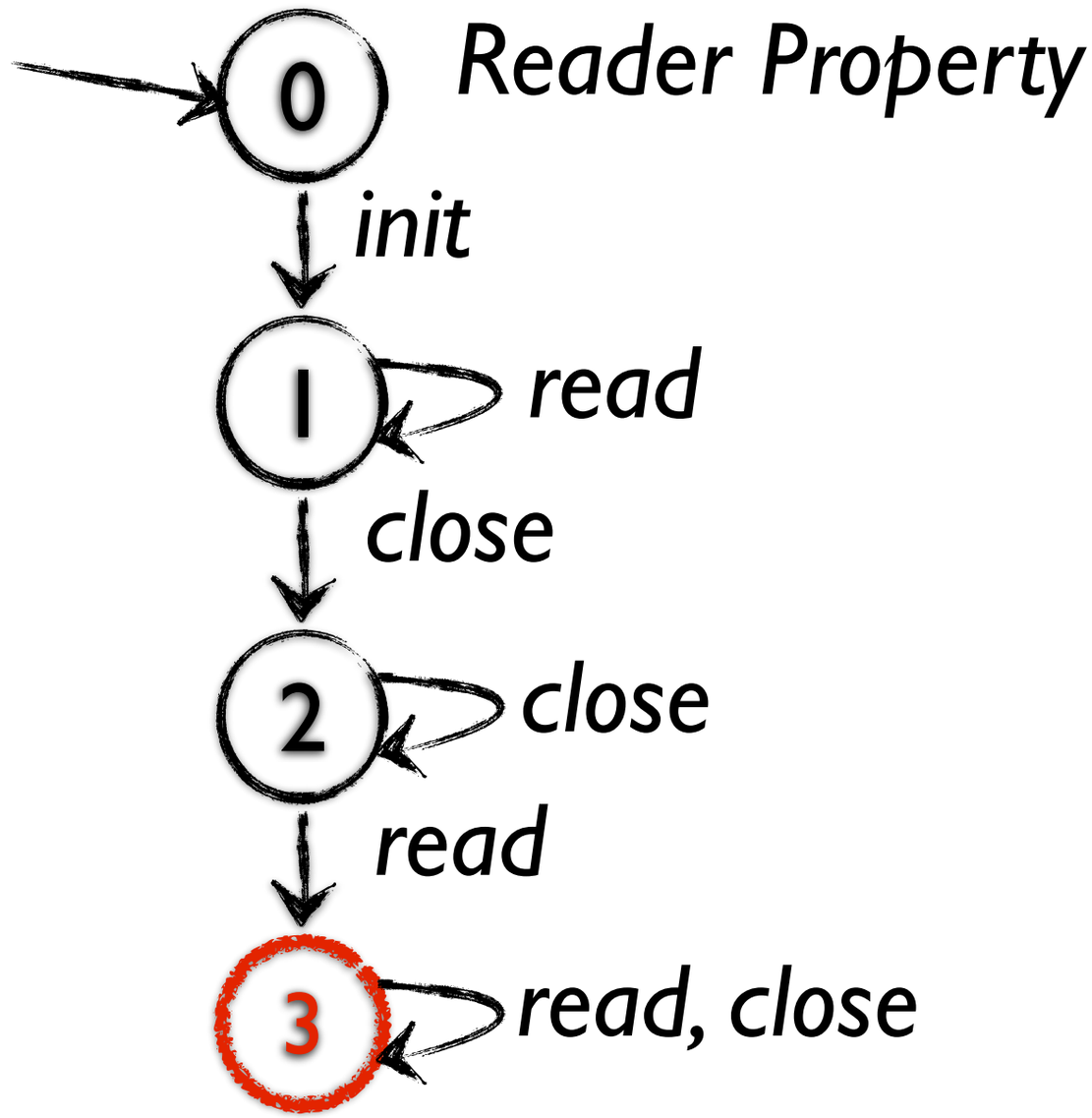
- For finding the first accepted path
  - >1880X speedup on iterations
  - >258X time speedup on the programs whose paths space is bigger than 100
- For 3 out of the 13 real world programs
  - Guided method succeeds in 1 hour
  - Pure method fails in 24 hours

Procedure


int foo(int m, n, tag) {
InputStreamReader w = new ...;
int result = 0, k = 0, i = -1;
while (k++ < m)
{
i = w.read();
if (i == -1) break;
result += i;
}
if (tag == 0) throw();
while (k++ < n){
i = w.read();
if (i == -1) break;
result -= i;
}
return result;
}
Program
(1)
Static analysis
Regular
Property
(FSM)
(2) DSE
Running
DSE
&Dynamic analysis
Input
generation
Finished?
Next branch
selection
Report results

An Example

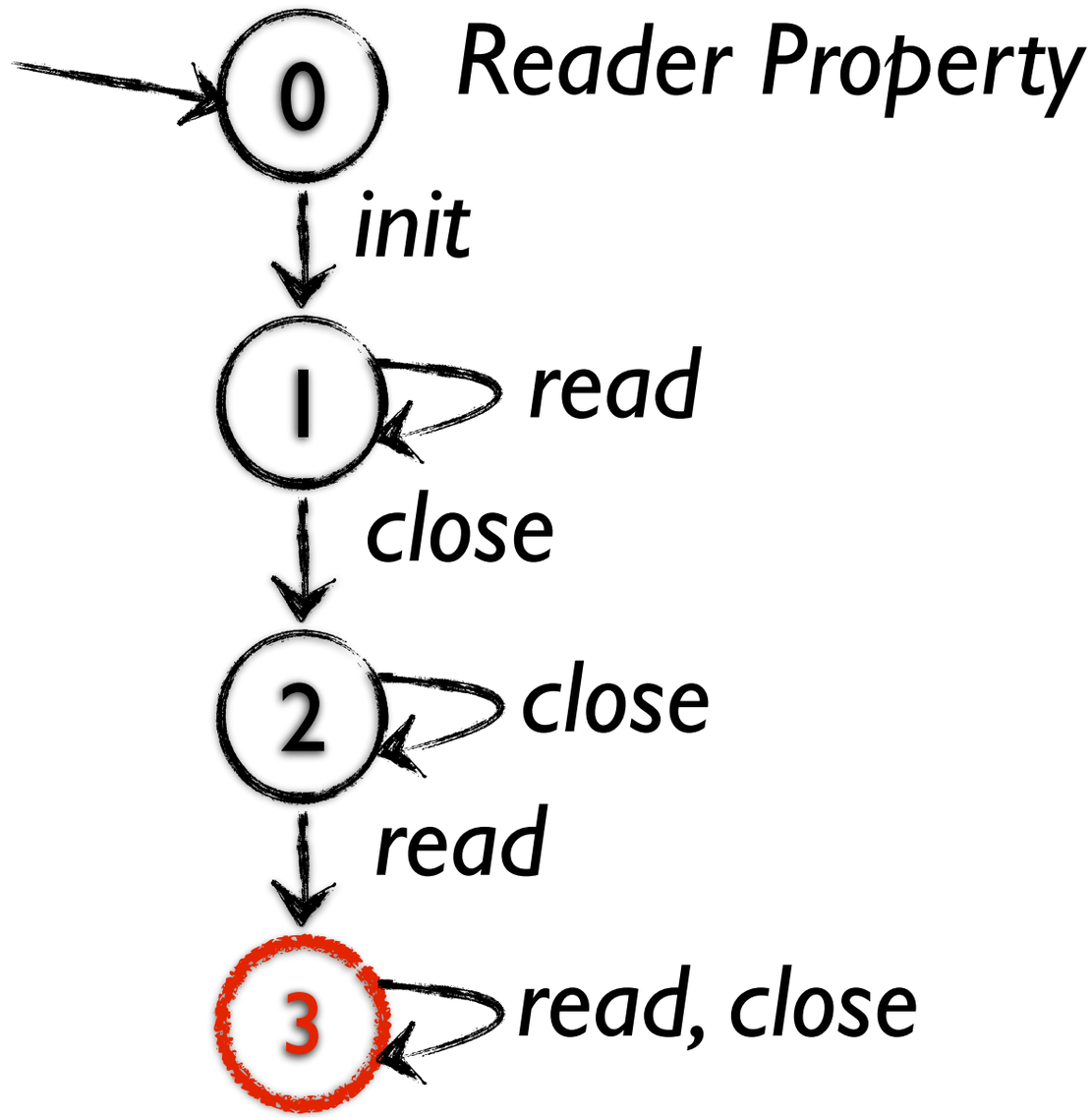
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```



A reader is read after closed

An Example

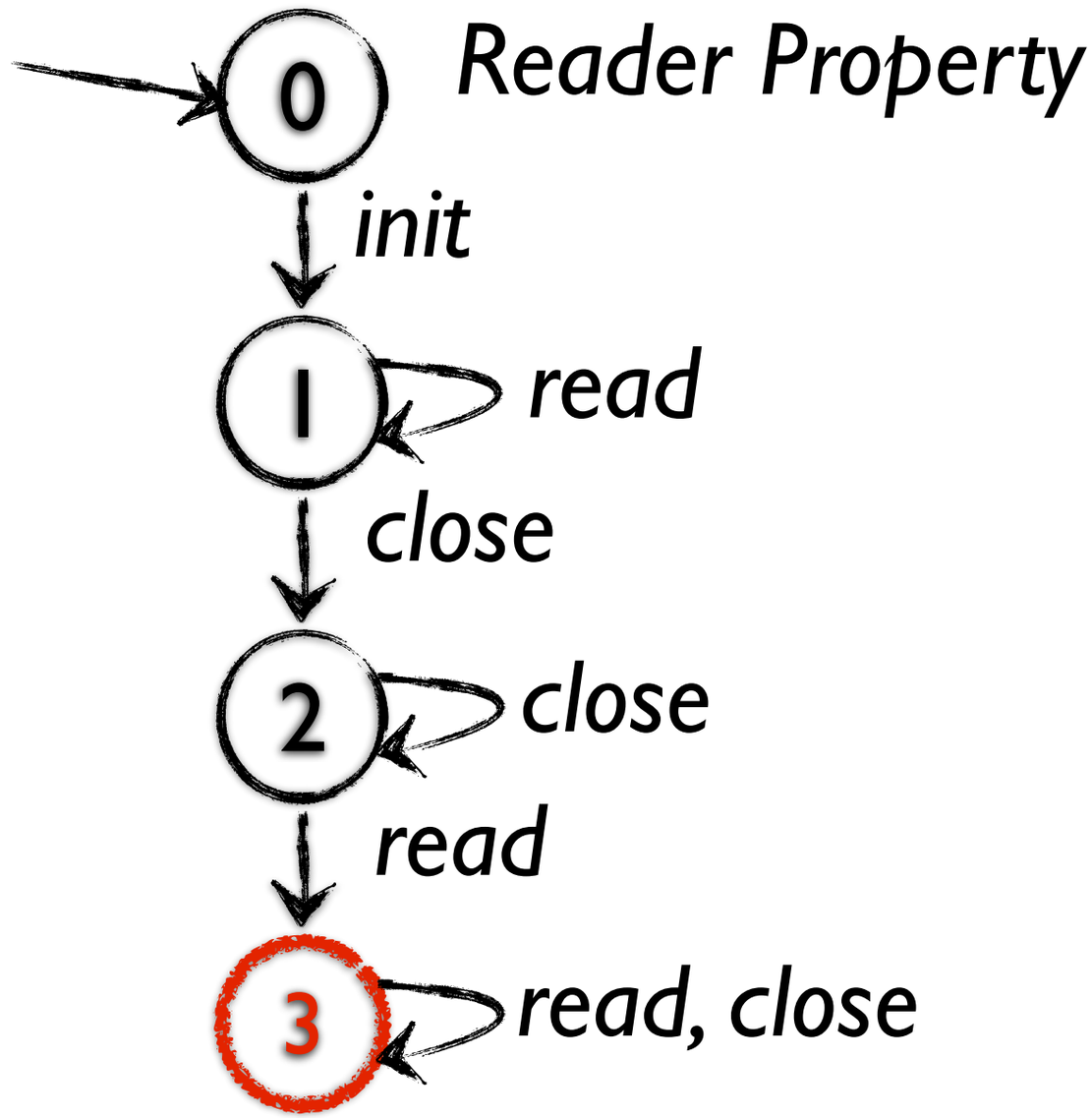
```
int foo(int m, n, tag) {
  InputStreamReader w = new ...;
  int result = 0, k = 0, i = -1;
  while (k++ < m)
  {
     i = w.read();
     if (i == -1) break;
     result += i;
  }
  if (tag == 0) w.close();
  k = 0;
  while (k++ < n){
     i = w.read();
     if (i == -1) break;
     result -= i;
  }
  return result;
}
```



A reader is read after closed

An Example

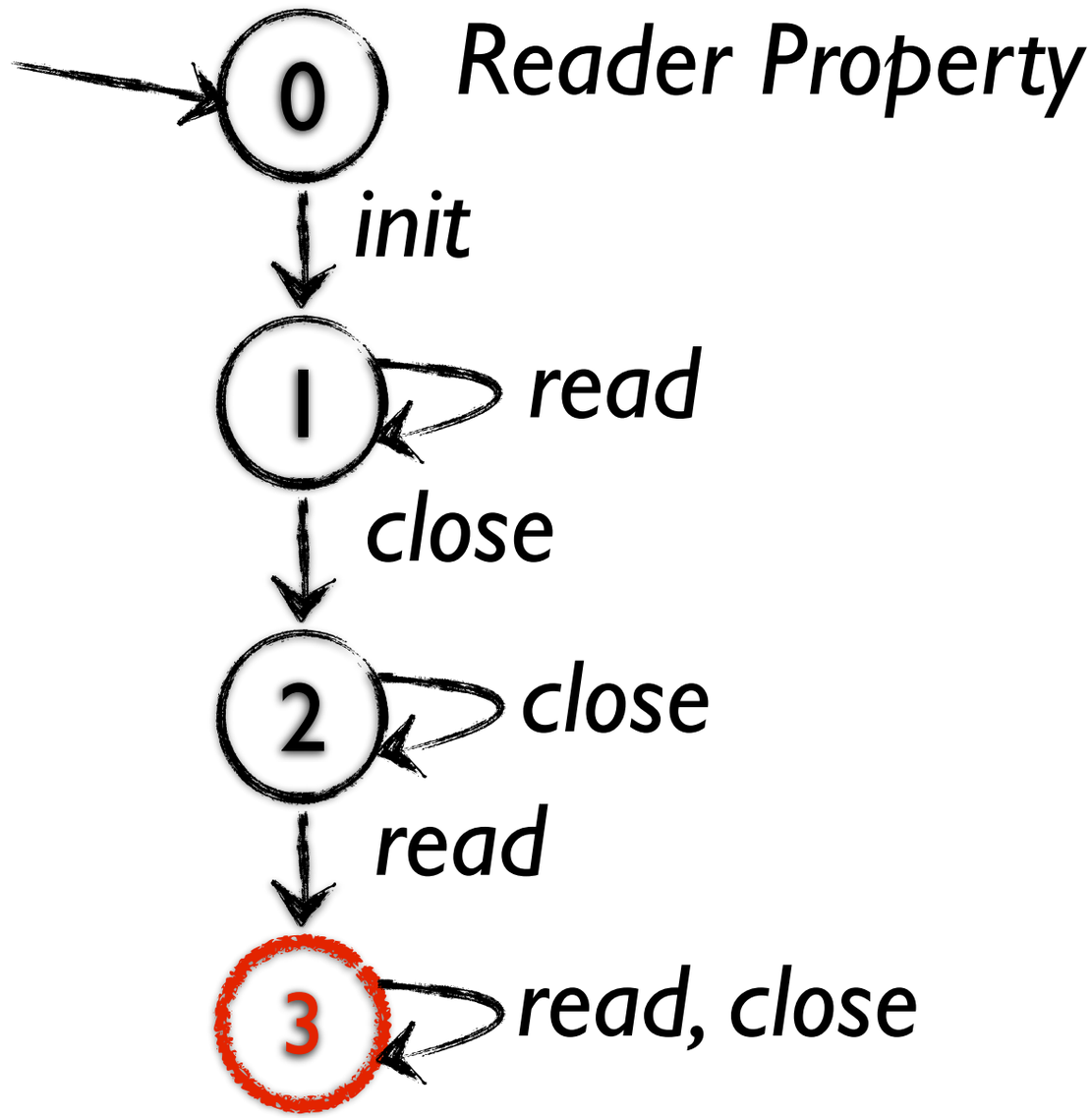
```
int foo(int m, n, tag) {
  InputStreamReader w = new ...;
  int result = 0, k = 0, i = -1;
  while (k++ < m)
  {
     i = w.read();
     if (i == -1) break;
     result += i;
  }
  if (tag == 0) w.close();
  k = 0;
  while (k++ < n){
     i = w.read();
     if (i == -1) break;
     result -= i;
  }
  return result;
}
```



A reader is read after closed

An Example

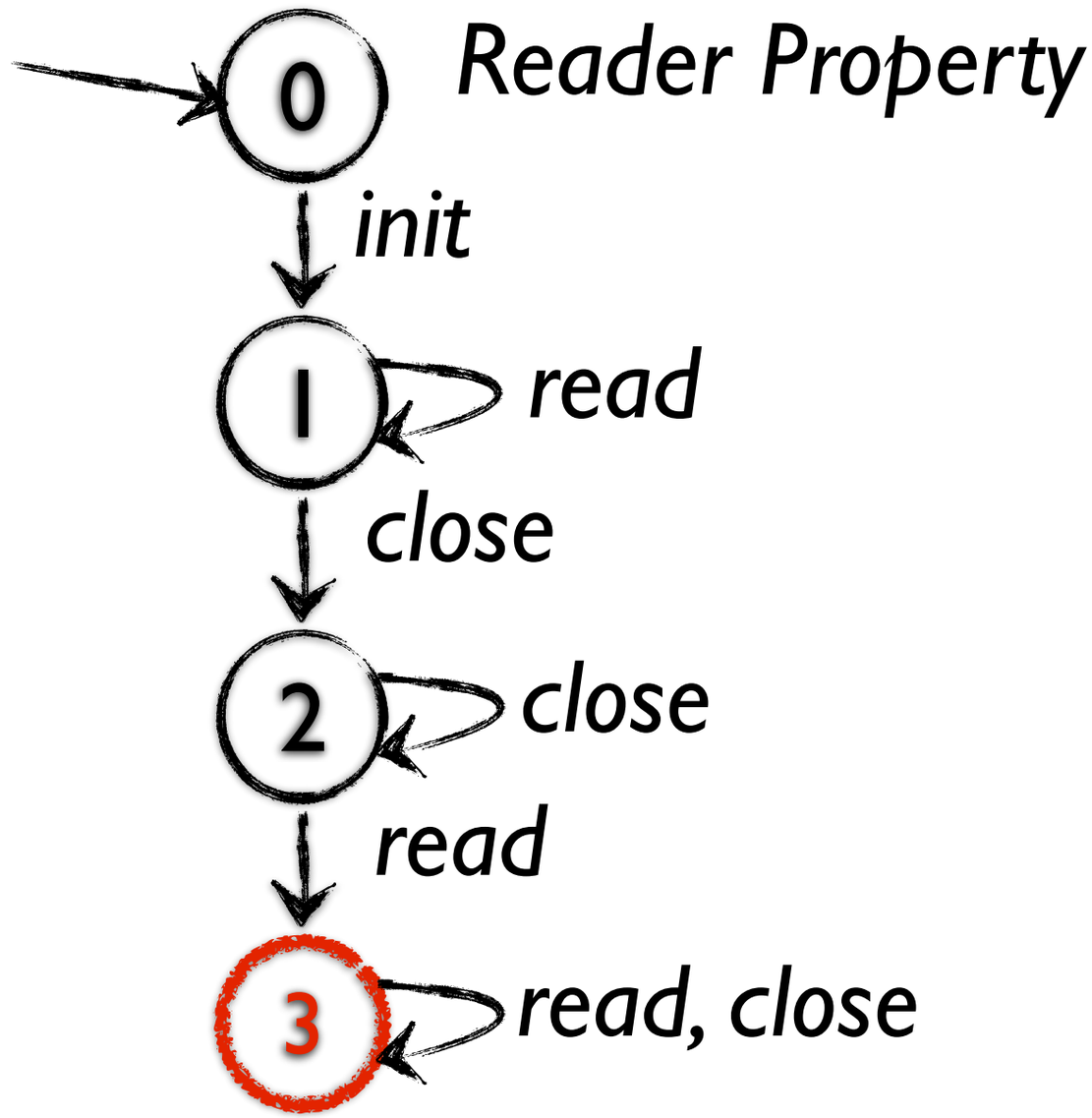
```
int foo(int m, n, tag) {
  InputStreamReader w = new ...;
  int result = 0, k = 0, i = -1;
  while (k++ < m)
  {
     i = w.read();
     if (i == -1) break;
     result += i;
  }
  if (tag == 0) w.close();
  k = 0;
  while (k++ < n){
     i = w.read();
     if (i == -1) break;
     result -= i;
  }
  return result;
}
```



A reader is read after closed

An Example

```
int foo(int m, n, tag) {
  InputStreamReader w = new ...;
  int result = 0, k = 0, i = -1;
  while (k++ < m)
  {
     i = w.read();
     if (i == -1) break;
     result += i;
  }
  if (tag == 0) w.close();
  k = 0;
  while (k++ < n){
     i = w.read();
     if (i == -1) break;
     result -= i;
  }
  return result;
}
```



