

Summary

Sink States:0(0×10^0)

Table 1: Pulse Analysis Summary

Classes	Methods	States	Unsatisfiable Clauses	Unreachable States	Possible concurrent Methods	Total. no. of pairs	No. of concurrent pairs	Percentage of concurrent Methods
JGFEulerBenchSizeA	2	1	0	0	0	3	0	0
JGFEulerBench	7	1	0	0	1	28	1	4
Tunnel	11	1	0	0	0	66	0	0
Statevector	6	1	0	0	0	21	0	0
Vector2	3	1	0	0	2	6	3	50
JGFTimer	9	1	0	0	3	45	6	13
JGFInstrumentor	13	1	0	0	12	91	12	13
Total Classes=7	51	7	0	0	18	260	22	8

Contents

1	JGFEulerBenchSizeA	3
2	JGFEulerBench	4
3	Tunnel	5
4	Statevector	6
5	Vector2	7
6	JGFTimer	8
7	JGFInstrumentor	9
8	Abbreviation	10
9	Annotated Version of Sequential Java Program generated by Sip4j	11

1 JGFEulerBenchSizeA

Table 2: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFEulerBenchSizeA	✓
main	✓

Table 3: State Transition Matrix

	alive
alive	↑

Table 4: Methods Concurrency Matrix

	JGFEulerBenchSizeA	main
JGFEulerBenchSizeA	⌘	⌘
main	⌘	⌘

2 JGFEulerBench

Table 5: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFEulerBench	✓
JGFrunk	✓
JGFsetsize	✓
JGFinitialise	✓
JGFapplication	✓
JGFvalidate	✓
JGFtidyup	✓

Table 6: State Transition Matrix

	alive
alive	↑

Table 7: Methods Concurrency Matrix

	JGFEulerBench	JGFrunk	JGFsetsize	JGFinitialise	JGFapplication	JGFvalidate	JGFtidyup
JGFEulerBench	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFrunk	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFsetsize	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFinitialise	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFapplication	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFvalidate	⌘	⌘	⌘	⌘	⌘	⌘	⌘
JGFtidyup	⌘	⌘	⌘	⌘	⌘	⌘	⌘

3 Tunnel

Table 8: Methods Requires Clause Satisfiability

Method	Satisfiability
Tunnel	✓
initialise	✓
runiters	✓
doIteration	✓
calculateDummyCells	✓
calculateDeltaT	✓
calculateDamping	✓
calculateF	✓
calculateG	✓
calculateR	✓
calculateStateVar	✓

Table 9: State Transition Matrix

	alive
alive	↑

Table 10: Methods Concurrency Matrix

	Tunnel	initialise	runiters	doIteration	calculateDummyCells	calculateDeltaT	calculateDamping	calculateF	calculateG	calculateR	calculateStateVar
Tunnel	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
initialise	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
runiters	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
doIteration	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
calculateDummyCells	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
calculateDeltaT	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
calculateDamping	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
calculateF	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
calculateG	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
calculateR	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘
calculateStateVar	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘

4 Statevector

Table 11: Methods Requires Clause Satisfiability

Method	Satisfiability
Statevector	✓
svect	✓
amvect	✓
avect	✓
mvect	✓
smvect	✓

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

	Statevector	svect	amvect	avect	mvect	smvect
Statevector	⌘	⌘	⌘	⌘	⌘	⌘
svect	⌘	⌘	⌘	⌘	⌘	⌘
amvect	⌘	⌘	⌘	⌘	⌘	⌘
avect	⌘	⌘	⌘	⌘	⌘	⌘
mvect	⌘	⌘	⌘	⌘	⌘	⌘
smvect	⌘	⌘	⌘	⌘	⌘	⌘

5 Vector2

Table 14: Methods Requires Clause Satisfiability

Method	Satisfiability
Vector2	✓
magnitude	✓
dot	✓

Table 15: State Transition Matrix

	alive
alive	↑

Table 16: Methods Concurrency Matrix

	Vector2	magnitude	dot
Vector2	⌋	⌋	⌋
magnitude	⌋		
dot	⌋		

6 JGFTimer

Table 17: Methods Requires Clause Satisfiability

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

Table 18: State Transition Matrix

	alive
alive	↑

Table 19: Methods Concurrency Matrix

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

7 JGFInstrumentor

Table 20: Methods Requires Clause Satisfiability

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

Table 21: State Transition Matrix

	alive
alive	↑

Table 22: 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	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘	⌘

