Summary

Sink States:$0(0 \times 10^0)$

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
SearchGame	3	1	0	0	0	6	0	0
Game	6	1	0	0	2	21	3	14
TransGame	10	1	0	0	1	55	1	2
JGFSearchBench	7	1	0	0	0	28	0	0
JGFInstrumentor	13	1	0	0	12	91	12	13
JGFTimer	9	1	0	0	3	45	6	13
JGFSearchBenchSizeA	2	1	0	0	0	3	0	0
Total Classes=7	50	7	0	0	18	249	22	9

Contents

1 SearchGame

Table 2: Methods Requires Clause Satisfiability

Method	Satisfiability
SearchGame	√
solve	√
ab	√

Table 3: State Transition Matrix

	alive
alive	↑

Table 4: Methods Concurrency Matrix

	SearchGame	solve	ab
SearchGame	∦	∦	∦
solve	∦	∦	∦
ab	∦	∦	∦

2 Game

Table 5: Methods Requires Clause Satisfiability

Method	Satisfiability
Game	√
wins	√
makemove	√
backmove	√
reset	√
toString	√

Table 6: State Transition Matrix

	alive
alive	↑

Table 7: Methods Concurrency Matrix

	Game	wins	makemove	backmove	reset	toString
Game	∦	∦	∦	∦	∦	∦
wins	∦	‖	∦	∦	∦	‖
makemove	∦	∦	∦	∦	∦	∦
backmove	∦	∦	∦	∦	∦	∦
reset	∦	∦	∦	∦	∦	∦
toString	∦	‖	∦	∦	∦	‖

3 TransGame

Table 8: Methods Requires Clause Satisfiability

Method	Satisfiability
TransGame	√
transpose	√
hash	√
transrestore	√
transput	√
transtore	√
emptyTT	√
hitRate	√
result	√
htstat	√

Table 9: State Transition Matrix

	alive
alive	↑

Table 10: Methods Concurrency Matrix

	TransGame	transpose	hash	transrestore	transput	transtore	emptyTT	hitRate	result	htstat
TransGame	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
transpose	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
hash	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
transrestore	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
transput	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
transtore	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
emptyTT	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
hitRate	∦	∦	∦	∦	∦	∦	∦	‖	∦	∦
result	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
htstat	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

4 JGFSearchBench

Table 11: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFSearchBench	√
JGFapplication	√
JGFsetsize	√
JGFrun	√
JGFinitialise	√
JGFvalidate	√
JGFtidyup	√

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

	JGFSearchBench	JGFapplication	JGFsetsize	JGFrun	JGFinitialise	JGFvalidate	JGFtidyup
JGFSearchBench	∦	∦	∦	∦	∦	∦	∦
JGFapplication	∦	∦	∦	∦	∦	∦	∦
JGFsetsize	∦	∦	∦	∦	∦	∦	∦
JGFrun	∦	∦	∦	∦	∦	∦	∦
JGFinitialise	∦	∦	∦	∦	∦	∦	∦
JGFvalidate	∦	∦	∦	∦	∦	∦	∦
JGFtidyup	∦	∦	∦	∦	∦	∦	∦

5 JGFInstrumentor

Table 14: Methods Requires Clause Satisfiability

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

Table 15: State Transition Matrix

	alive
alive	↑

Table 16: Methods Concurrency Matrix

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

6 JGFTimer

Table 17: Methods Requires Clause Satisfiability

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

Table 18: State Transition Matrix

	alive
alive	↑

Table 19: Methods Concurrency Matrix

	JGFTimer	addops	print	perf	reset	start	stop	longprint	printperf
JGFTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦
addops	∦	∦	∦	∦	∦	∦	∦	∦	∦
print	∦	∦	∦	∦	∦	∦	∦	∦	∦
perf	∦	∦	∦	‖	∦	∦	∦	‖	‖
reset	∦	∦	∦	∦	∦	∦	∦	∦	∦
start	∦	∦	∦	∦	∦	∦	∦	∦	∦
stop	∦	∦	∦	∦	∦	∦	∦	∦	∦
longprint	∦	∦	∦	‖	∦	∦	∦	‖	‖
printperf	∦	∦	∦	‖	∦	∦	∦	‖	‖

7 JGFSearchBenchSizeA

Table 20: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFSearchBenchSizeA	√
main	√

Table 21: State Transition Matrix

	alive
alive	↑

Table 22: Methods Concurrency Matrix

	JGFSearchBenchSizeA	main
JGFSearchBenchSizeA	∦	∦
main	∦	∦

8 Abbreviation

Table 23: Used Abbreviation

Symbol	Meaning
$\surd$	requires clause of the method is satisfiable
$\times$	requires clause of the method is unsatisfiable
$\uparrow$	The row-state can be transitioned to the column-state
$\times$	The row-state cannot be transitioned to the column-state
$\parallel$	The row-method can be possibly executed parallel with the column-method
$\nparallel$	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.*;

4  @ClassStates({@State(name = "alive")})
5  class SearchGame {
6  @Perm(ensures="unique(this) in alive")
7  SearchGame() {   }