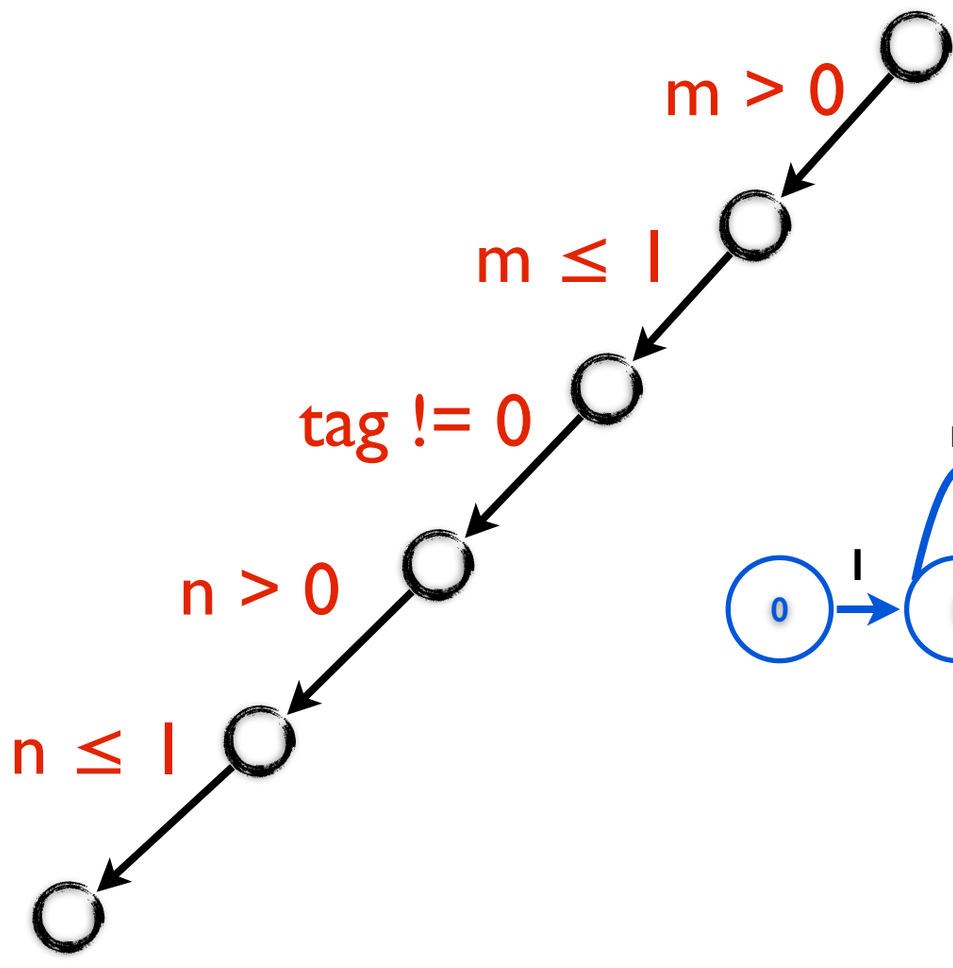
A reader is read after closed

Pure DSE

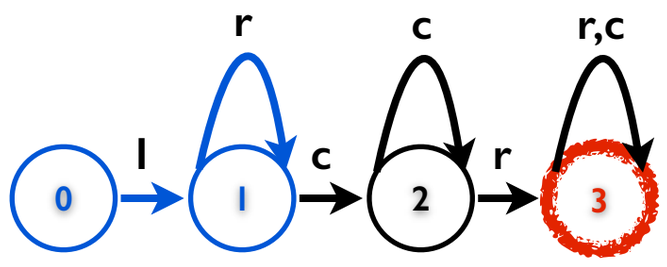
Pure DSE-1st Iteration

```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

(m=1, n=1, tag=1)



DFS

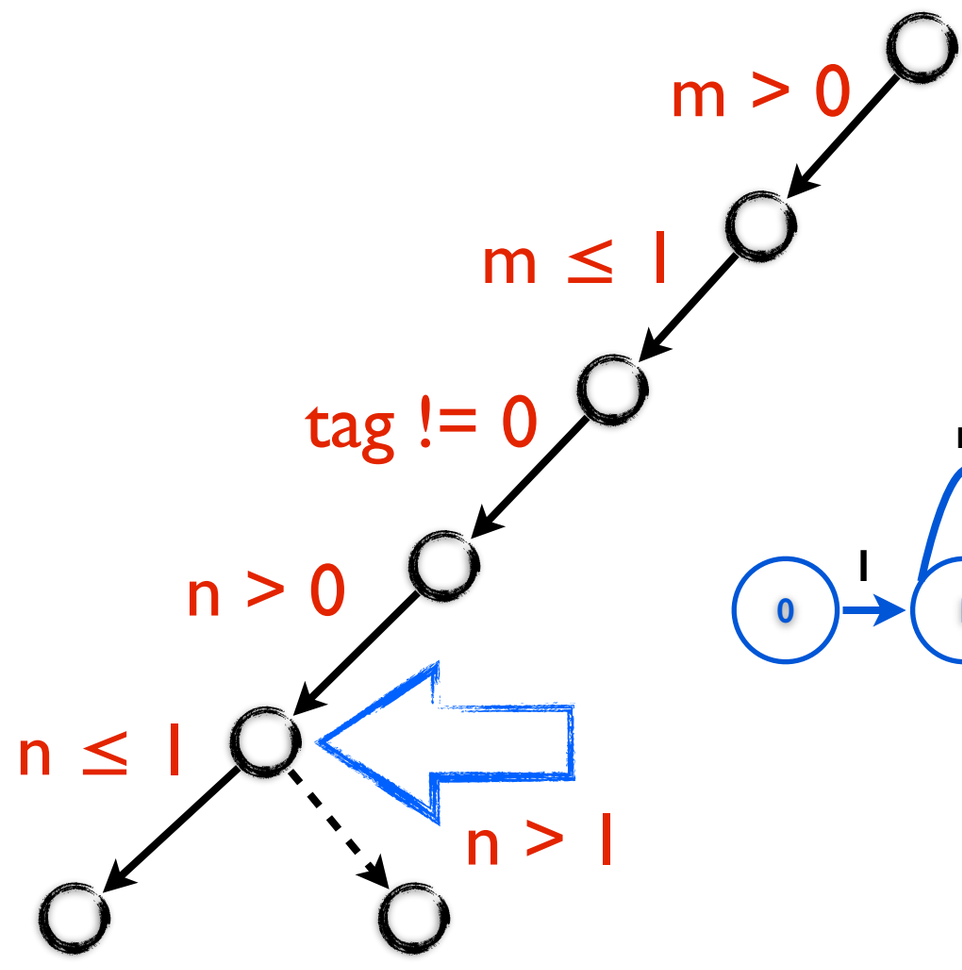


m > 0 ∧ m ≤ 1 ∧ tag != 0 ∧ n > 0 ∧ n ≤ 1

Pure DSE-1st Iteration

```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

(m=1, n=1, tag=1)



DFS

m > 0 ∧ m ≤ 1 ∧ tag != 0 ∧ n > 0 ∧ n ≤ 1

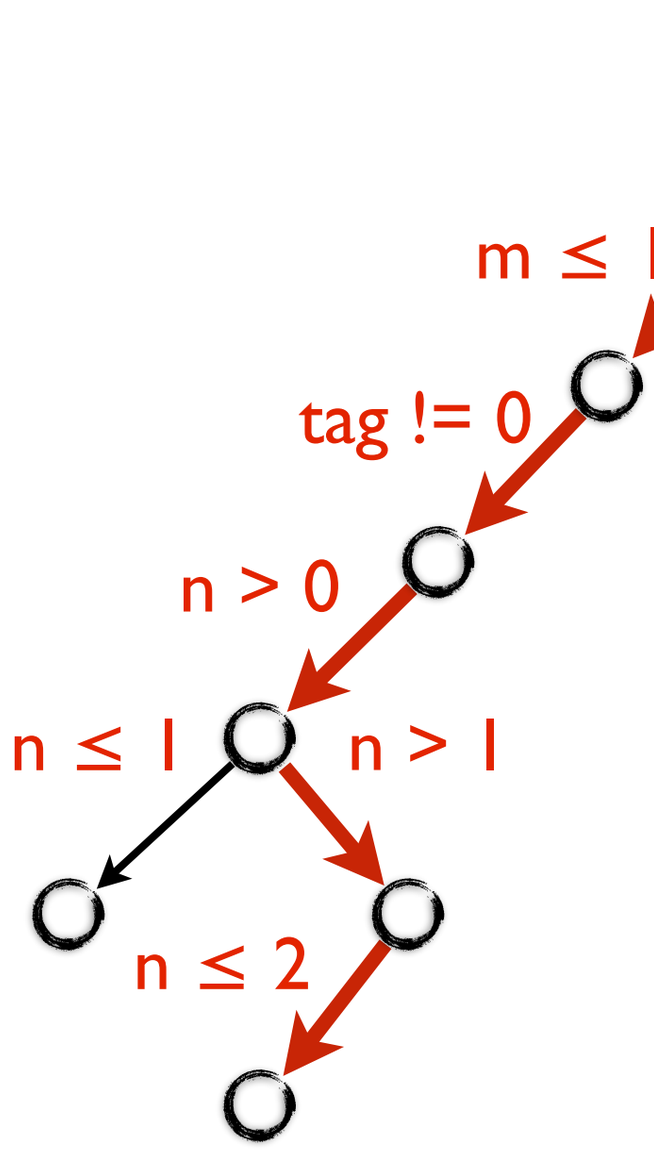
m > 0 ∧ m ≤ 1 ∧ tag != 0 ∧ n > 0 ∧ n > 1

(m=1, n=2, tag=1)

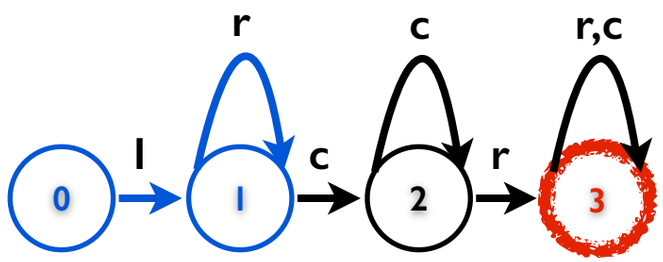
Pure DSE-2nd Iteration

```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

(m=1, n=2, tag=1)



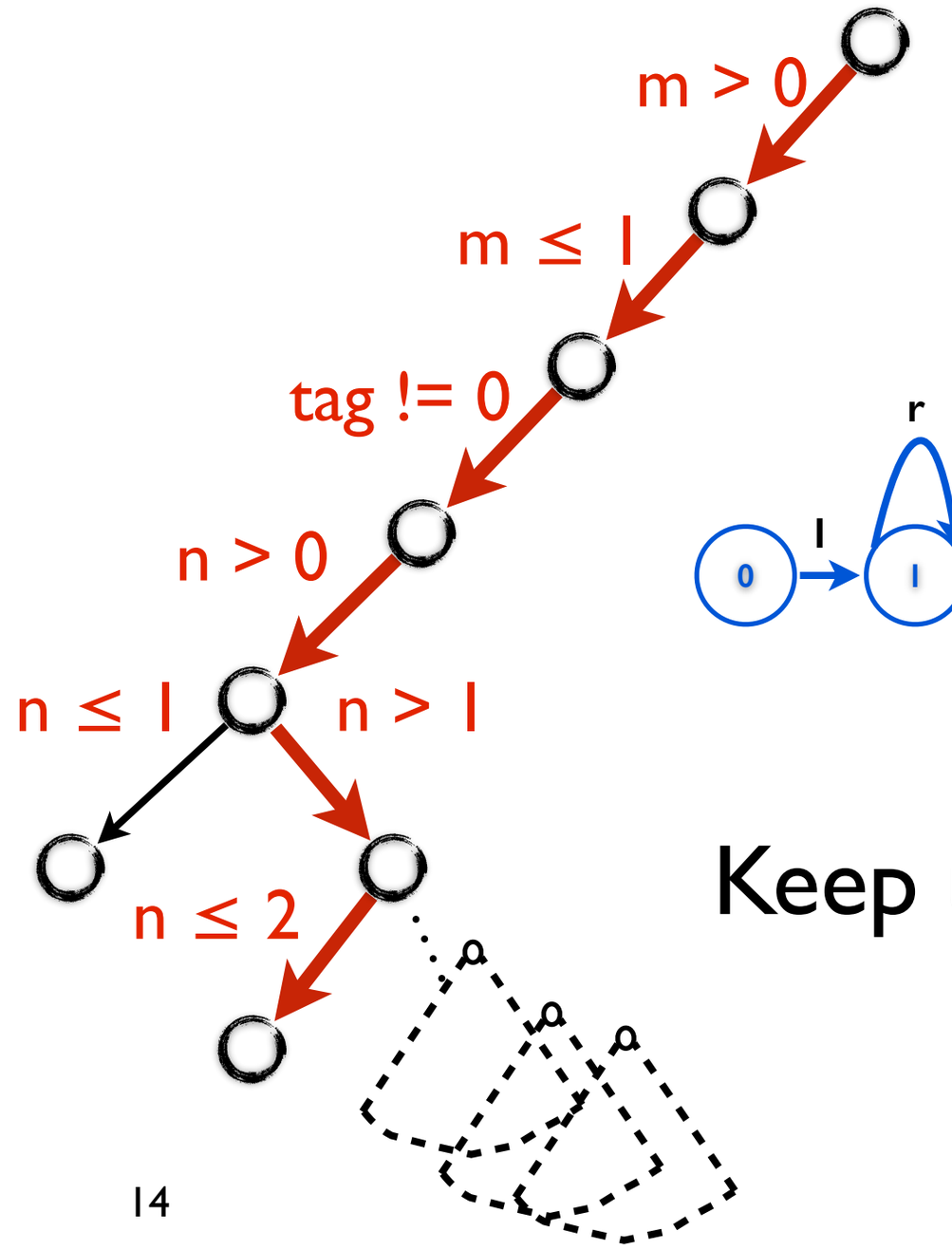
DFS



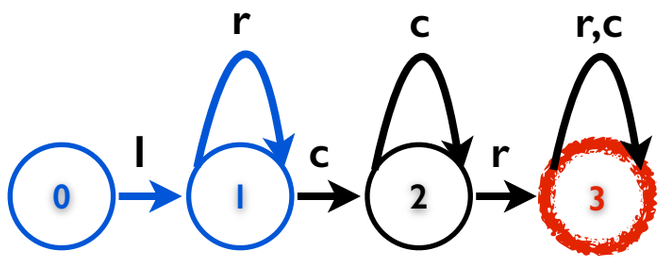
Pure DSE-2nd Iteration

```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

(m=1, n=2, tag=1)