8 Abbreviation

Table 23: 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

9 Annotated Version of Sequential Java Program generated by Sip4j

```
1 package outputs;
2 import edu.cmu.cs.plural.annot.*;
3
4 @ClassStates({@State(name = "alive")})
5 class JGFEulerBenchSizeA {
6   @Perm(ensures="unique(this) in alive")
7   JGFEulerBenchSizeA() { }
8
9   @Perm(requires="unique(this) in alive",
10  ensures="unique(this) in alive")
11   void main(String argv[]) {
12   }
13
14 }ENDOFCLASS
15
16 @ClassStates({@State(name = "alive")})
17
18 class JGFEulerBench {
19   @Perm(ensures="unique(this) in alive")
20   JGFEulerBench() { }
21
22   @Perm(requires="unique(this) in alive",
23  ensures="unique(this) in alive")
24   public void JGFrun(int size) {
25   }
26   @Perm(requires="full(this) in alive",
27  ensures="full(this) in alive")
28   public void JGFsetsize(int size) {
29   }
30   @Perm(requires="unique(this) in alive",
31  ensures="unique(this) in alive")
32   public void JGFinitialise() {
33   }
34   @Perm(requires="full(this) in alive",
35  ensures="full(this) in alive")
36   public void JGFapplication() {
37   }
38   @Perm(requires="pure(this) in alive",
39  ensures="pure(this) in alive")
40   public void JGFvalidate() {
41   }
42   @Perm(requires="unique(this) in alive",
43  ensures="unique(this) in alive")
44   public void JGFtidyup() {
45   }
46
47 }ENDOFCLASS
48
49 @ClassStates({@State(name = "alive")})
50
51 class Tunnel {
52   @Perm(ensures="unique(this) in alive")
53   Tunnel() { }
54
55   @Perm(requires="unique(this) in alive",
56  ensures="unique(this) in alive")
57   public void initialise() {
58   }
59   @Perm(requires="full(this) in alive",
60  ensures="full(this) in alive")
61   public void runiters() {
62   }
63   @Perm(requires="full(this) in alive",
64  ensures="full(this) in alive")
65   void doIteration() {
66   }
67   @Perm(requires="full(this) in alive",
68  ensures="full(this) in alive")
69   private void calculateDummyCells(double localpg[][], double localtg[][], Statevector localug[][]) {
70   }
71   @Perm(requires="full(this) in alive",
72  ensures="full(this) in alive")
73   private void calculateDeltaT() {
74   }
75   @Perm(requires="full(this) in alive",
```

```

76 ensures="full(this) in alive")
77 private void calculateDamping(double localpg[][], Statevector localug[][]) {
78 }
79 @Perm(requires="full(this) in alive",
80 ensures="full(this) in alive")
81 private void calculateF(double localpg[][], double localtg[][], Statevector localug[][]) {
82 }
83 @Perm(requires="full(this) in alive",
84 ensures="full(this) in alive")
85 private void calculateG(double localpg[][], double localtg[][], Statevector localug[][]) {
86 }
87 @Perm(requires="full(this) in alive",
88 ensures="full(this) in alive")
89 private void calculateR() {
90 }
91 @Perm(requires="full(this) in alive",
92 ensures="full(this) in alive")
93 private void calculateStateVar(double localpg[][], double localtg[][], Statevector localug[][]) {
94 }
95
96 }ENDOFCLASS
97
98 @ClassStates({@State(name = "alive")})
99
100 class Statevector {
101 @Perm(ensures="unique(this) in alive")
102 Statevector() { }
103
104 @Perm(requires="full(this) in alive",
105 ensures="full(this) in alive")
106 public Statevector svect(Statevector that) {
107     return null;
108 }
109 @Perm(requires="full(this) in alive",
110 ensures="full(this) in alive")
111 public Statevector amvect(double m, Statevector that) {
112     return null;
113 }
114 @Perm(requires="full(this) in alive",
115 ensures="full(this) in alive")
116 public Statevector avect(Statevector that) {
117     return null;
118 }
119 @Perm(requires="full(this) in alive",
120 ensures="full(this) in alive")
121 public Statevector mvect(double m) {
122     return null;
123 }
124 @Perm(requires="full(this) in alive",
125 ensures="full(this) in alive")
126 public Statevector smvect(double m, Statevector that) {
127     return null;
128 }
129
130 }ENDOFCLASS
131
132 @ClassStates({@State(name = "alive")})
133
134 class Vector2 {
135 @Perm(ensures="unique(this) in alive")
136 Vector2() { }
137
138 @Perm(requires="pure(this) in alive",
139 ensures="pure(this) in alive")
140 public double magnitude() {
141     return 0;
142 }
143 @Perm(requires="pure(this) in alive",
144 ensures="pure(this) in alive")
145 public double dot(Vector2 that) {
146     return 0;
147 }
148
149 }ENDOFCLASS
150
151 @ClassStates({@State(name = "alive")})
152
153 class JGFTimer {
154 @Perm(ensures="unique(this) in alive")
155 JGFTimer() { }

