

Summary

Sink States:0(0×10^0)

Table 1: Sip4J Analysis Summary

Classes	Methods	States	Unreachable clauses	Unreachable states	Possible concurrent methods	Total. no. of method pairs	No. of concurrent method pairs	Percentage of concurrent methods pairs
JGFTimer	9	1	0	0	3	45	6	13
JGFInstrumentor	13	1	0	0	12	91	12	13
SOR	2	1	0	0	0	3	0	0
JGFSORBenchSizeB	2	1	0	0	0	3	0	0
JGFSORBench	8	1	0	0	1	36	1	3
Total Classes=5	34	5	0	0	16	178	19	11

Contents

1	JGFTimer	3
2	JGFInstrumentor	4
3	SOR	5
4	JGFSORBenchSizeB	6
5	JGFSORBench	7
6	Abbreviation	8
7	Annotated version of the input program generated by Sip4J	9

1 JGFTimer

Table 2: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFTimer	✓
reset	✓
start	✓
stop	✓
addops	✓
perf	✓
longprint	✓
print	✓
printperf	✓

Table 3: State Transition Matrix

	alive
alive	↑

Table 4: Methods Concurrency Matrix

	JGFTimer	reset	start	stop	addops	perf	longprint	print	printperf
JGFTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
reset	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
start	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
stop	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
addops	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
perf	⌘	⌘	⌘	⌘	⌘			⌘	
longprint	⌘	⌘	⌘	⌘	⌘			⌘	
print	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
printperf	⌘	⌘	⌘	⌘	⌘			⌘	

2 JGFInstrumentor

Table 5: Method's Satisfiability(Code Reachability Analysis)

Method	Satisfiability
JGFInstrumentor	✓
addTimer	✓
addOpsToTimer	✓
startTimer	✓
stopTimer	✓
readTimer	✓
resetTimer	✓
printTimer	✓
printperfTimer	✓
storeData	✓
retrieveData	✓
printHeader	✓
main	✓

Table 6: State Transition Matrix

	alive
alive	↑

Table 7: Methods Concurrency Matrix

	JGFInstrumentor	addTimer	addOpsToTimer	startTimer	stopTimer	readTimer	resetTimer	printTimer	printperfTimer	storeData	retrieveData	printHeader	main
JGFInstrumentor	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
addTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
addOpsToTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
startTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
stopTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
readTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
resetTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
printTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
printperfTimer	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
storeData	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
retrieveData	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
printHeader	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
main	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘

3 SOR

Table 8: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
SOR	✓
SORrun	✓

Table 9: State Transition Matrix

	alive
alive	↑

Table 10: Methods Concurrency Matrix

	SOR	SORrun
SOR	⧻	⧻
SORrun	⧻	⧻

4 JGFSORBenchSizeB

Table 11: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFSORBenchSizeB	✓
main	✓

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

	JGFSORBenchSizeB	main
JGFSORBenchSizeB	⌘	⌘
main	⌘	⌘

5 JGFSORBench

Table 14: Method's Satisfiability(Code Reachability Analysis)

Method	Satisfiability
JGFSORBench	✓
JGFrnn	✓
JGFsetsize	✓
JGFinitialise	✓
JGFkernel	✓
RandomMatrix	✓
JGFvalidate	✓
JGFtidyup	✓

Table 15: State Transition Matrix

	alive
alive	↑

Table 16: Methods Concurrency Matrix

	JGFSORBench	JGFrnn	JGFsetsize	JGFinitialise	JGFkernel	RandomMatrix	JGFvalidate	JGFtidyup
JGFSORBench	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFrnn	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFsetsize	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFinitialise	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFkernel	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
RandomMatrix	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFvalidate	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFtidyup	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘

6 Abbreviation

Table 17: Used Abbreviation

Symbol	Meaning
✓	requires clause of the method is satisfiable
✗	requires clause of the method is unsatisfiable
↑	The row-state can be transitioned to the column-state
✕	The row-state cannot be transitioned to the column-state
	The row-method can be possibly executed parallel with the column-method
⋈	The row-method cannot be executed parallel with the column-method