DFS

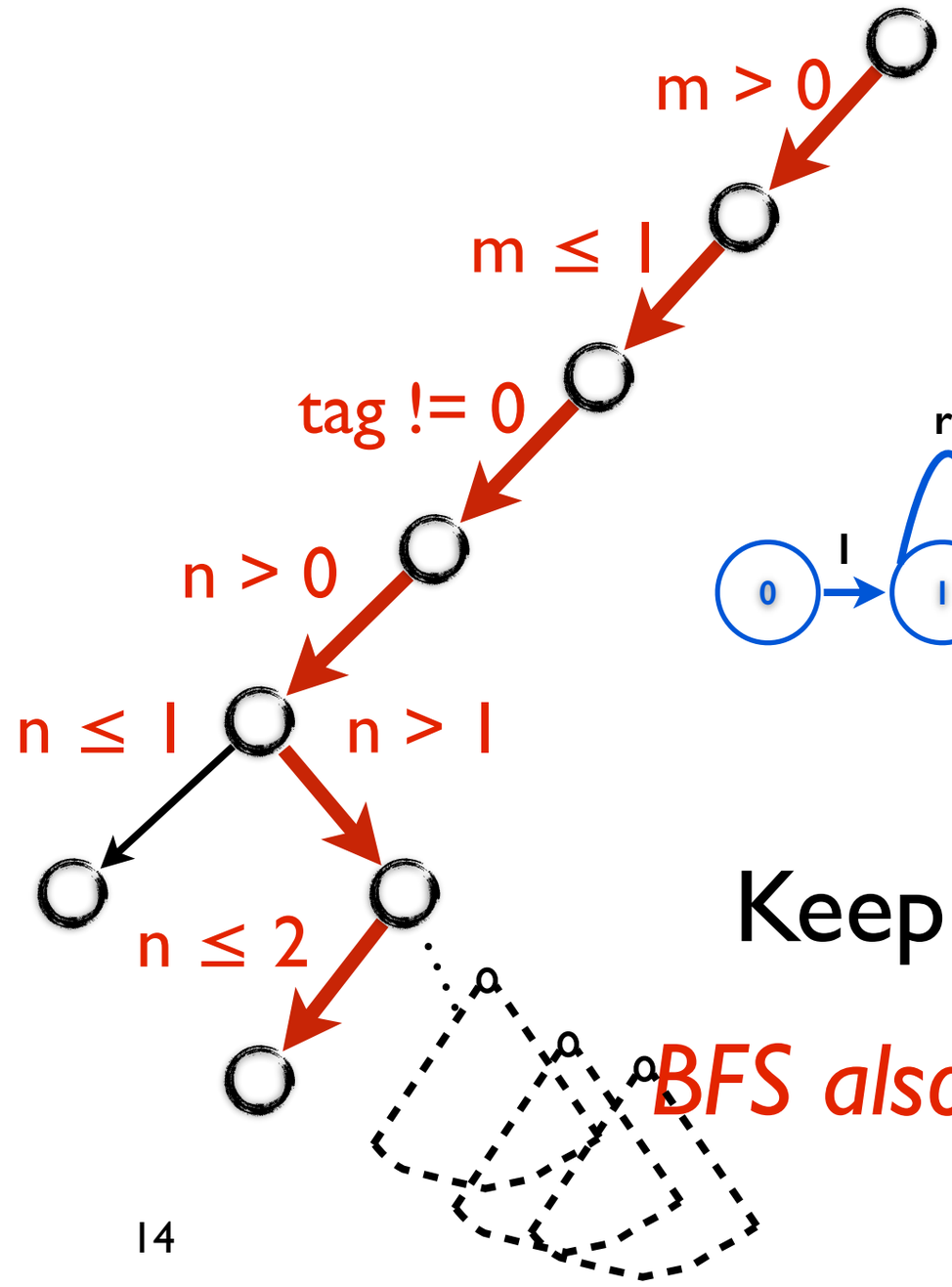


Keep unfolding

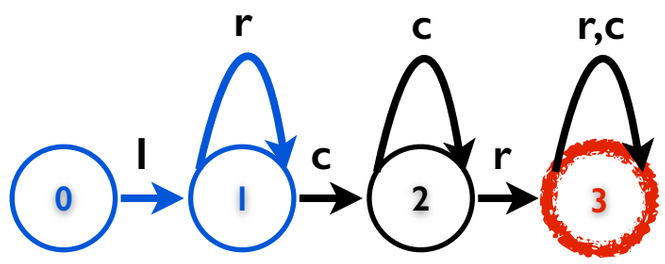
Pure DSE-2nd Iteration

```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

(m=1, n=2, tag=1)



DFS

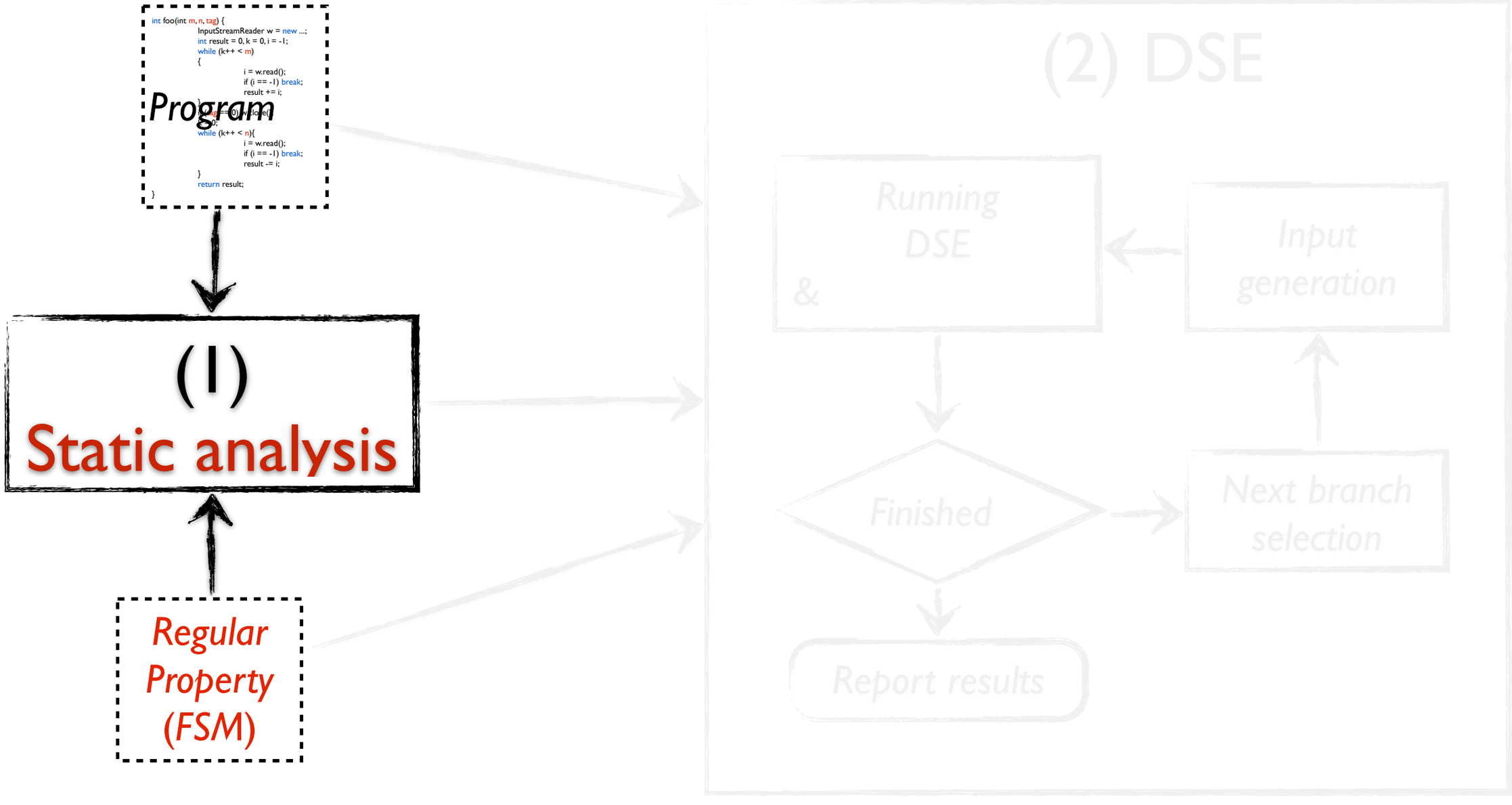


Keep unfolding

BFS also gets stuck

Guided DSE

Guided DSE Procedure



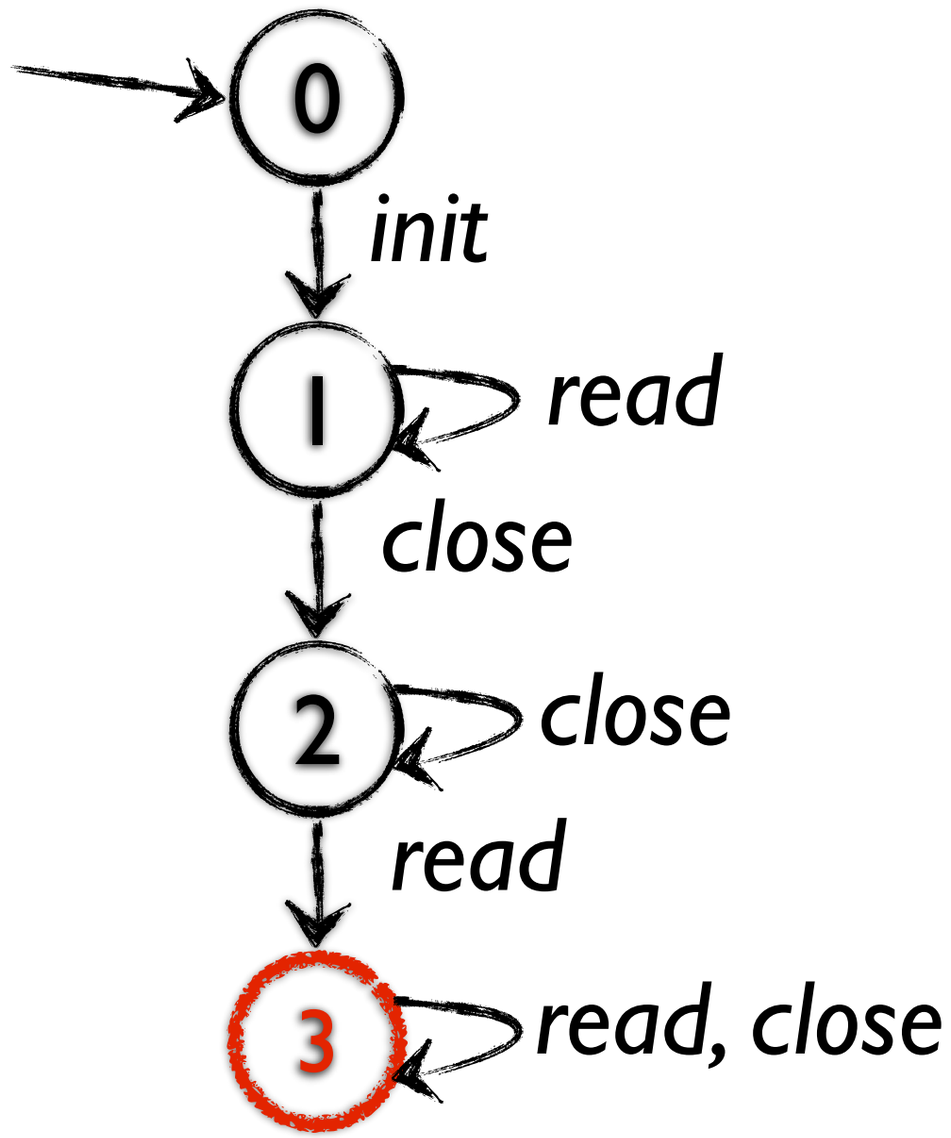
Postset Calculation

```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0)
       w.close();
   k = 0;
   while (k++ < n)
   {
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

Postset

Calculation

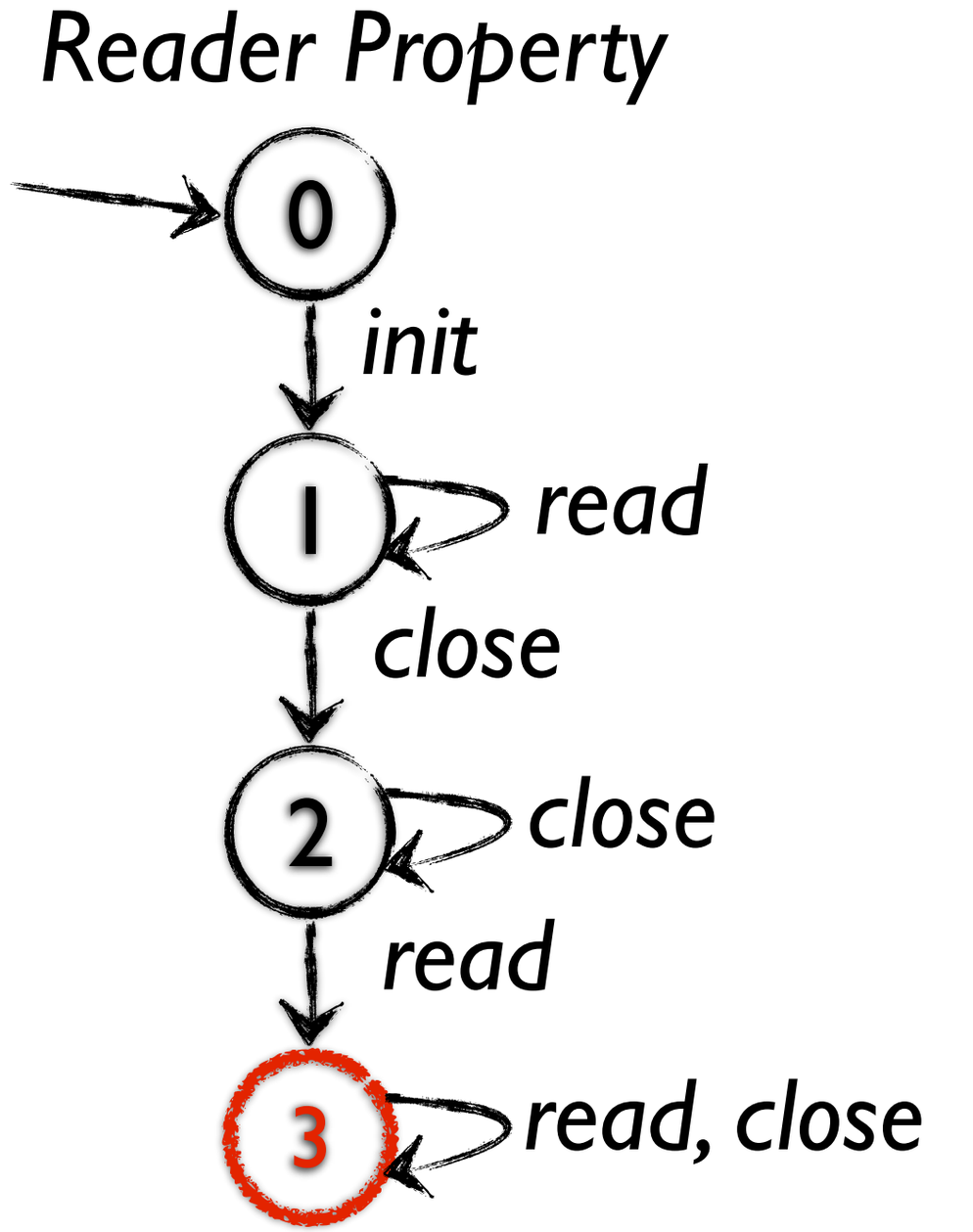
Reader Property



Backward data flow analysis [Clara, ICSE'10]

$O(|E| \times |D|^3)$

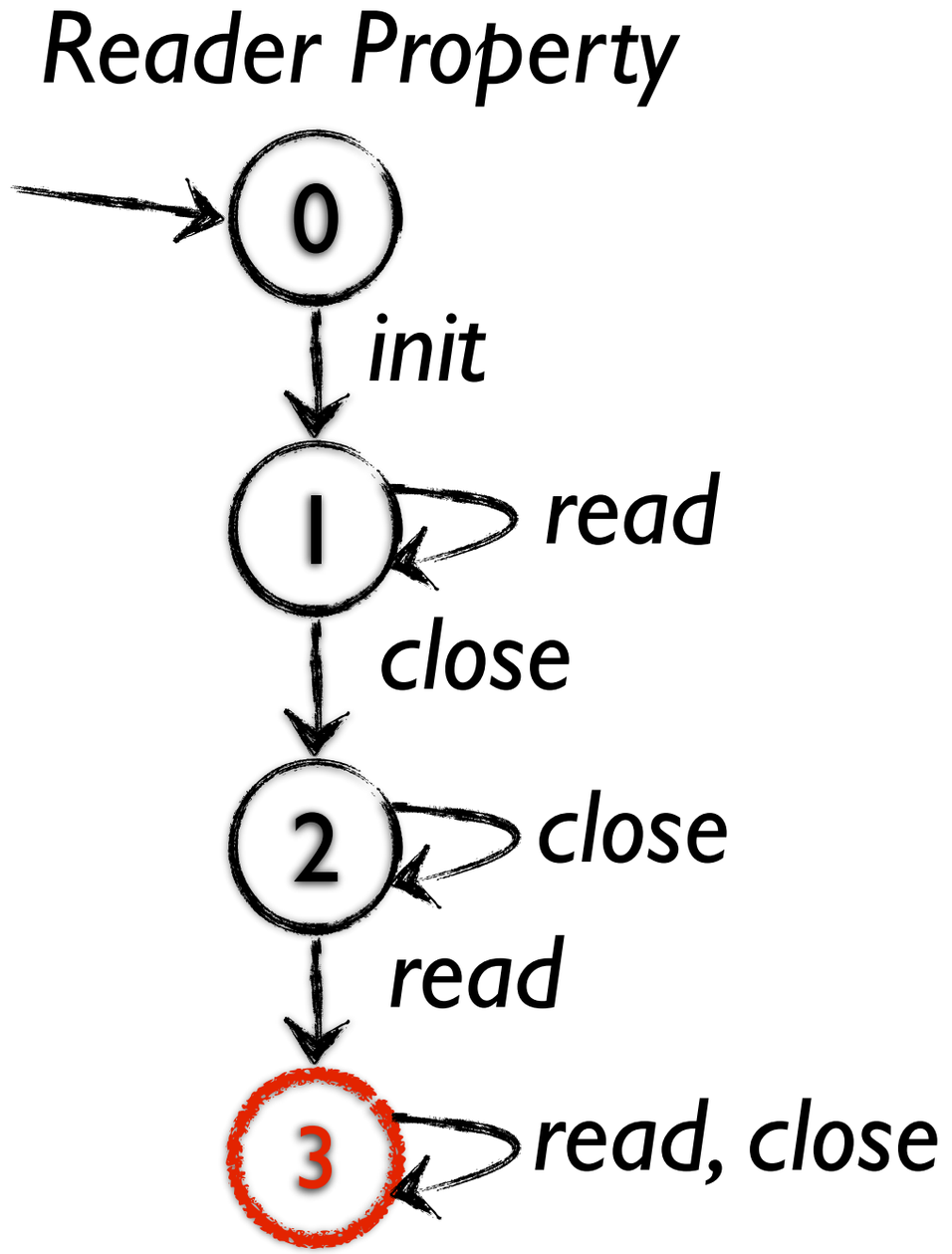
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```



Backward IFDS data flow analysis

$O(|E| \times |D|^3)$

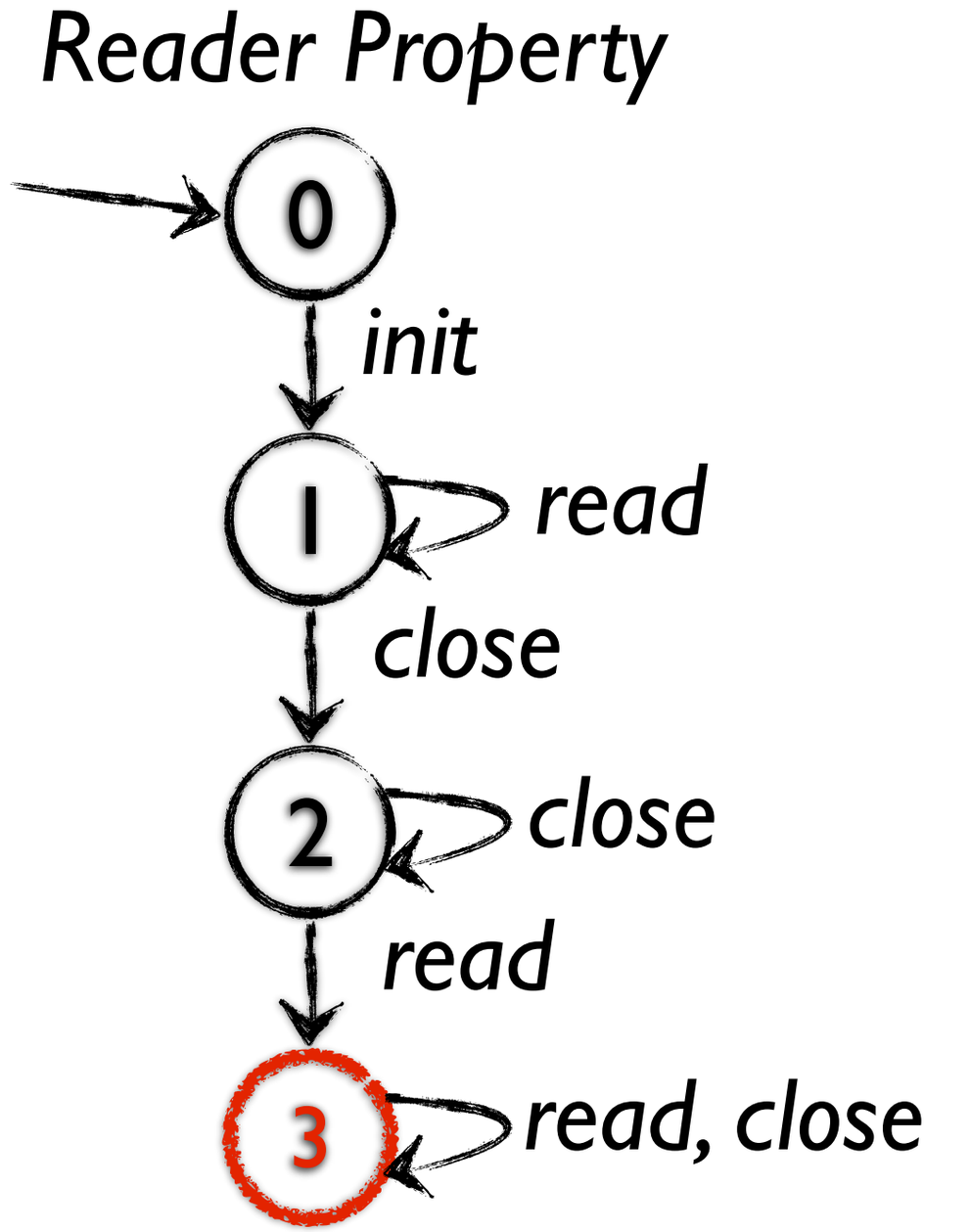
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```



Backward IFDS data flow analysis

$O(|E| \times |D|^3)$

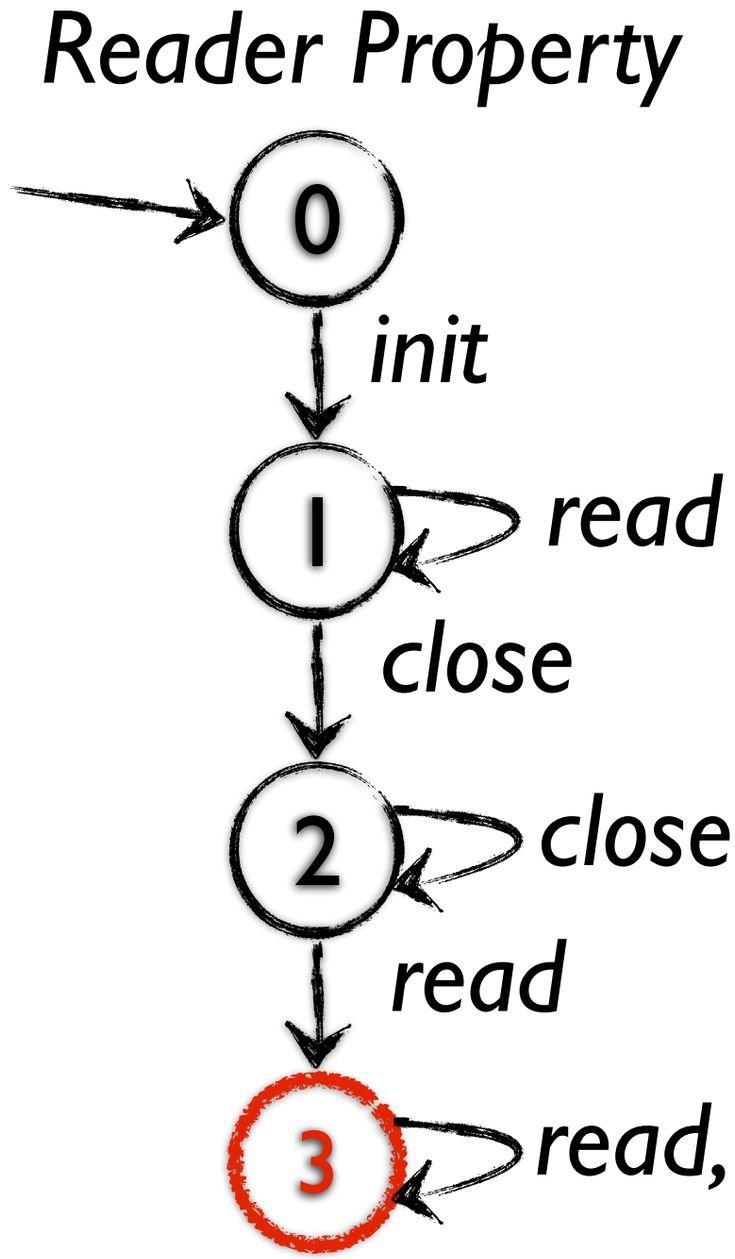
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```