```

```

157 @Perm(requires="full(this) in alive",
158 ensures="full(this) in alive")
159 public void reset() {
160 }
161 @Perm(requires="full(this) in alive",
162 ensures="full(this) in alive")
163 public void start() {
164 }
165 @Perm(requires="full(this) in alive",
166 ensures="full(this) in alive")
167 public void stop() {
168 }
169 @Perm(requires="full(this) in alive",
170 ensures="full(this) in alive")
171 public void addops(double count) {
172 }
173 @Perm(requires="pure(this) in alive",
174 ensures="pure(this) in alive")
175 public double perf() {
176     return 0;
177 }
178 @Perm(requires="pure(this) in alive",
179 ensures="pure(this) in alive")
180 public void longprint() {
181 }
182 @Perm(requires="full(this) in alive",
183 ensures="full(this) in alive")
184 public void print() {
185 }
186 @Perm(requires="pure(this) in alive",
187 ensures="pure(this) in alive")
188 public void printperf() {
189 }
190
191 }ENDOFCLASS
192
193 @ClassStates({@State(name = "alive")})
194
195 class JGFInstrumentor {
196 @Perm(ensures="unique(this) in alive")
197 JGFInstrumentor() { }
198
199 @Perm(requires="full(this) in alive",
200 ensures="full(this) in alive")
201 void addTimer(String name) {
202 }
203 @Perm(requires="full(this) in alive",
204 ensures="full(this) in alive")
205 void addOpsToTimer(String name, double count) {
206 }
207 @Perm(requires="full(this) in alive",
208 ensures="full(this) in alive")
209 void startTimer(String name) {
210 }
211 @Perm(requires="full(this) in alive",
212 ensures="full(this) in alive")
213 void stopTimer(String name) {
214 }
215 @Perm(requires="full(this) in alive",
216 ensures="full(this) in alive")
217 double readTimer(String name) {
218     return 0;
219 }
220 @Perm(requires="full(this) in alive",
221 ensures="full(this) in alive")
222 void resetTimer(String name) {
223 }
224 @Perm(requires="full(this) in alive",
225 ensures="full(this) in alive")
226 void printTimer(String name) {
227 }
228 @Perm(requires="full(this) in alive",
229 ensures="full(this) in alive")
230 void printperfTimer(String name) {
231 }
232 @Perm(requires="full(this) in alive",
233 ensures="full(this) in alive")
234 void storeData(String name, Object obj) {
235 }
236 @Perm(requires="full(this) in alive",
237 ensures="full(this) in alive")

```

```
238 void retrieveData(String name, Object obj) {  
239 }  
  
241 void printHeader(int section, int size) {  
242 }  
243 @Perm(requires="unique(this) in alive",  
244 ensures="unique(this) in alive")  
245 void main(String argv[]) {  
246 }  
  
248 }ENDOFCLASS
```