9  @Perm(requires="full(this) in alive",
10 ensures="full(this) in alive")
11  int solve() {
12  return 0;
13 }
14 @Perm(requires="full(this) in alive",
15 ensures="full(this) in alive")
16  int ab(int alpha, int beta) {
17  return 0;
18 }

20 }ENDOFCLASS

22 @ClassStates({@State(name = "alive")})

24 class Game {
25 @Perm(ensures="unique(this) in alive")
26 Game() {   }

28 @Perm(requires="pure(this) in alive",
29 ensures="pure(this) in alive")
30 final boolean wins(int n, int h, int sidemask) {
31  return 0;
32 }
33 @Perm(requires="full(this) in alive",
34 ensures="full(this) in alive")
35  void makemove(int n) {
36 }
37 @Perm(requires="full(this) in alive",
38 ensures="full(this) in alive")
39  void backmove() {
40 }
41 @Perm(requires="full(this) in alive",
42 ensures="full(this) in alive")
43  void reset() {
44 }
45 @Perm(requires="pure(this) in alive",
46 ensures="pure(this) in alive")
47 public String toString() {
48  return null;
49 }

51 }ENDOFCLASS

53 @ClassStates({@State(name = "alive")})

55 class TransGame {
56 @Perm(ensures="unique(this) in alive")
57 TransGame() {   }

59 @Perm(requires="full(this) in alive",
60 ensures="full(this) in alive")
61  int transpose() {
62  return 0;
63 }
64 @Perm(requires="full(this) in alive",
65 ensures="full(this) in alive")
66  void hash() {
67 }
68 @Perm(requires="full(this) in alive",
69 ensures="full(this) in alive")
70  void transrestore(int score, int work) {
71 }
72 @Perm(requires="full(this) in alive",
73 ensures="full(this) in alive")
74  void transput(int score, int work) {
75 }
```

```
76  @Perm(requires="full(this) in alive",
77  ensures="full(this) in alive")
78   void transtore(int score, int work) {
79  }
80  @Perm(requires="full(this) in alive",
81  ensures="full(this) in alive")
82   void emptyTT() {
83  }
84  @Perm(requires="pure(this) in alive",
85  ensures="pure(this) in alive")
86   double hitRate() {
87   return 0;
88  }
89  @Perm(requires="full(this) in alive",
90  ensures="full(this) in alive")
91   String result() {
92   return null;
93  }
94  @Perm(requires="full(this) in alive",
95  ensures="full(this) in alive")
96   String htstat() {
97   return null;
98  }

100 }ENDOFCLASS

102 @ClassStates({@State(name = "alive")})

104 class JGFSearchBench {
105 @Perm(ensures="unique(this) in alive")
106 JGFSearchBench() {   }

108 @Perm(requires="full(this) in alive",
109 ensures="full(this) in alive")
110 public void JGFapplication() {
111 }
112 @Perm(requires="full(this) in alive",
113 ensures="full(this) in alive")
114 public void JGFsetsize(int size) {
115 }
116 @Perm(requires="unique(this) in alive",
117 ensures="unique(this) in alive")
118 public void JGFrun(int size) {
119 }
120 @Perm(requires="full(this) in alive",
121 ensures="full(this) in alive")
122 public void JGFinitialise() {
123 }
124 @Perm(requires="full(this) in alive",
125 ensures="full(this) in alive")
126 public void JGFvalidate() {
127 }
128 @Perm(requires="unique(this) in alive",
129 ensures="unique(this) in alive")
130 public void JGFtidyup() {
131 }

133 }ENDOFCLASS

135 @ClassStates({@State(name = "alive")})

137 class JGFInstrumentor {
138 @Perm(ensures="unique(this) in alive")
139 JGFInstrumentor() {   }

141 @Perm(requires="full(this) in alive",
142 ensures="full(this) in alive")
143  void addTimer(String name, String opname, int size) {
144 }
145 @Perm(requires="full(this) in alive",
146 ensures="full(this) in alive")
147  void addOpsToTimer(String name, double count) {
148 }
149 @Perm(requires="full(this) in alive",
150 ensures="full(this) in alive")
151  void printTimer(String name) {
152 }
153 @Perm(requires="full(this) in alive",
154 ensures="full(this) in alive")
155  void startTimer(String name) {
156 }
```

```
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void stopTimer(String name) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 double readTimer(String name) {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void resetTimer(String name) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void printperfTimer(String name) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void storeData(String name, Object obj) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void retrieveData(String name, Object obj) {
}

 void printHeader(int section, int size) {
}
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
 void main(String argv[]) {
}

}ENDOFCLASS

@ClassStates({@State(name = "alive")})

class JGFTimer {
@Perm(ensures="unique(this) in alive")
JGFTimer() {   }

@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void addops(double count) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void print() {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double perf() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void reset() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void start() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void stop() {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public void longprint() {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public void printperf() {
}

}ENDOFCLASS

@ClassStates({@State(name = "alive")})

class JGFSearchBenchSizeA {
@Perm(ensures="unique(this) in alive")
```

```
238 JGFSearchBenchSizeA() {   }

240 @Perm(requires="unique(this) in alive",
241 ensures="unique(this) in alive")
242  void main(String argv[]) {
243 }

245 }ENDOFCLASS
```