Backward IFDS data flow analysis

$O(|E| \times |D|^3)$

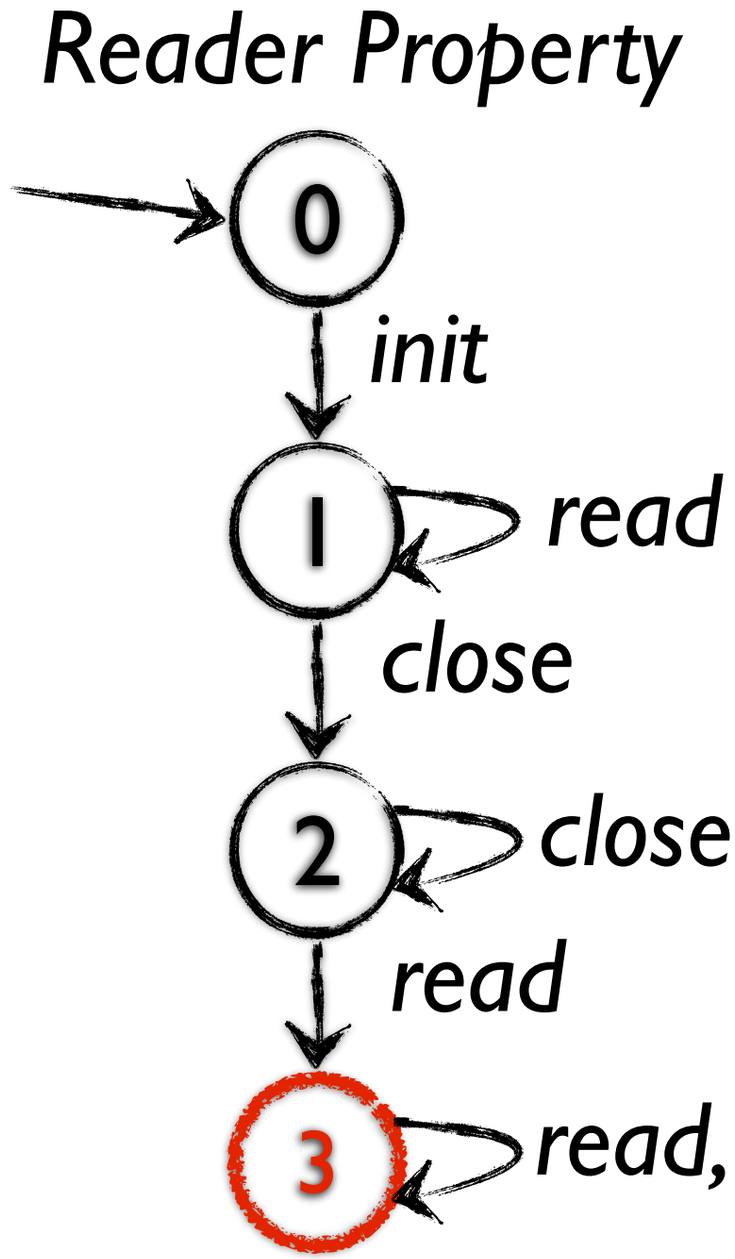
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```



Backward IFDS data flow analysis

$O(|E| \times |D|^3)$

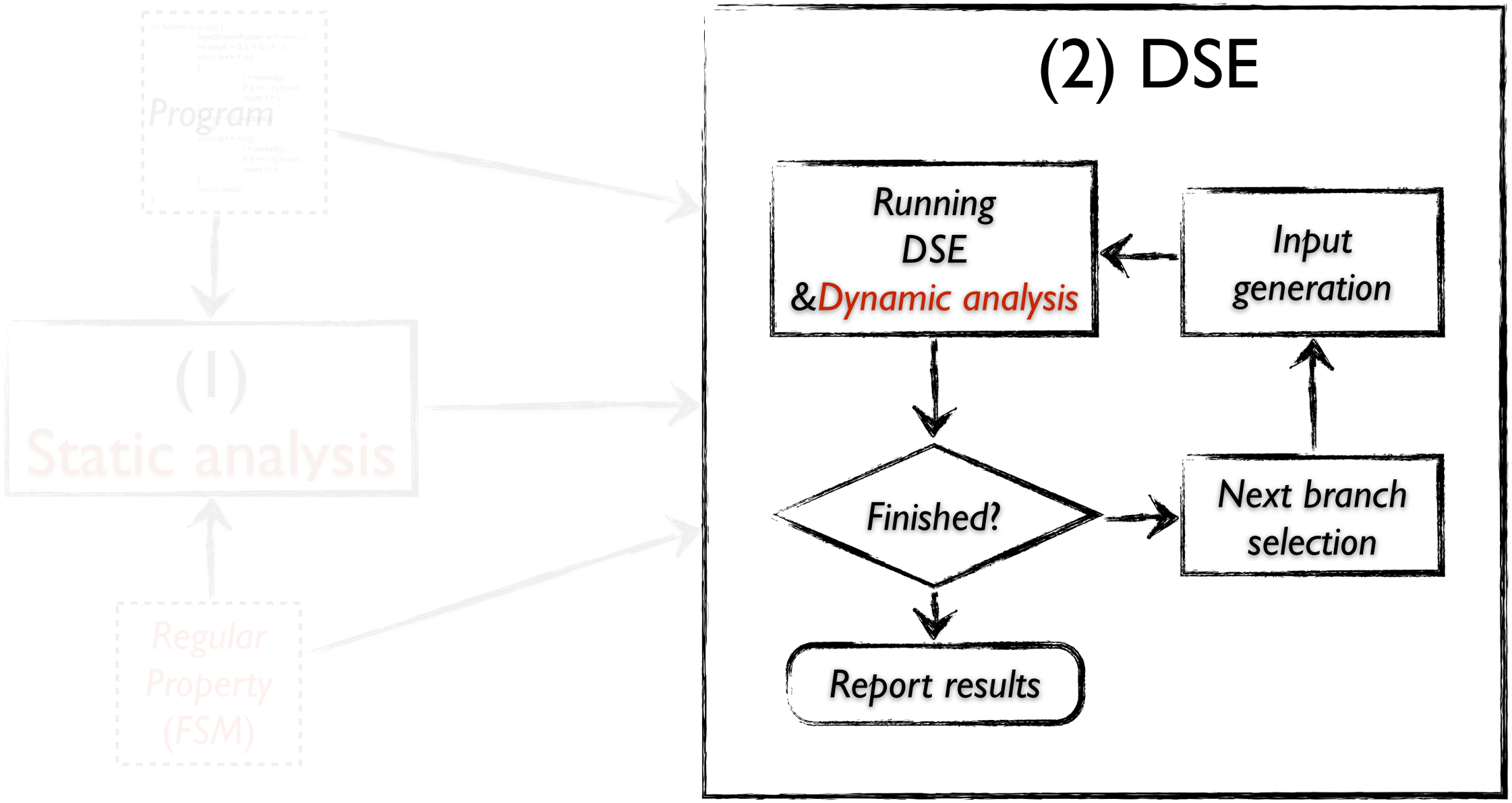
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
      w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```



Backward IFDS
data flow analysis

$O(|E| \times |D|^3)$

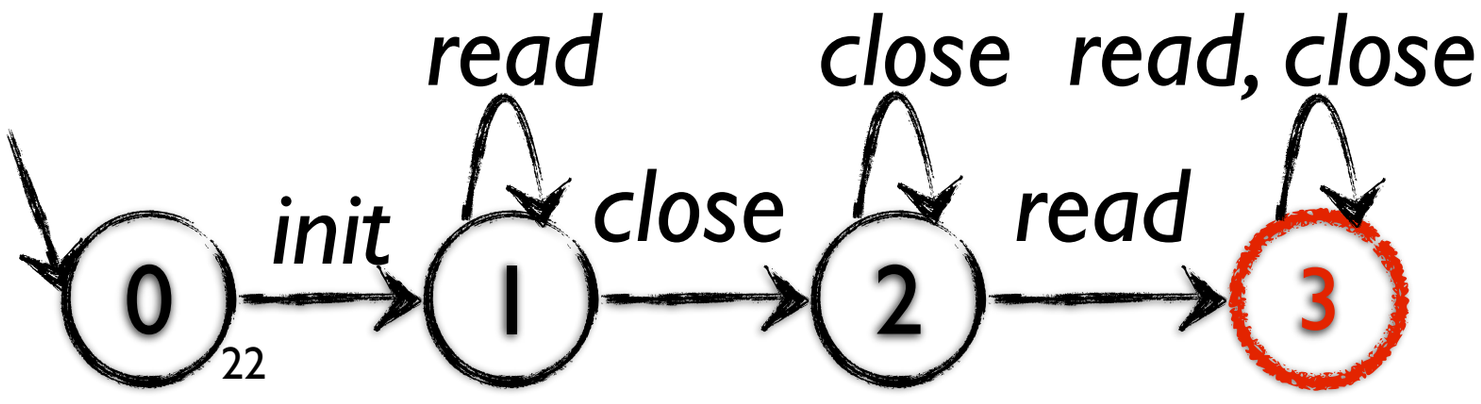
Guided DSE Procedure



Preset-1st Iteration

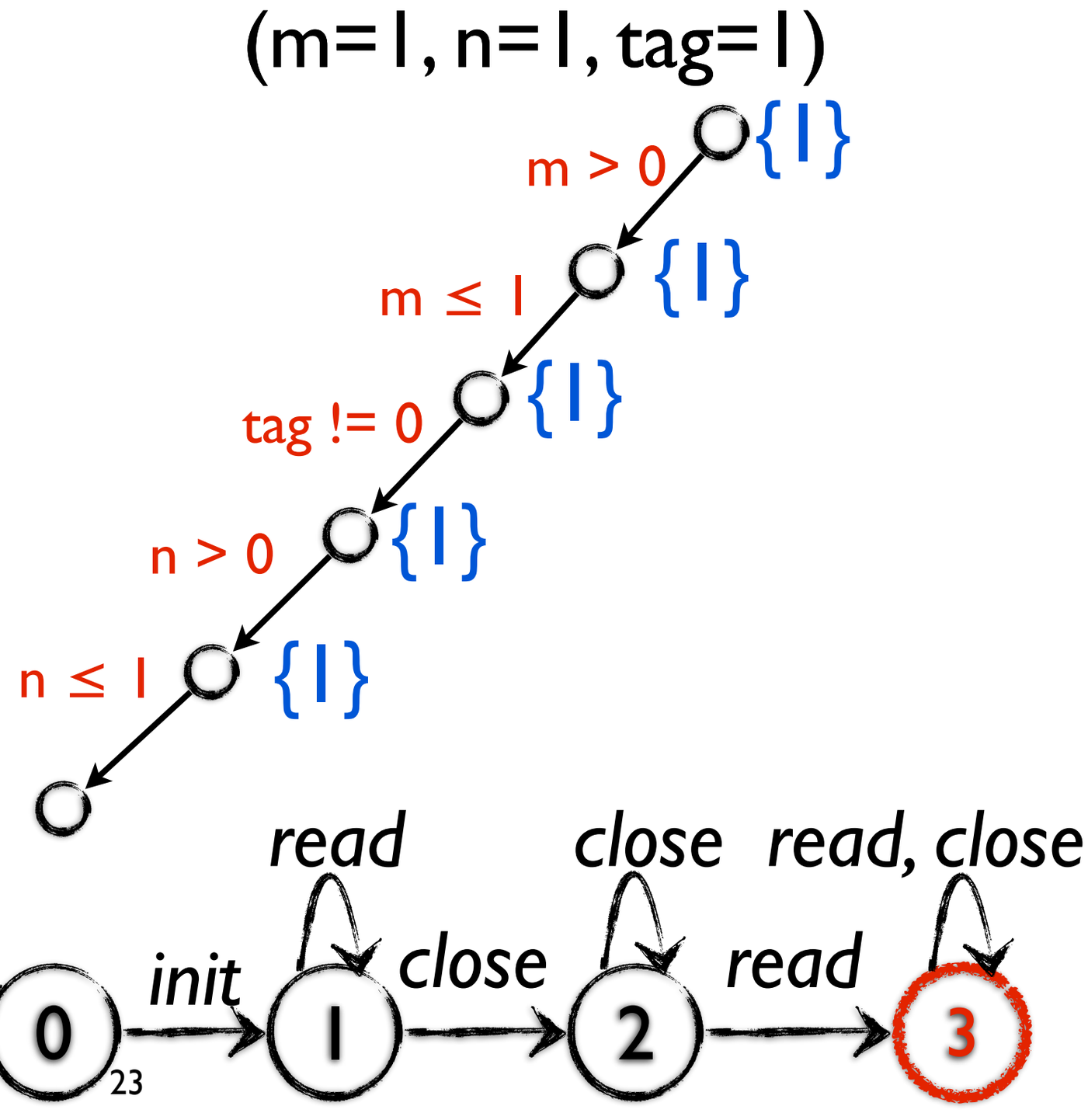
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

(m=1, n=1, tag=1)



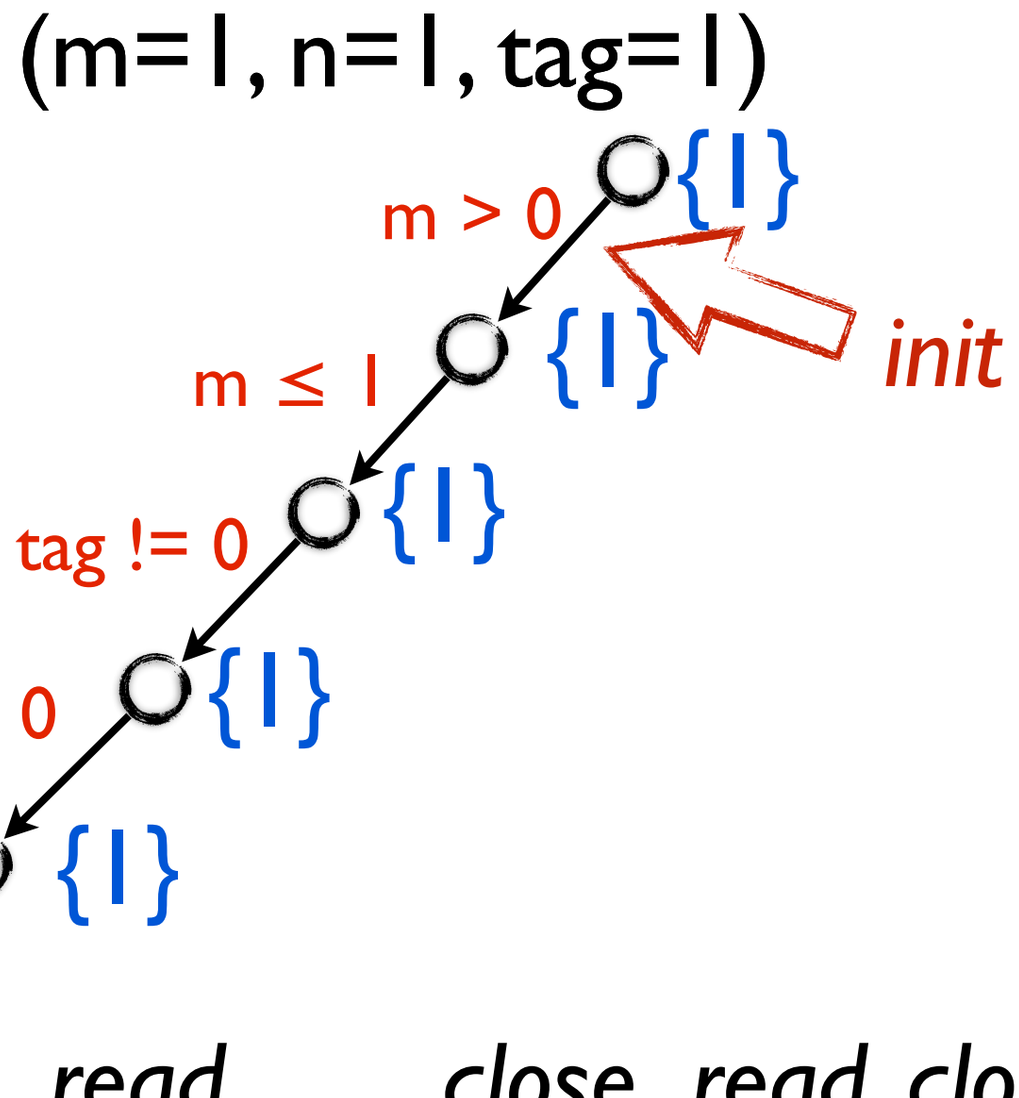
Preset-1st Iteration

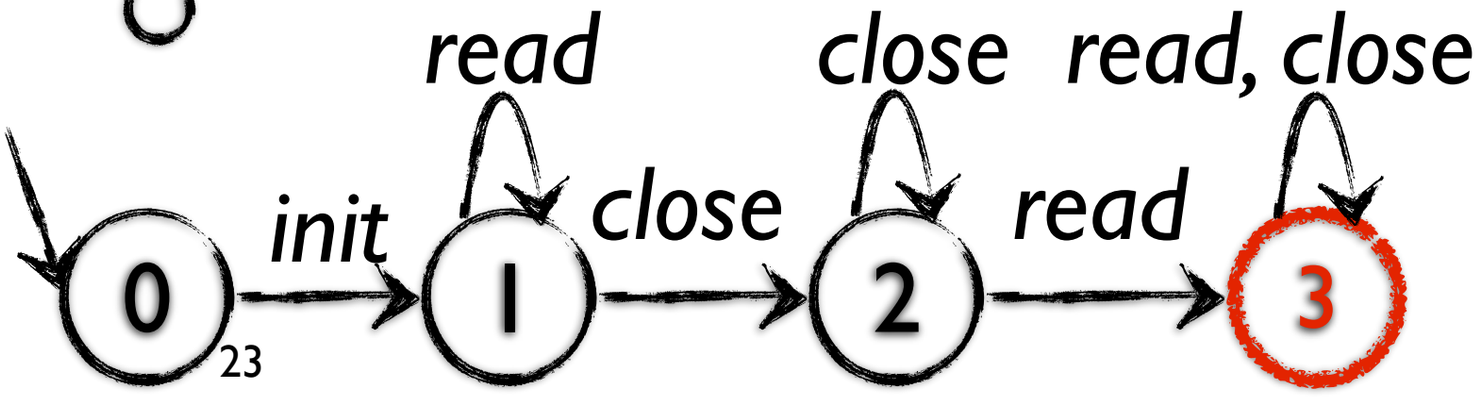
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```



Preset-1st Iteration

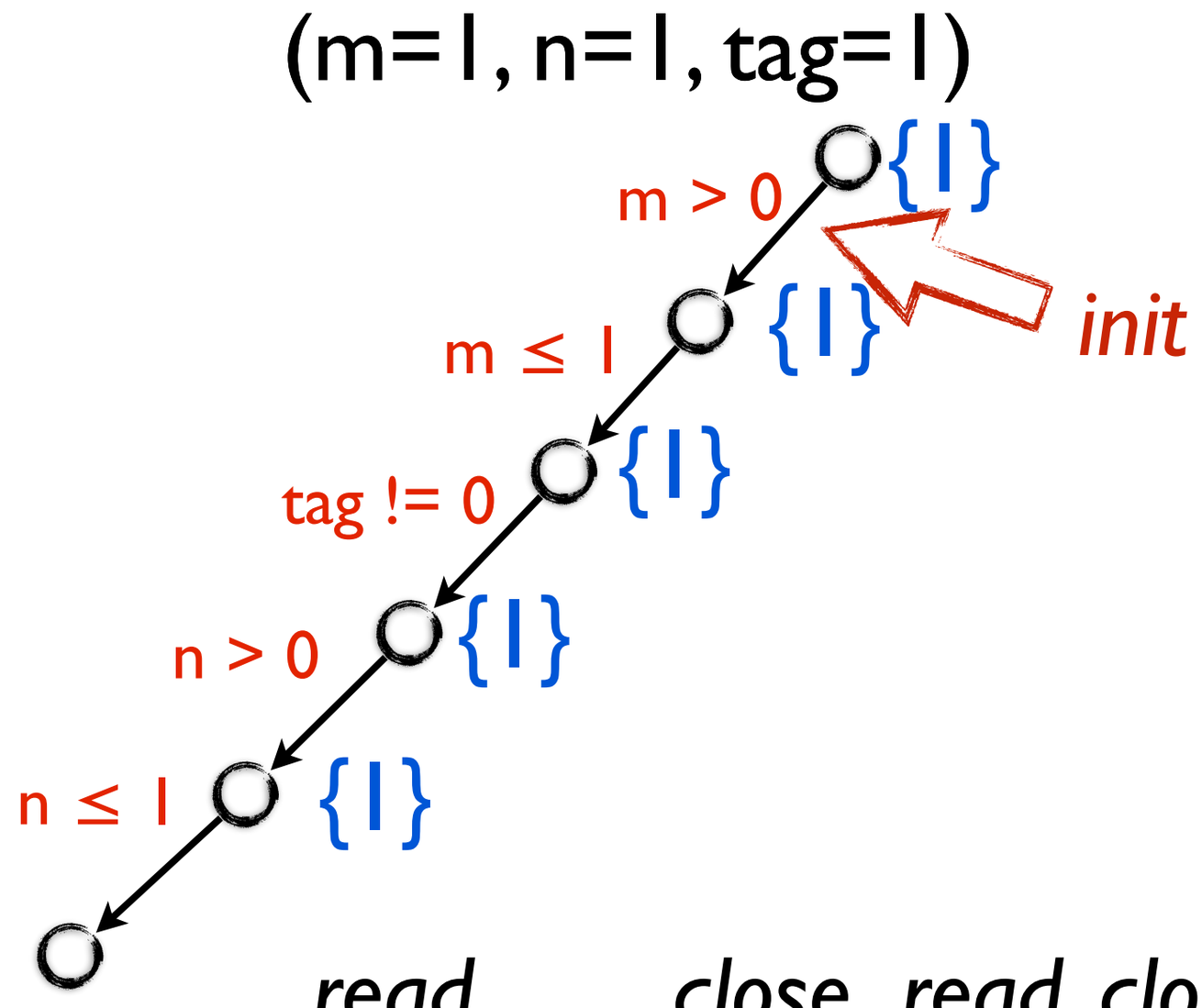
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

