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	2	1	0	0	0	3	0	0
TransGame	9	1	0	0	8	45	8	18
Game	6	1	0	0	5	21	5	24
JGFSearchBench	6	1	0	0	0	21	0	0
JGFInstrumentor	3	1	0	0	0	6	0	0
JGFTimer	3	1	0	0	2	6	2	33
ConnectFourConstants	1	1	0	0	0	1	0	0
JGFSearchBenchSizeA	2	1	0	0	1	3	1	33
Total Classes=8	32	8	0	0	16	106	16	15

Contents

1 SearchGame

Table 2: Methods Requires Clause Satisfiability

Method	Satisfiability
SearchGame	√
ab	√

Table 3: State Transition Matrix

	alive
alive	↑

Table 4: Methods Concurrency Matrix

	SearchGame	ab
SearchGame	∦	∦
ab	∦	∦

2 TransGame

Table 5: Methods Requires Clause Satisfiability

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

Table 6: State Transition Matrix

	alive
alive	↑

Table 7: Methods Concurrency Matrix

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

3 Game

Table 8: Methods Requires Clause Satisfiability

Method	Satisfiability
Game	$\surd$
reset	$\surd$
wins	$\surd$
makemove	$\surd$
backmove	$\surd$
toString	$\surd$

Table 9: State Transition Matrix

	alive
alive	$\uparrow$

Table 10: Methods Concurrency Matrix

	Game	reset	wins	makemove	backmove	toString
Game	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
reset	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
wins	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
makemove	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
backmove	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
toString	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$

4 JGFSearchBench

Table 11: Methods Requires Clause Satisfiability

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

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

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

5 JGFInstrumentor

Table 14: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFInstrumentor	√
addTimer	√
printTimer	√

Table 15: State Transition Matrix

	alive
alive	↑

Table 16: Methods Concurrency Matrix

	JGFInstrumentor	addTimer	printTimer
JGFInstrumentor	∦	∦	∦
addTimer	∦	∦	∦
printTimer	∦	∦	∦

6 JGFTimer

Table 17: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFTimer	$\surd$
print	$\surd$
perf	$\surd$

Table 18: State Transition Matrix

	alive
alive	$\uparrow$

Table 19: Methods Concurrency Matrix

	JGFTimer	print	perf
JGFTimer	$\nparallel$	$\nparallel$	$\nparallel$
print	$\nparallel$	$\nparallel$	$\parallel$
perf	$\nparallel$	$\parallel$	$\parallel$

7 ConnectFourConstants

Table 20: Methods Requires Clause Satisfiability

Method	Satisfiability
ConnectFourConstants	√

Table 21: State Transition Matrix

	alive
alive	↑

8 JGFSearchBenchSizeA

Table 22: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFSearchBenchSizeA	√
main	√

Table 23: State Transition Matrix

	alive
alive	↑

Table 24: Methods Concurrency Matrix

	JGFSearchBenchSizeA	main
JGFSearchBenchSizeA	⫫	⫫
main	⫫	‖

9 Abbreviation

Table 25: 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

10 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 SearchGame {
6  @Perm(ensures="unique(this) in alive")
7  SearchGame() {   }
8
9  @Perm(requires="full(this) * full(#0) * full(#1) in alive",
10 ensures="full(this) * full(#0) * full(#1) in alive")
11  int ab(int alpha, int beta) {
12  return 0;
13 }
14
15 }ENDOFCLASS
16
17 @ClassStates({@State(name = "alive")})
18
19 class TransGame {
20 @Perm(ensures="unique(this) in alive")
21 TransGame() {   }
22
23 @Perm(requires="full(this) in alive",
24 ensures="full(this) in alive")
25  void transtore(int score, int work) {
26 }
27 @Perm(requires="full(this) in alive",
28 ensures="full(this) in alive")
29  void hash() {
30 }
31 @Perm(requires="full(this) in alive",
32 ensures="full(this) in alive")
33  void transput(int score, int work) {
34 }
35 @Perm(requires="full(this) in alive",
36 ensures="full(this) in alive")
37  void transrestore(int score, int work) {
38 }
39 @Perm(requires="pure(this) in alive",
40 ensures="pure(this) in alive")
41  double hitRate() {
42  return 0;
43 }
44 @Perm(requires="full(this) in alive",
45 ensures="full(this) in alive")
46  int transpose() {
47  return 0;
48 }
49 @Perm(requires="full(this) in alive",
50 ensures="full(this) in alive")
51  String result() {
52  return null;
53 }
54 @Perm(requires="full(this) in alive",
55 ensures="full(this) in alive")
56  String htstat() {
57  return null;
58 }
59
60 }ENDOFCLASS
61
62 @ClassStates({@State(name = "alive")})
63
64 class Game {
65 @Perm(ensures="unique(this) in alive")
66 Game() {   }
67
68 @Perm(requires="full(this) in alive",
69 ensures="full(this) in alive")
70  void reset() {
71 }
72 @Perm(requires="full(this) in alive",
73 ensures="full(this) in alive")
74 final boolean wins(int n, int h, int sidemask) {
75  return 0;
```

```
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void makemove(int n) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void backmove() {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public String toString() {
 return null;
}

}ENDOFCLASS

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

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

@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void JGFsetsize(int size) {
}
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void JGFrun(int size) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void JGFinitialise() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void JGFvalidate() {
}
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void JGFtidyup() {
}

}ENDOFCLASS

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

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

@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void addTimer(String name, String opname, int size) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void printTimer(String name) {
}

}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 print() {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double perf() {
 return 0;
}

}ENDOFCLASS
```

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

159 class ConnectFourConstants {
160 @Perm(ensures="unique(this) in alive")
161 ConnectFourConstants() {   }


164 }ENDOFCLASS

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

168 class JGFSearchBenchSizeA {
169 @Perm(ensures="unique(this) in alive")
170 JGFSearchBenchSizeA() {   }

172 @Perm(requires="none(this) in alive",
173 ensures="unique(this) in alive")
174  void main(String argv[]) {
175 }

177 }ENDOFCLASS
```