7 Annotated version of the input program generated by Sip4J

```
1 package outputs;
2 import edu.cmu.cs.plural.annot.*;
3
4 @ClassStates({@State(name = "alive")})
5 class JGFTimer {
6   @Perm(ensures="unique(this) in alive")
7   JGFTimer() { }
8
9   @Perm(requires="share(this) in alive",
10  ensures="share(this) in alive")
11   public void reset() {
12
13   }
14   @Perm(requires="share(this) in alive",
15  ensures="share(this) in alive")
16   public void start() {
17
18   }
19   @Perm(requires="share(this) in alive",
20  ensures="share(this) in alive")
21   public void stop() {
22
23   }
24   @Perm(requires="share(this) in alive",
25  ensures="share(this) in alive")
26   public void addops(double count) {
27
28   }
29   @Perm(requires="pure(this) in alive",
30  ensures="pure(this) in alive")
31   public double perf() {
32     return 0;
33   }
34   @Perm(requires="pure(this) in alive",
35  ensures="pure(this) in alive")
36   public void longprint() {
37
38   }
39   @Perm(requires="share(this) in alive",
40  ensures="share(this) in alive")
41   public void print() {
42
43   }
44   @Perm(requires="pure(this) in alive",
45  ensures="pure(this) in alive")
46   public void printperf() {
47
48   }
49 }
50
51 }ENDOFCLASS
52
53 @ClassStates({@State(name = "alive")})
54
55 class JGFInstrumentor {
56   @Perm(ensures="unique(this) in alive")
57   JGFInstrumentor() { }
58
59   @Perm(requires="share(this) in alive",
60  ensures="share(this) in alive")
61   void addTimer(String name) {
62
63   }
64   @Perm(requires="share(this) in alive",
65  ensures="share(this) in alive")
66   void addOpsToTimer(String name, double count) {
67
68   }
69   @Perm(requires="share(this) in alive",
70  ensures="share(this) in alive")
71   void startTimer(String name) {
72
73   }
74   @Perm(requires="share(this) in alive",
75  ensures="share(this) in alive")
76   void stopTimer(String name) {
77
78   }
79 }
```

```

79 @Perm(requires="share(this) in alive",
80 ensures="share(this) in alive")
81 double readTimer(String name) {
82     return 0;
83 }
84
85 @Perm(requires="share(this) in alive",
86 ensures="share(this) in alive")
87 void resetTimer(String name) {
88 }
89
90 @Perm(requires="share(this) in alive",
91 ensures="share(this) in alive")
92 void printTimer(String name) {
93 }
94
95 @Perm(requires="share(this) in alive",
96 ensures="share(this) in alive")
97 void printperfTimer(String name) {
98 }
99
100 @Perm(requires="share(this) in alive",
101 ensures="share(this) in alive")
102 void storeData(String name, Object obj) {
103 }
104
105 @Perm(requires="share(this) in alive",
106 ensures="share(this) in alive")
107 void retrieveData(String name, Object obj) {
108 }
109
110 void printHeader(int section, int size) {
111 }
112
113 @Perm(requires="unique(this) in alive",
114 ensures="unique(this) in alive")
115 void main(String argv[]) {
116 }
117
118 }
119
120 }ENDOFCLASS
121
122 @ClassStates({@State(name = "alive")})
123
124 class SOR {
125     @Perm(ensures="unique(this) in alive")
126     SOR() { }
127
128     @Perm(requires="share(this) in alive",
129     ensures="share(this) in alive")
130     void SORrun(int num_iterations, double G[][], double omega) {
131 }
132 }
133
134 }ENDOFCLASS
135
136 @ClassStates({@State(name = "alive")})
137
138 class JGFSORBenchSizeB {
139     @Perm(ensures="unique(this) in alive")
140     JGFSORBenchSizeB() { }
141
142     @Perm(requires="unique(this) in alive",
143     ensures="unique(this) in alive")
144     void main(String argv[]) {
145 }
146 }
147
148 }ENDOFCLASS
149
150 @ClassStates({@State(name = "alive")})
151
152 class JGFSORBench {
153     @Perm(ensures="unique(this) in alive")
154     JGFSORBench() { }
155
156     @Perm(requires="unique(this) in alive",
157     ensures="unique(this) in alive")
158     public void JGFrunk(int size) {

```

```

160 }
161 @Perm(requires="share(this) in alive",
162 ensures="share(this) in alive")
163 public void JGFsetsize(int size) {
164
165 }
166 @Perm(requires="unique(this) in alive",
167 ensures="unique(this) in alive")
168 public void JGFinitialise() {
169
170 }
171 @Perm(requires="share(this) in alive",
172 ensures="share(this) in alive")
173 public void JGFkernel() {
174
175 }
176 @Perm(requires="share(this) in alive",
177 ensures="share(this) in alive")
178 double[][] RandomMatrix(int M, int N, java.util.Random R) {
179 return null;
180
181 }
182 @Perm(requires="pure(this) in alive",
183 ensures="pure(this) in alive")
184 public void JGFvalidate() {
185
186 }
187 @Perm(requires="unique(this) in alive",
188 ensures="unique(this) in alive")
189 public void JGFtidyup() {
190
191 }
192
193 }ENDOFCLASS

```