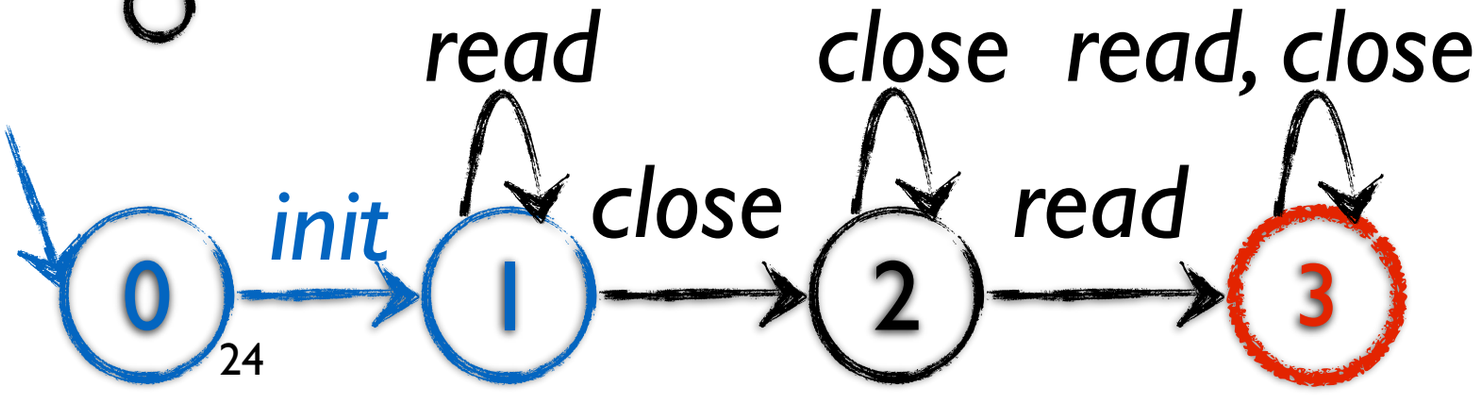




Preset-1st Iteration

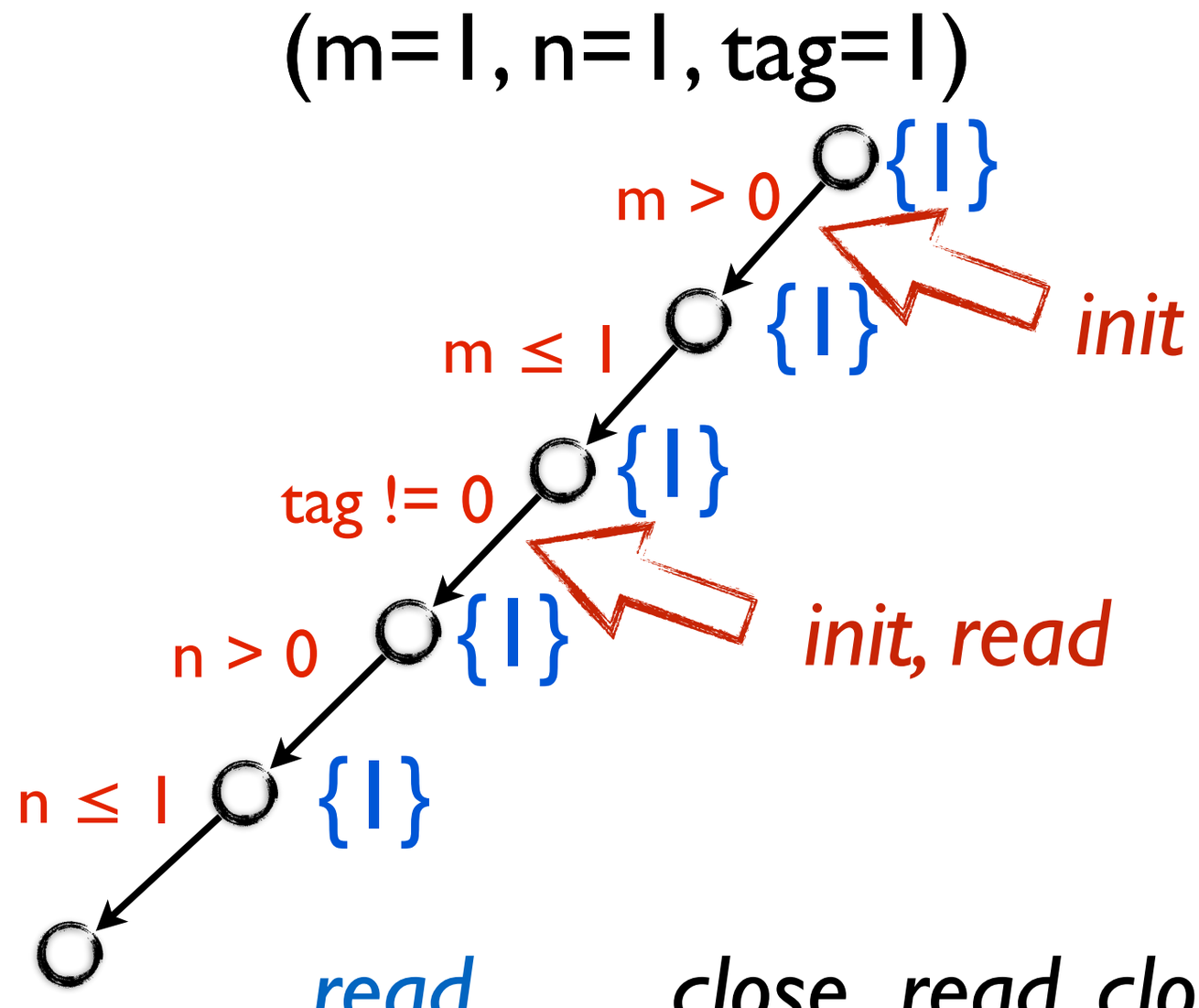
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

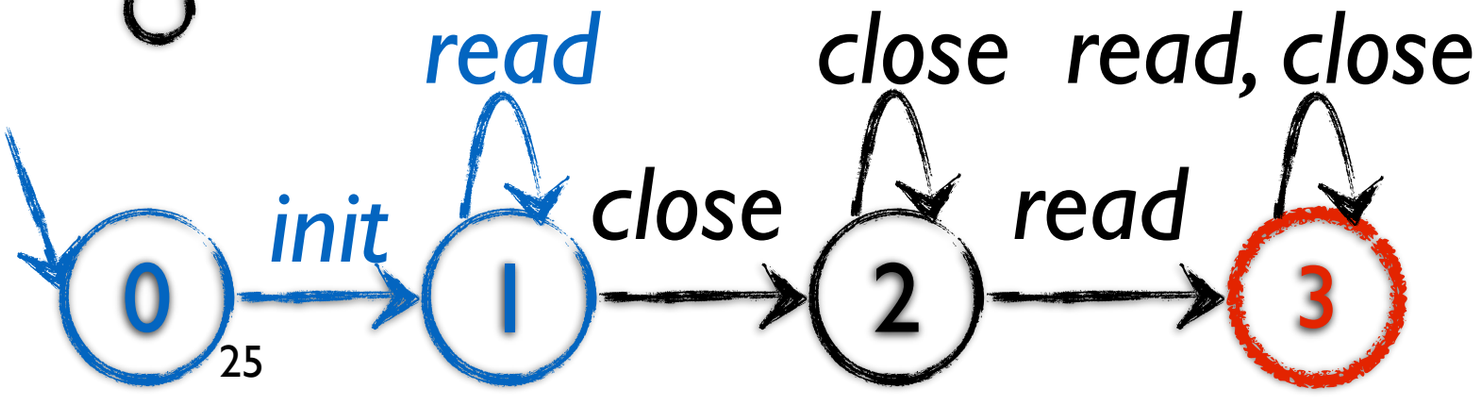




Preset-1st Iteration

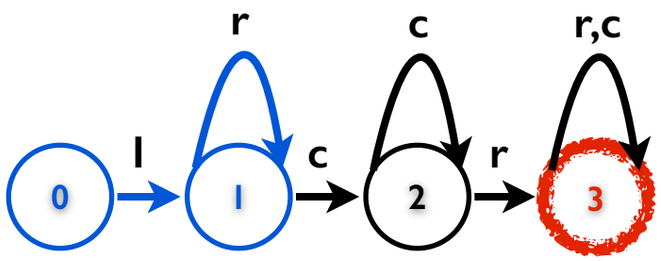
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...;
   int result = 0, k = 0, i = -1;
   while (k++ < m)
   {
      i = w.read();
      if (i == -1) break;
      result += i;
   }
   if (tag == 0) w.close();
   k = 0;
   while (k++ < n){
      i = w.read();
      if (i == -1) break;
      result -= i;
   }
   return result;
}
```

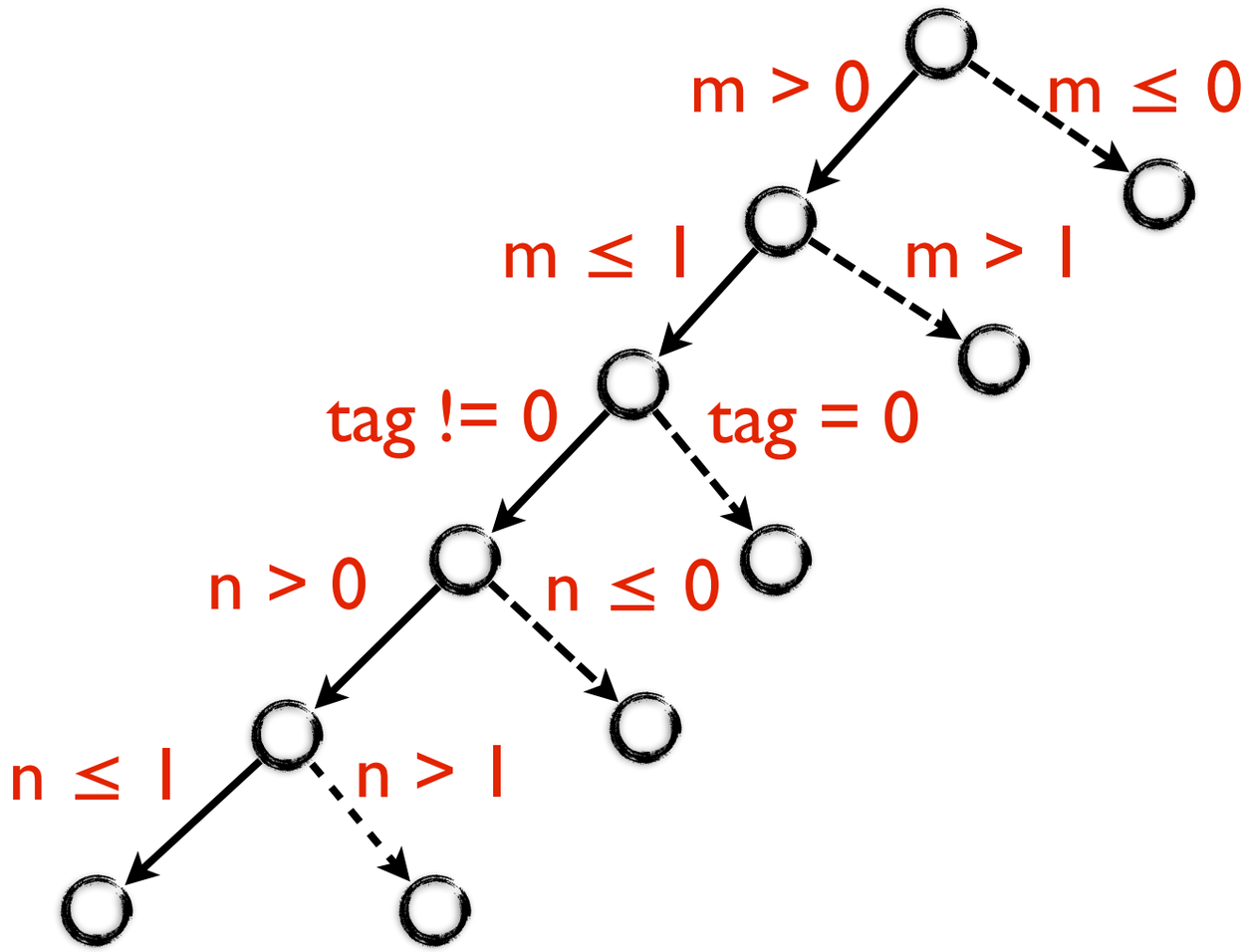




Guided DSE-1st Iteration

(m=1, n=1, tag=1)

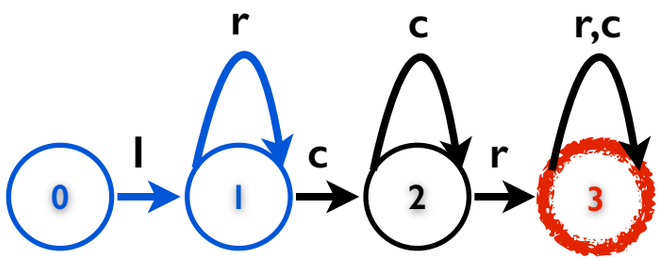


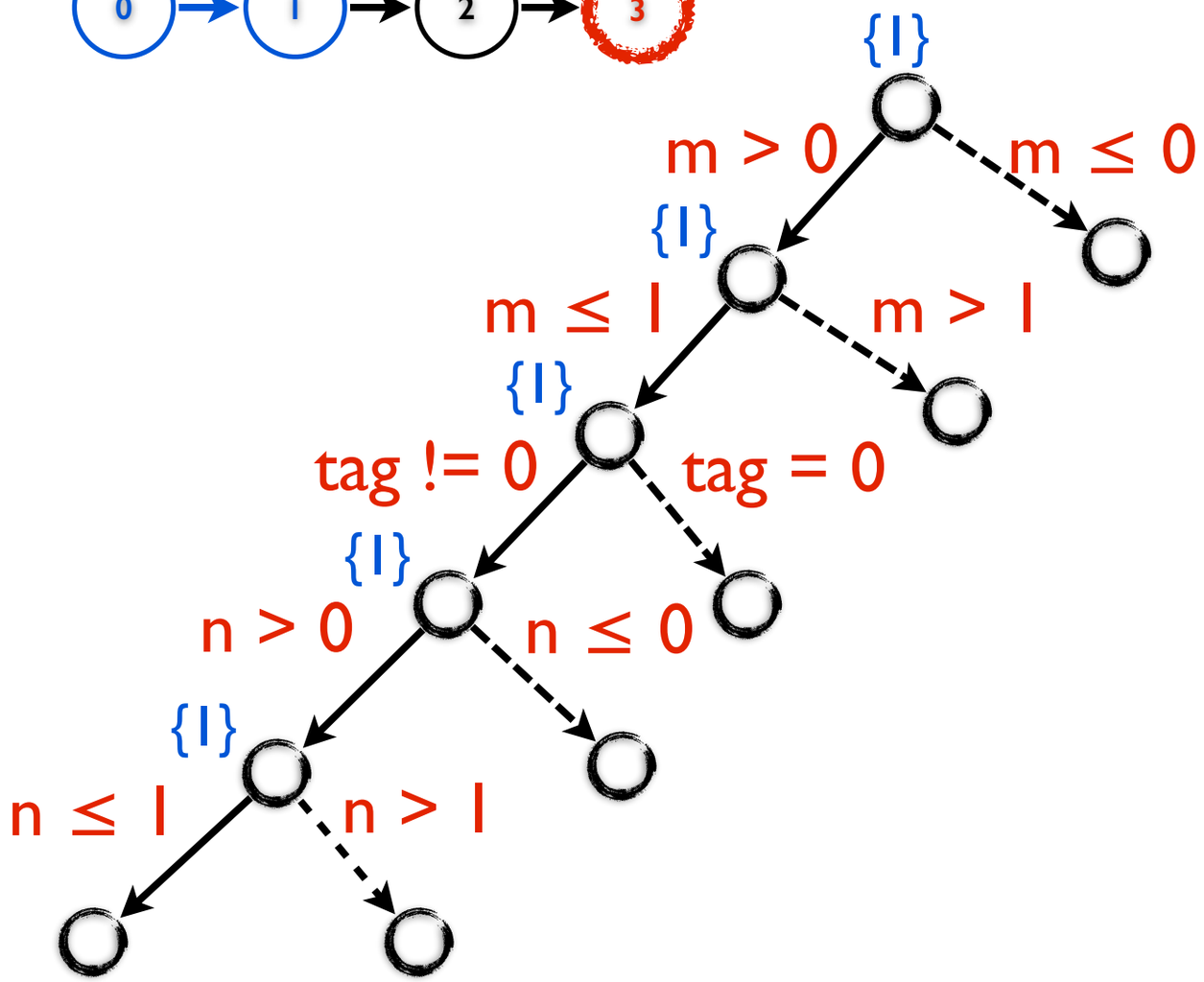



```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE-1st Iteration

(m=1, n=1, tag=1)

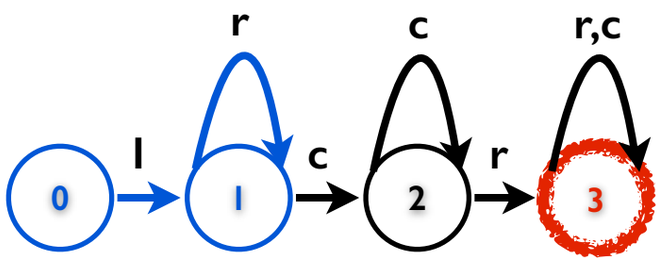


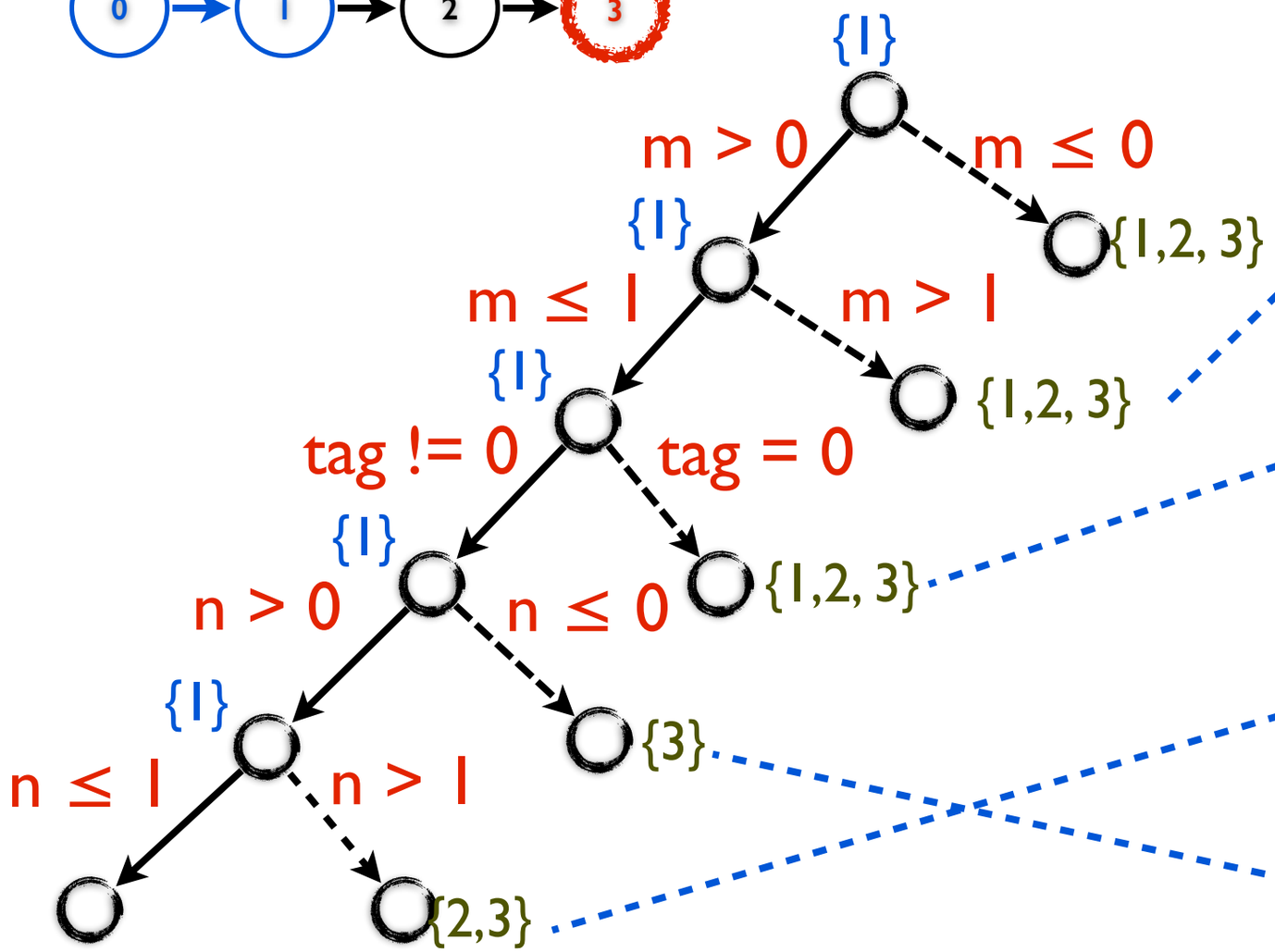



```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE-1st Iteration

(m=1, n=1, tag=1)

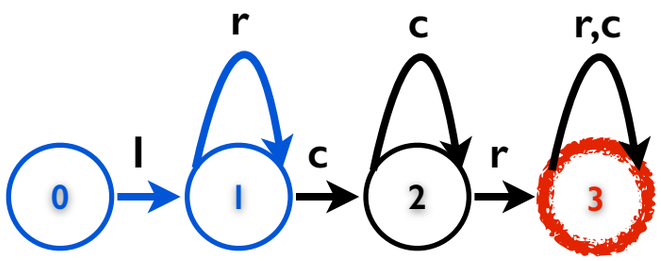


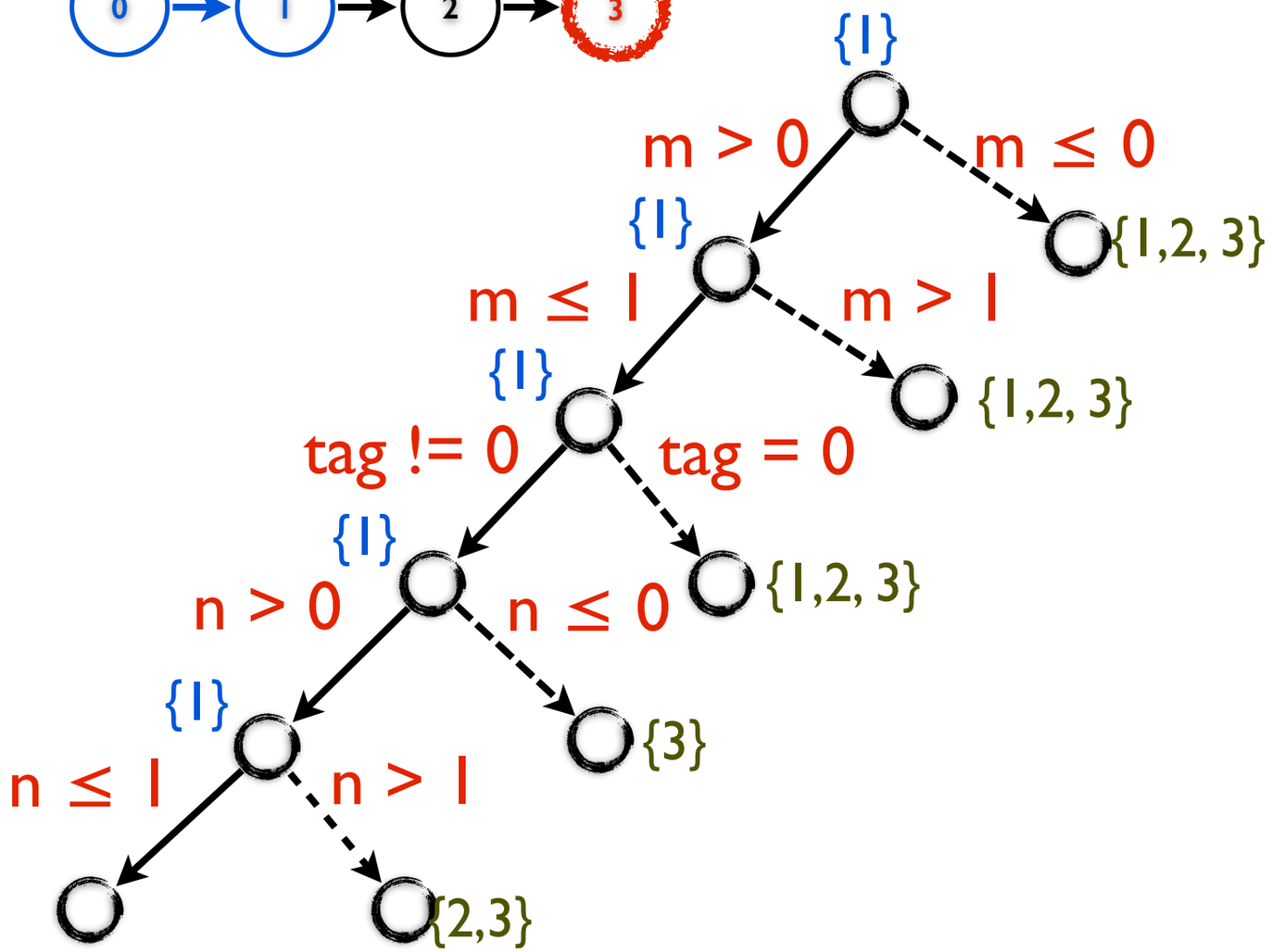



```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
      w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE-1st Iteration

(m=1, n=1, tag=1)

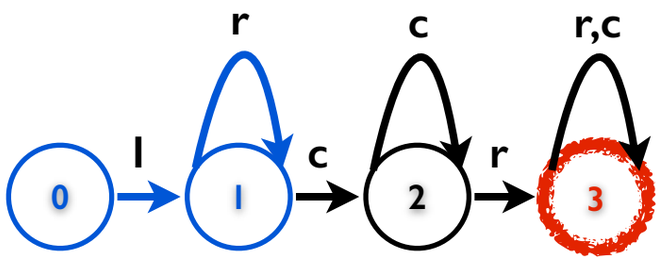


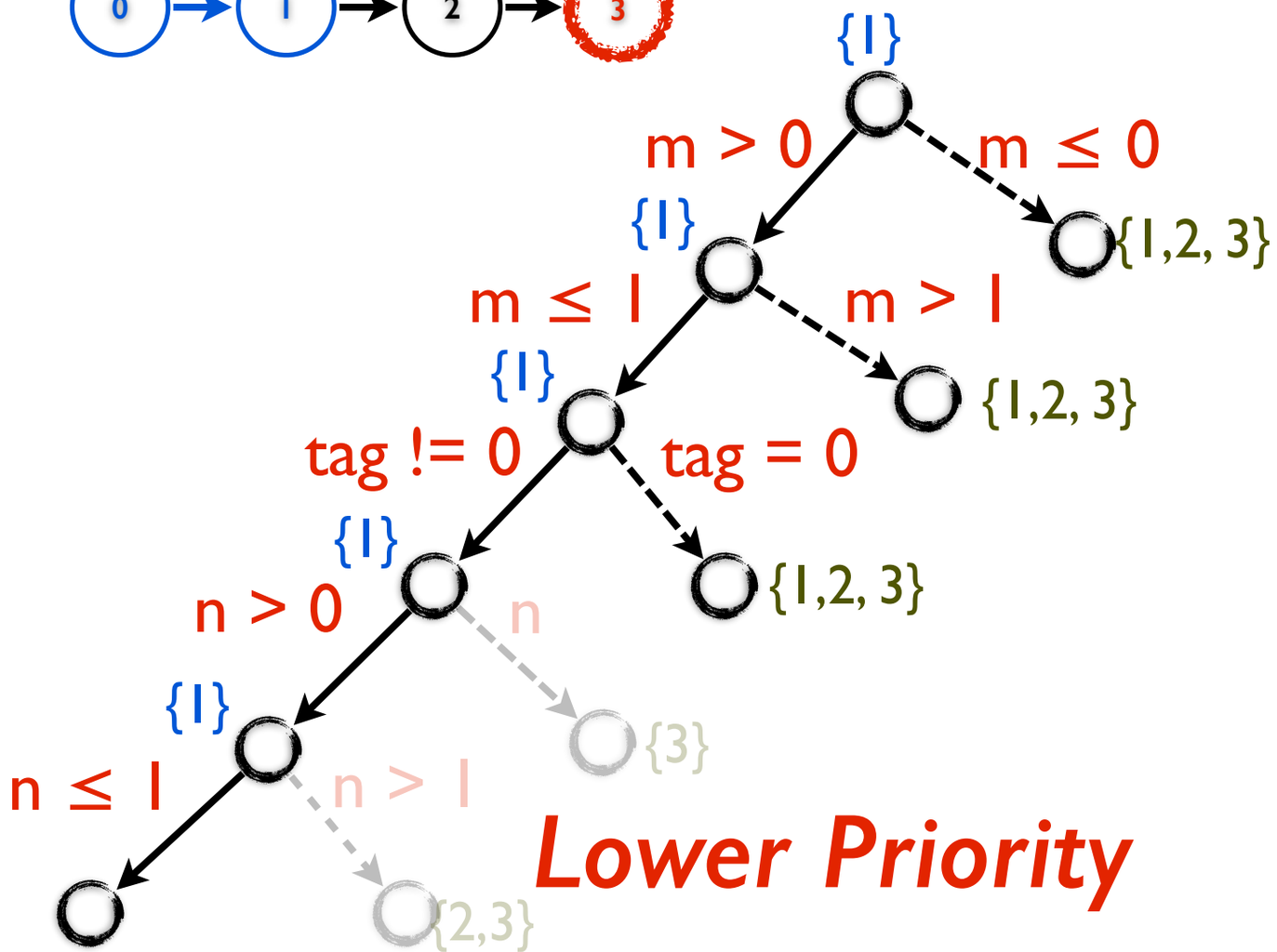



```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE-1st Iteration

(m=1, n=1, tag=1)



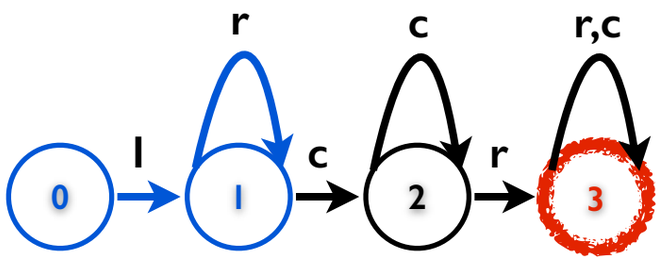


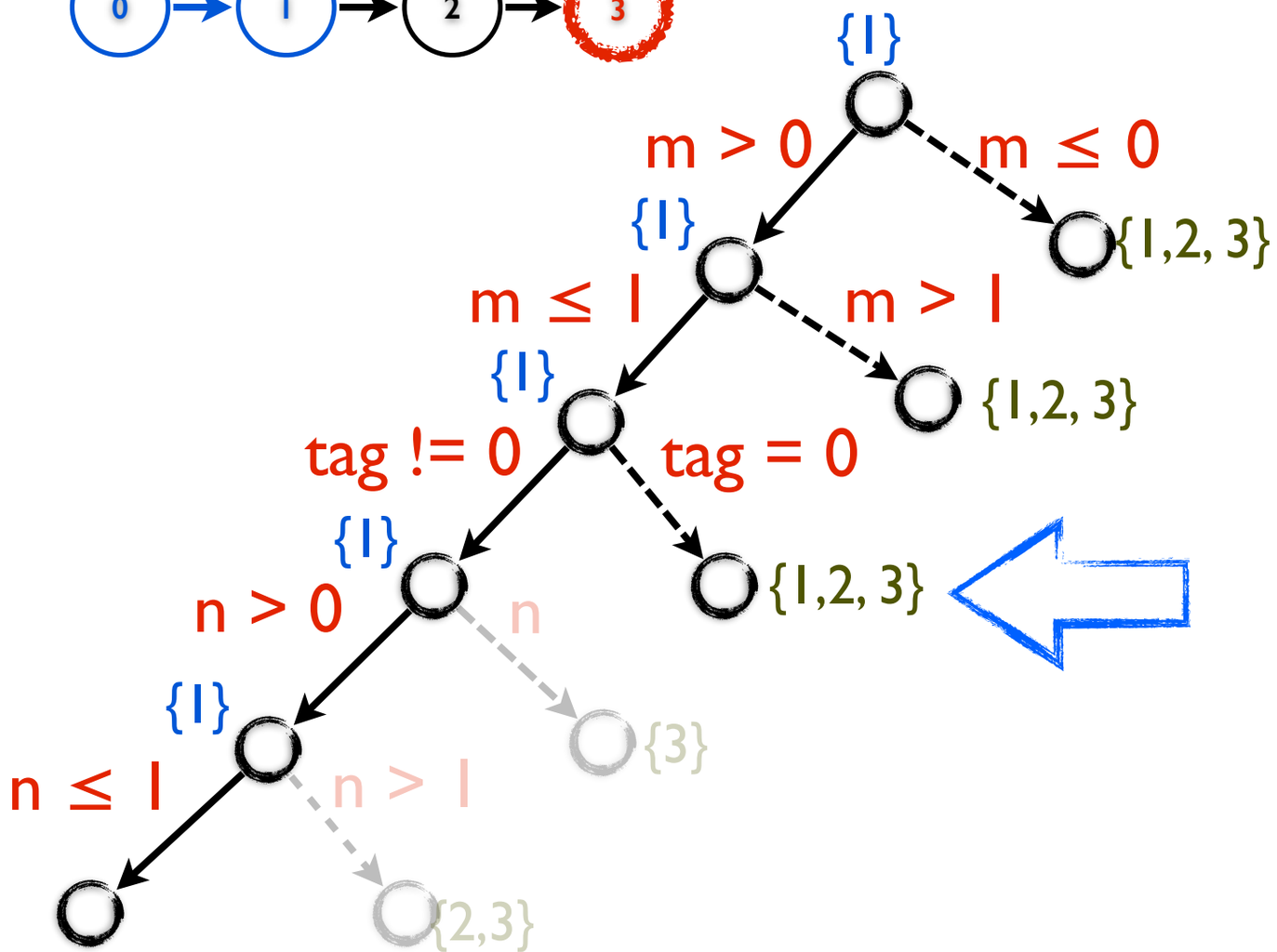
Lower Priority

```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE-1st Iteration

(m=1, n=1, tag=1)

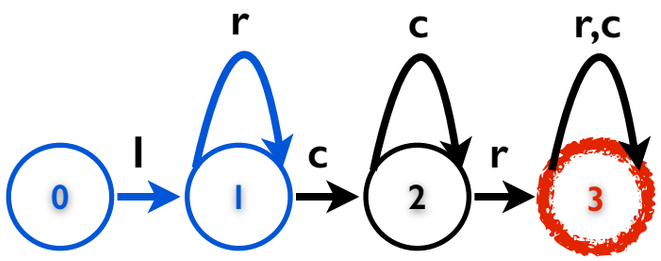


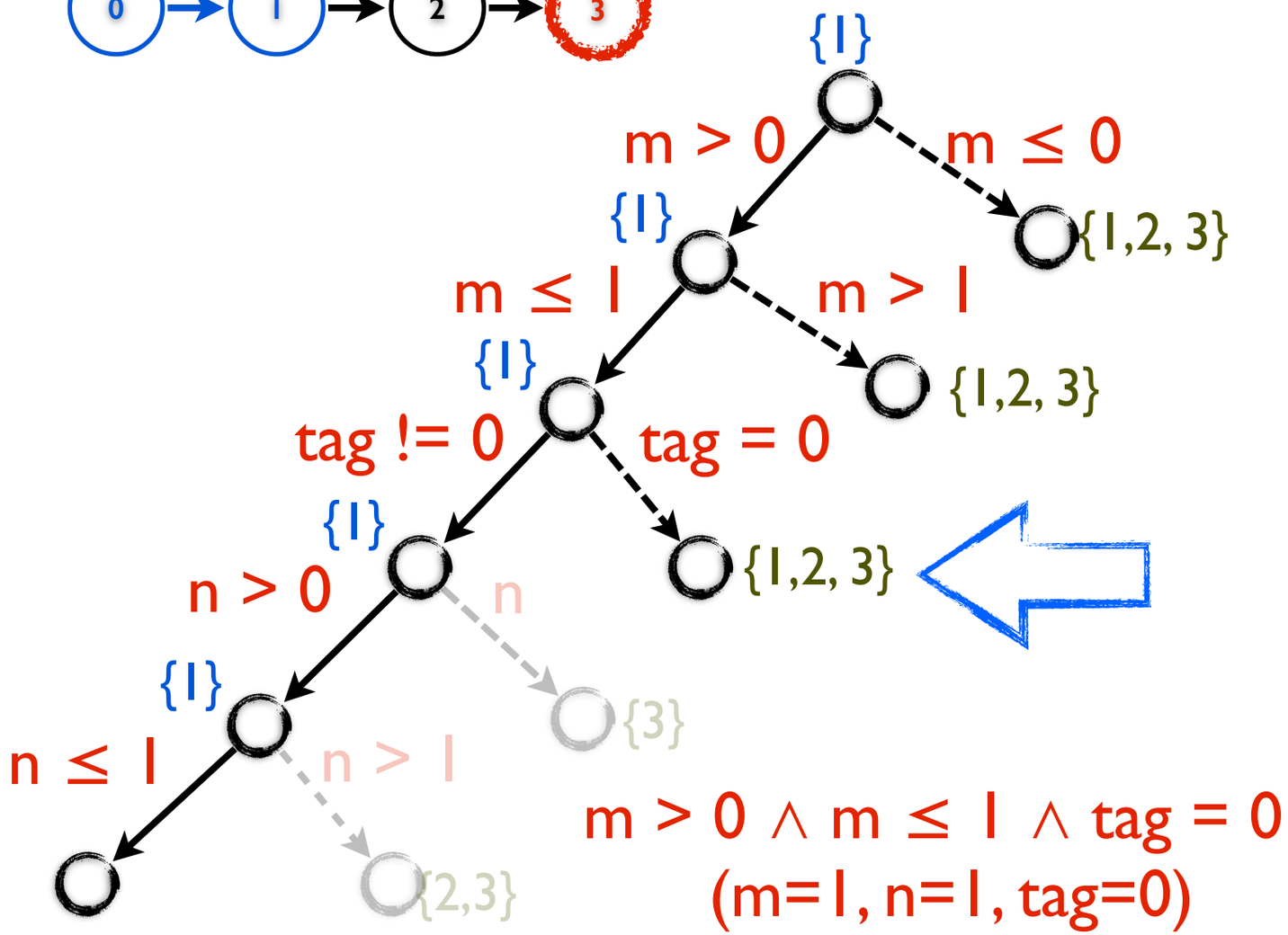



```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE-1st Iteration

(m=1, n=1, tag=1)





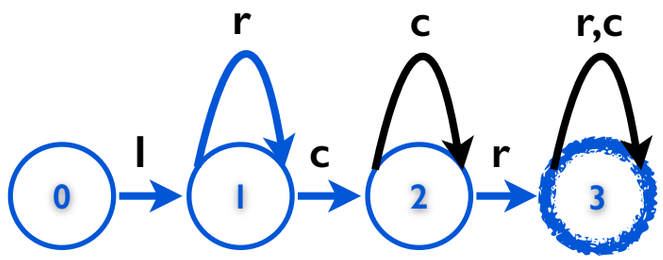
m > 0 ∧ m ≤ 1 ∧ tag = 0

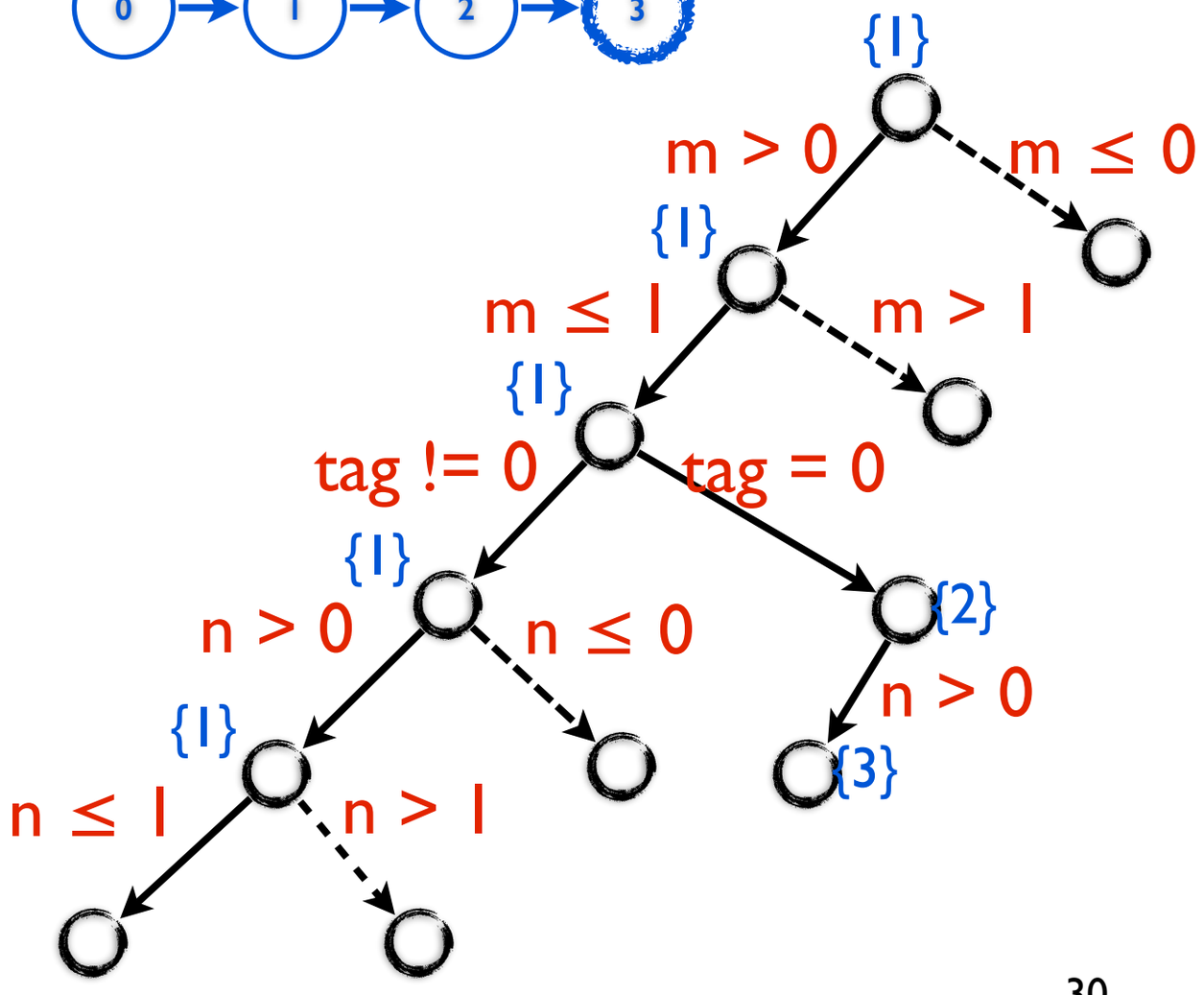
(m=1, n=1, tag=0)

```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE-2nd Iteration

(m=1, n=1, tag=0)

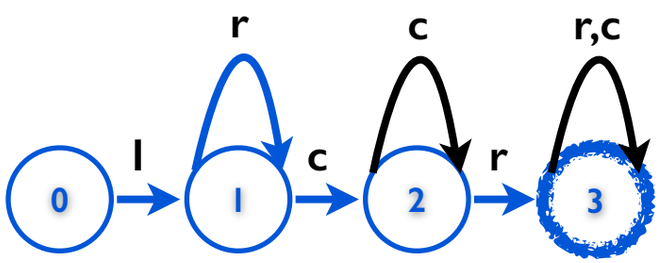


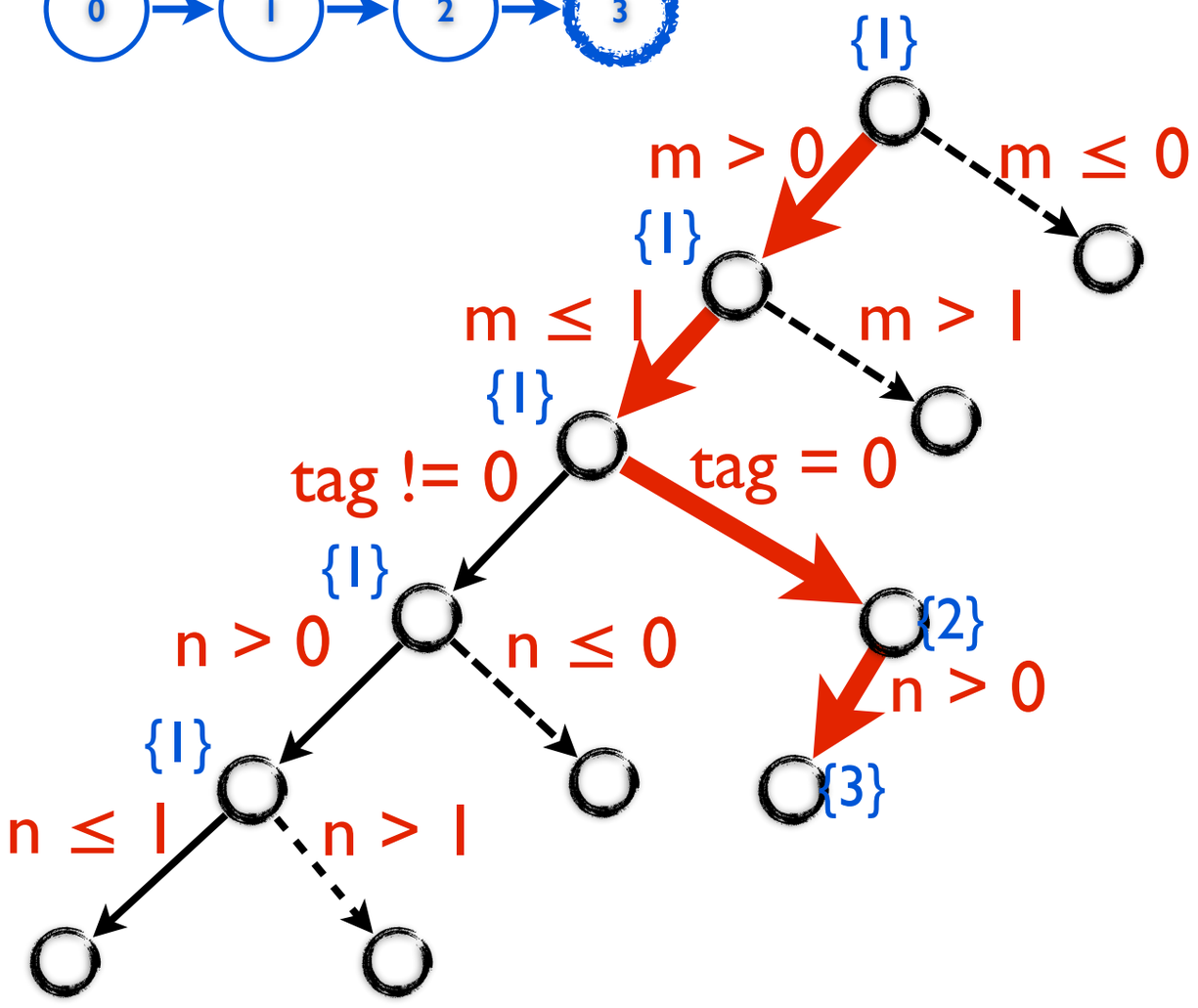



```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE

(m=1, n=1, tag=0)

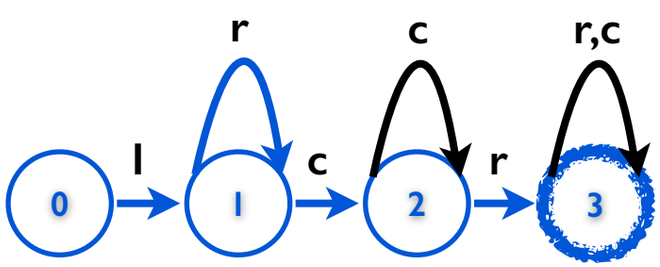


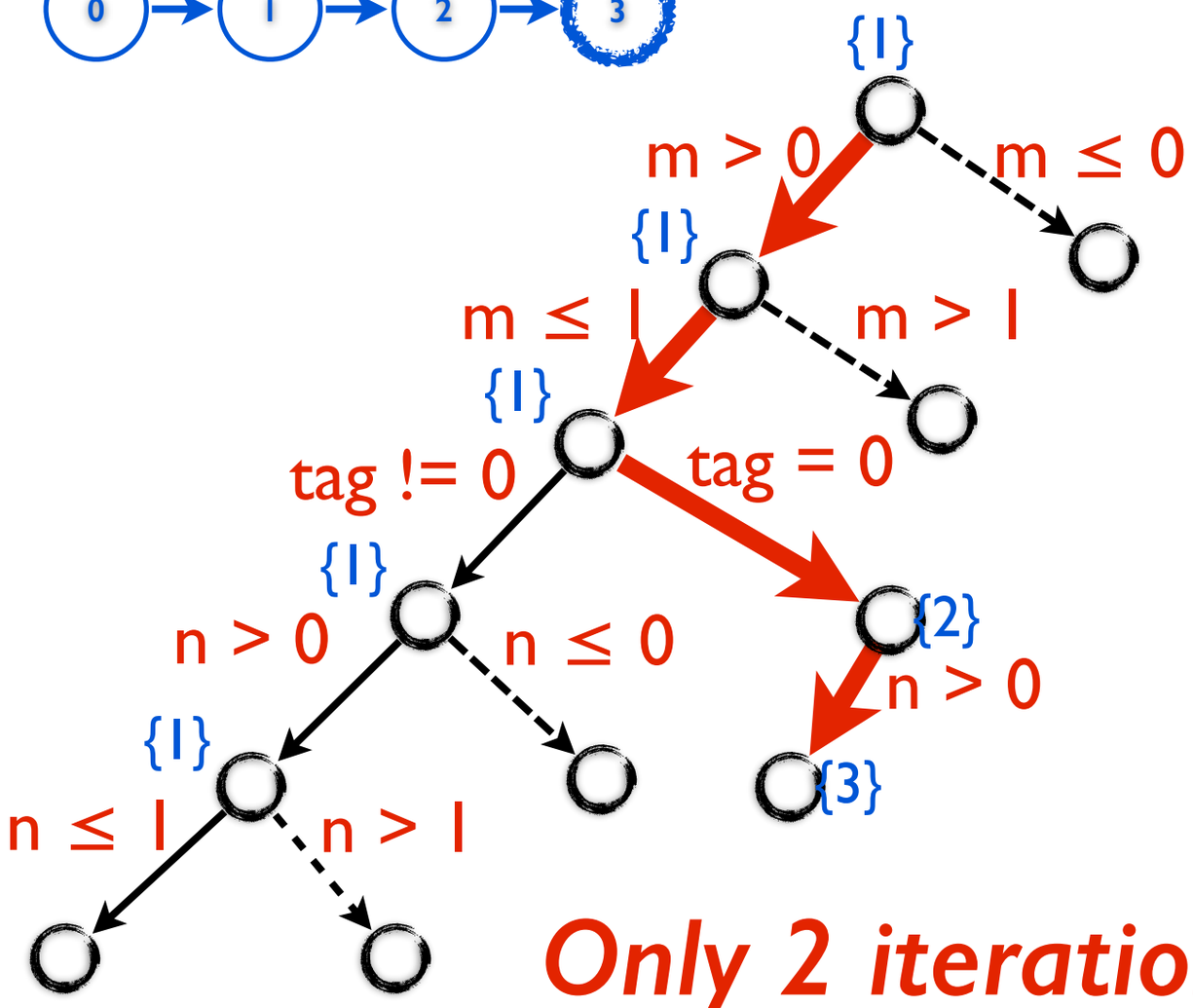



```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

Guided DSE

(m=1, n=1, tag=0)





Only 2 iterations

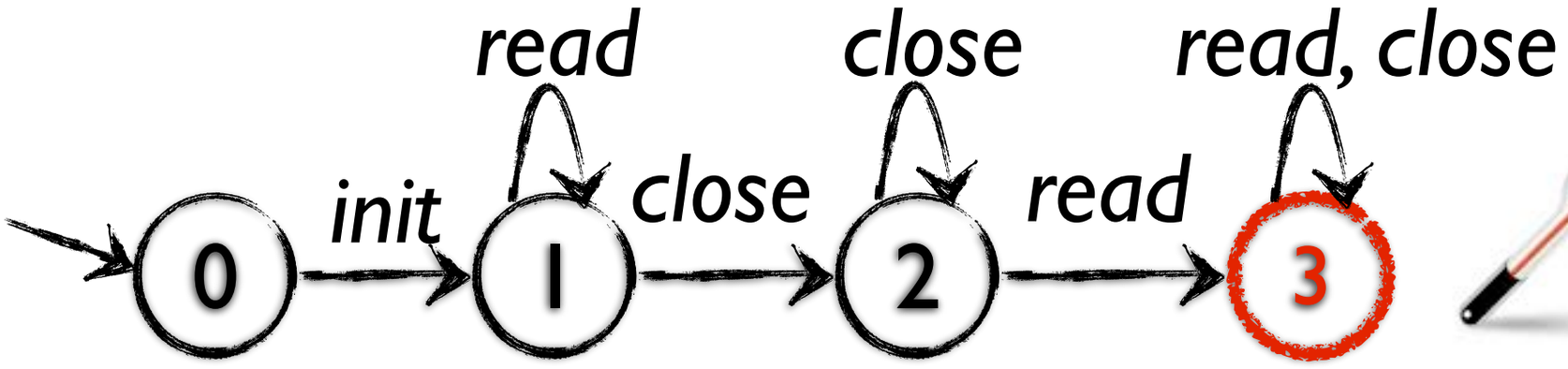
```
int foo(int m, n, tag) {
   InputStreamReader w = new ...; //{0}
   int result = 0, k = 0, i = -1; //{1,2,3}
   while (k++ < m) //{1,2,3}
   {
      i = w.read(); //{1,2,3}
      if (i == -1) break; //{1,2,3}
      result += i; //{1,2,3}
   }
   if (tag == 0) //{1,2,3}
       w.close(); //{1,2,3}
   k = 0; //{2,3}
   while (k++ < n) //{2,3}
   {
      i = w.read(); //{2,3}
      if (i == -1) break; //{2,3}
      result -= i; //{2,3}
   }
   return result; //{3}
}
```

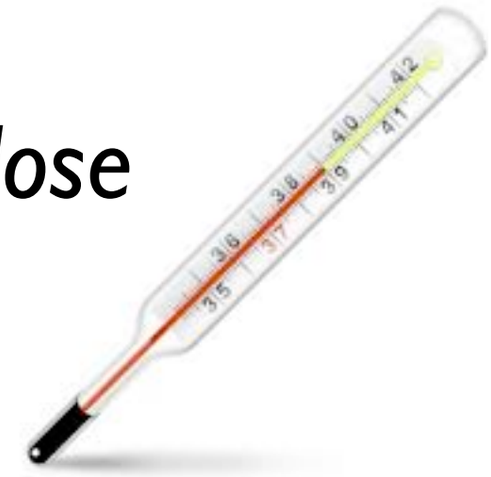
Implementation & Experiment Setup

- Implement based on JPF-JDart and WALA
- 13 real world open source Java programs
  - 225K LOC in total
- Properties
  - Typestate bug && User defined
- Analyze each program/property in 24 hours

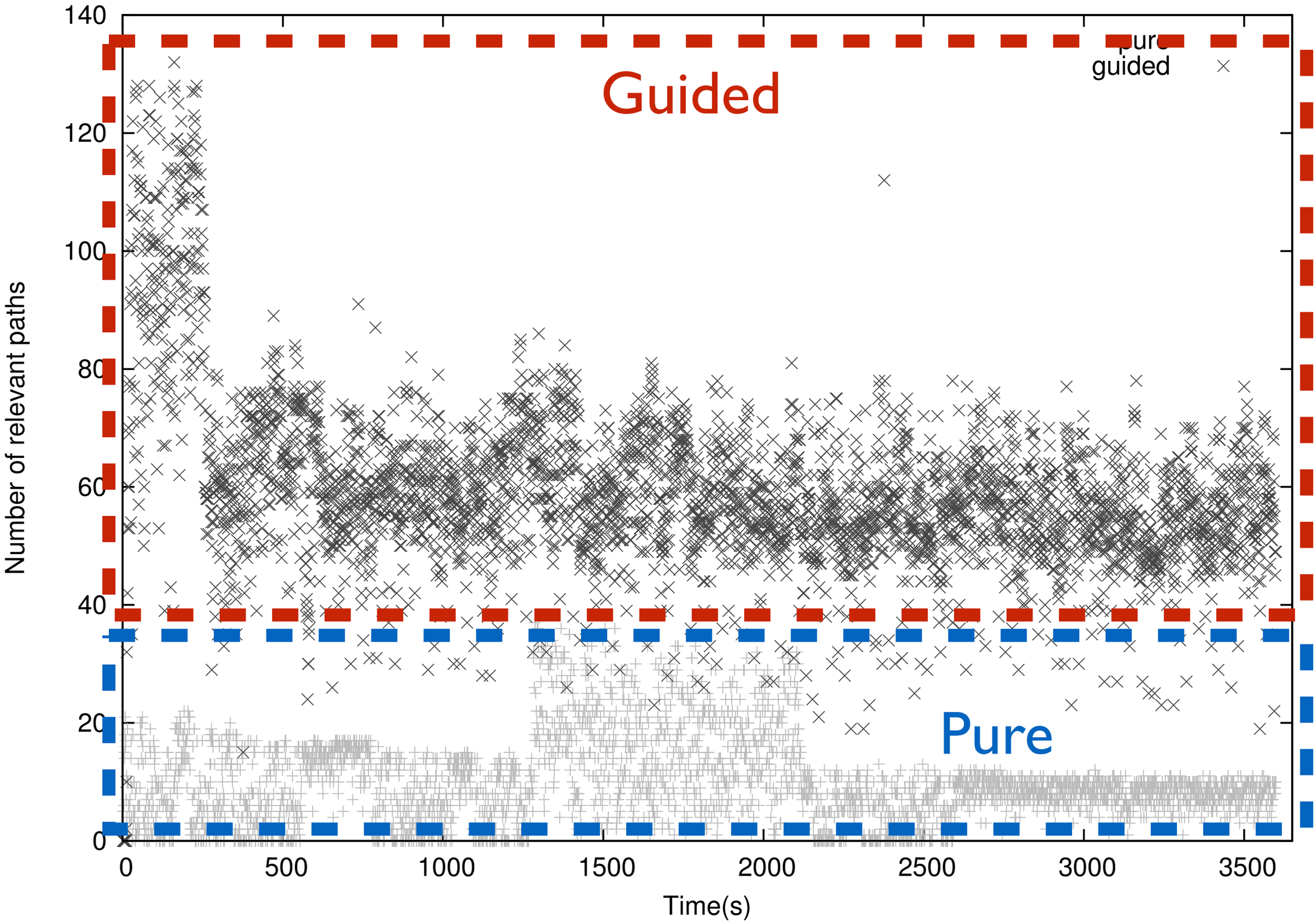
Evaluate Guiding Further

- Relevant path
- Transition times
- Shortest distance to the final state

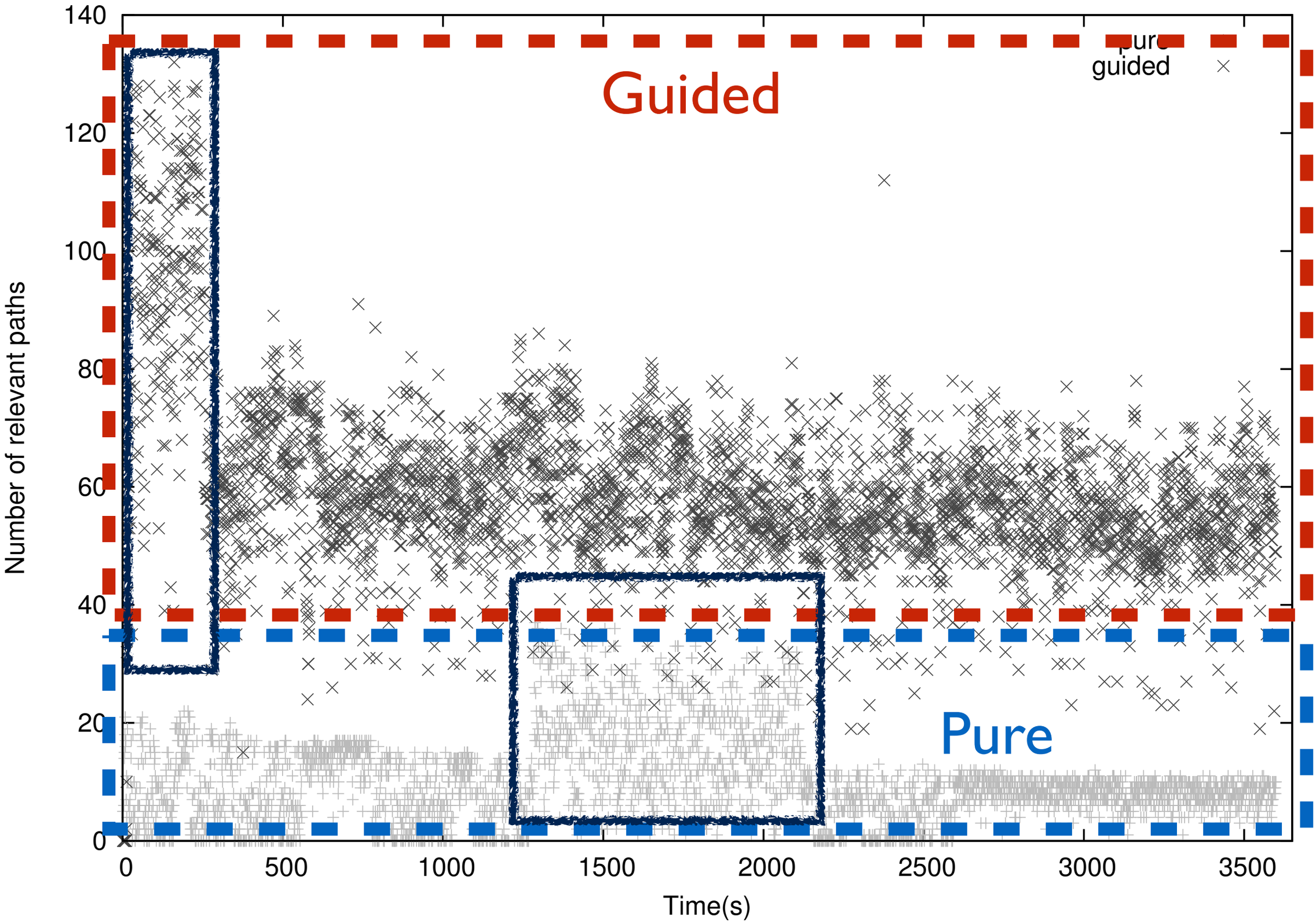




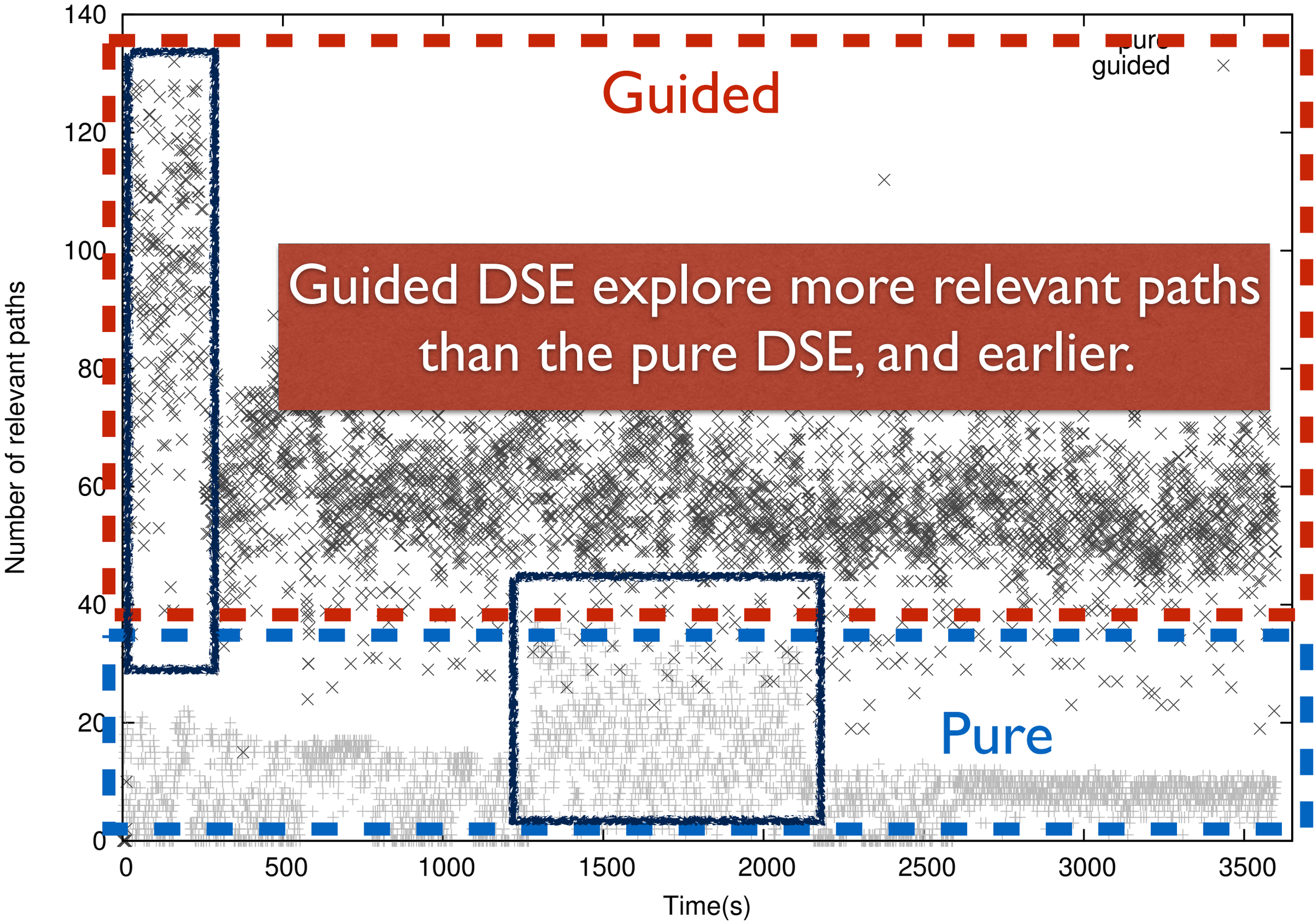
Relevant path distribution



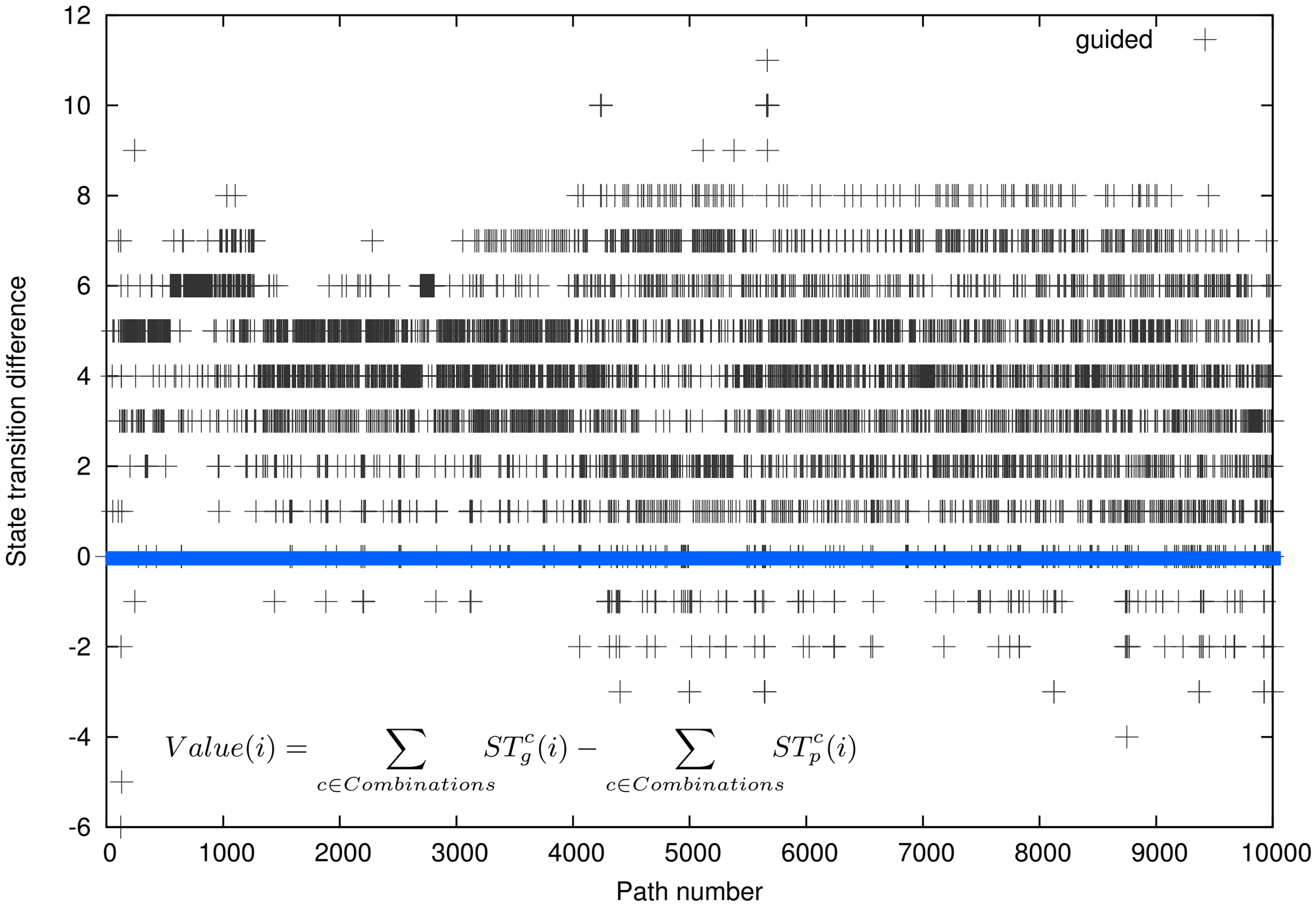
Relevant path distribution



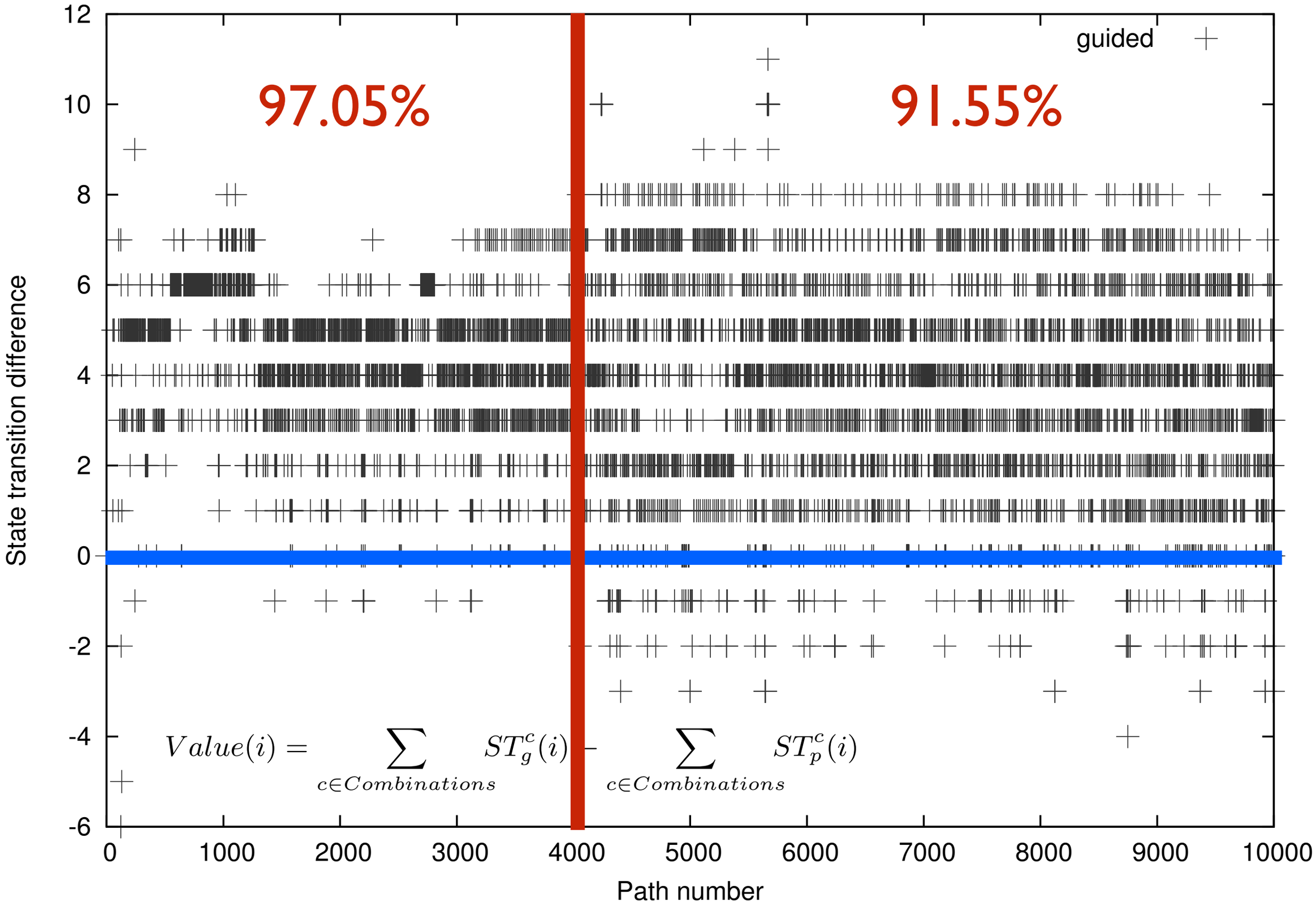
Relevant path distribution



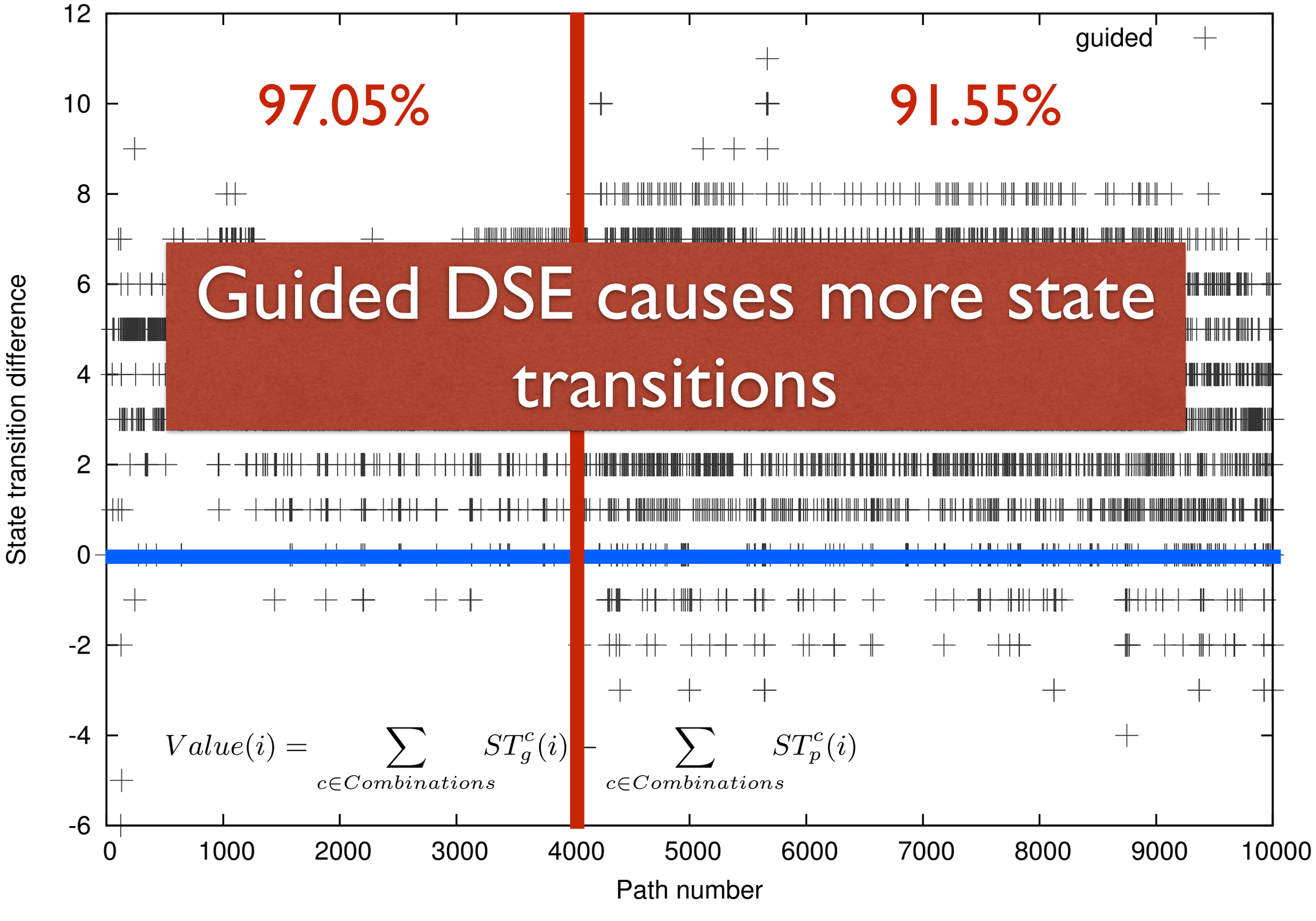
State Transition Difference



State Transition Difference

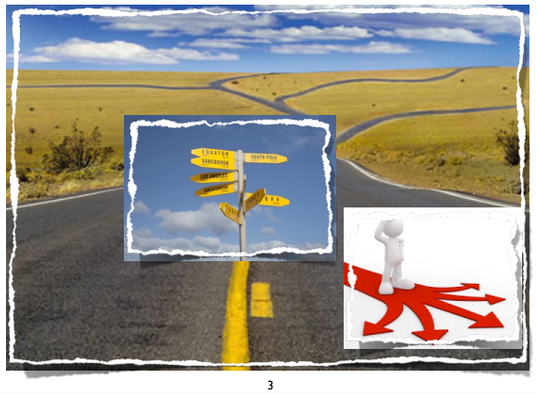


State Transition Difference



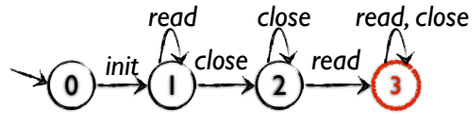
Conclusion

DSE needs guiding



3

How about a Regular Property?



A bug property: a file is read after closed

How to guide DSE to find a program path satisfying P as soon as possible?

5

Key Idea

history *Preset ∩ Postset ≠ ∅* *future*

Preset: the state that can be reached from the beginning to the branch location

Postset: the states from which it can reach a final state after executing the rest program after the branch location

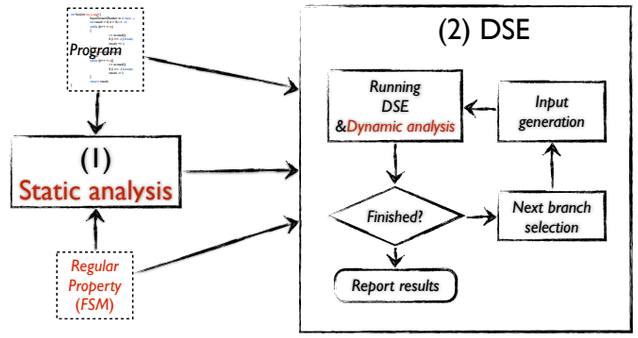
Dynamic analysis

Static analysis

Evaluate a branch based on its history and future behaviors

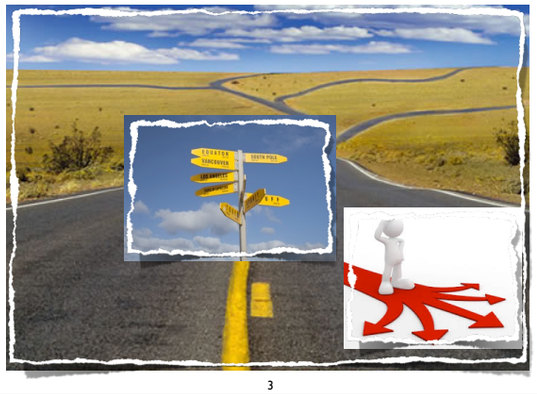
8

Procedure



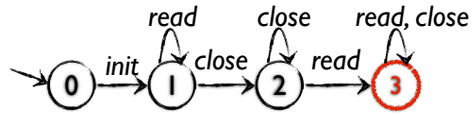
Conclusion

DSE needs guiding



3

How about a Regular Property?

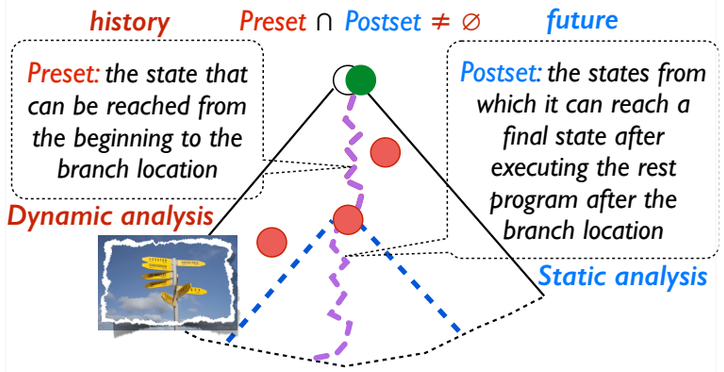


A bug property: a file is read after closed

How to guide DSE to find a program path satisfying P as soon as possible?

5

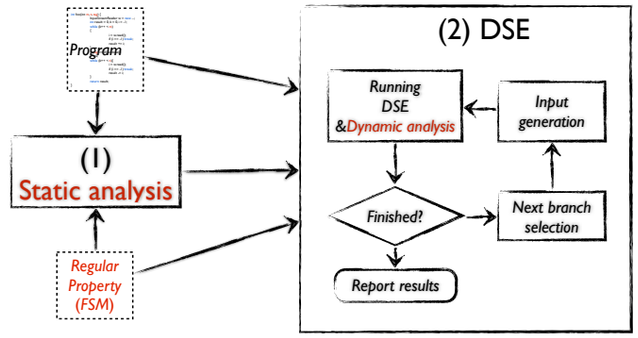
Key Idea



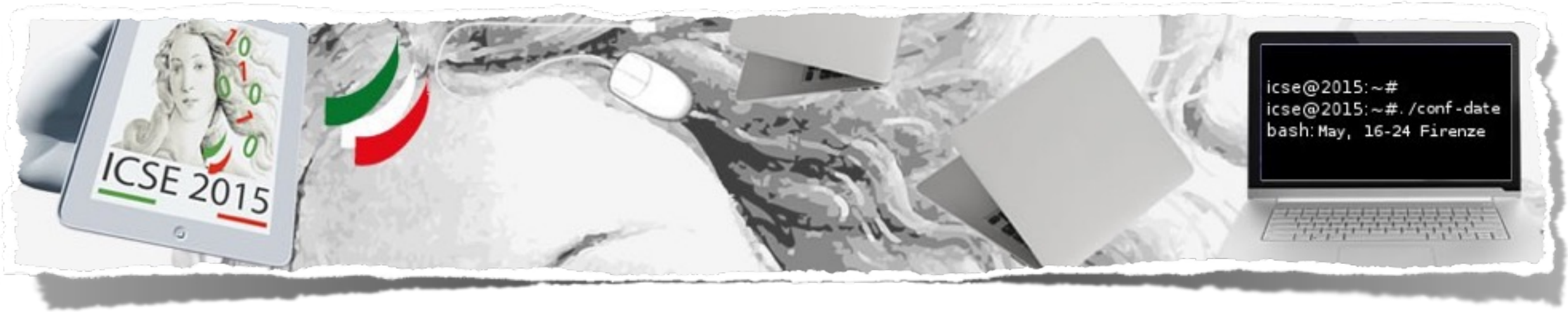
Evaluate a branch based on its history and future behaviors

8

Procedure



- Next step: multi-objects properties, combination with slicing, applications…



Thank you

Any Questions?