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
JGFTimer	9	1	0	0	3	45	6	13
JGFInstrumentor	13	1	0	0	12	91	12	13
JGFMolDynBenchSizeA	2	1	0	0	0	3	0	0
JGFMolDynBench	7	1	0	0	1	28	1	4
md	3	1	0	0	0	6	0	0
particle	6	1	0	0	0	21	0	0
random	3	1	0	0	0	6	0	0
Total Classes=7	43	7	0	0	16	200	19	10

Contents

1 JGFTimer

Table 2: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFTimer	$\surd$
reset	$\surd$
start	$\surd$
stop	$\surd$
addops	$\surd$
perf	$\surd$
longprint	$\surd$
print	$\surd$
printperf	$\surd$

Table 3: State Transition Matrix

	alive
alive	$\uparrow$

Table 4: Methods Concurrency Matrix

	JGFTimer	reset	start	stop	addops	perf	longprint	print	printperf
JGFTimer	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$
reset	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$
start	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$
stop	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$
addops	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$
perf	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\parallel$	$\parallel$	$\not\parallel$	$\parallel$
longprint	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\parallel$	$\parallel$	$\not\parallel$	$\parallel$
print	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$
printperf	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\not\parallel$	$\parallel$	$\parallel$	$\not\parallel$	$\parallel$

2 JGFInstrumentor

Table 5: Methods Requires Clause Satisfiability

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 JGFMolDynBenchSizeA

Table 8: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFMolDynBenchSizeA	√
main	√

Table 9: State Transition Matrix

	alive
alive	↑

Table 10: Methods Concurrency Matrix

	JGFMolDynBenchSizeA	main
JGFMolDynBenchSizeA	∦	∦
main	∦	∦

4 JGFMolDynBench

Table 11: Methods Requires Clause Satisfiability

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

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

	JGFMolDynBench	JGFrun	JGFsetsize	JGFinitialise	JGFapplication	JGFvalidate	JGFtidyup
JGFMolDynBench	∦	∦	∦	∦	∦	∦	∦
JGFrun	∦	∦	∦	∦	∦	∦	∦
JGFsetsize	∦	∦	∦	∦	∦	∦	∦
JGFinitialise	∦	∦	∦	∦	∦	∦	∦
JGFapplication	∦	∦	∦	∦	∦	∦	∦
JGFvalidate	∦	∦	∦	∦	∦	‖	∦
JGFtidyup	∦	∦	∦	∦	∦	∦	∦

5 md

Table 14: Methods Requires Clause Satisfiability

Method	Satisfiability
md	√
initialise	√
runiters	√

Table 15: State Transition Matrix

	alive
alive	↑

Table 16: Methods Concurrency Matrix

	md	initialise	runiters
md	∦	∦	∦
initialise	∦	∦	∦
runiters	∦	∦	∦

6 particle

Table 17: Methods Requires Clause Satisfiability

Method	Satisfiability
particle	√
domove	√
force	√
mkekin	√
velavg	√
dscal	√

Table 18: State Transition Matrix

	alive
alive	↑

Table 19: Methods Concurrency Matrix

	particle	domove	force	mkekin	velavg	dscal
particle	∦	∦	∦	∦	∦	∦
domove	∦	∦	∦	∦	∦	∦
force	∦	∦	∦	∦	∦	∦
mkekin	∦	∦	∦	∦	∦	∦
velavg	∦	∦	∦	∦	∦	∦
dscal	∦	∦	∦	∦	∦	∦

7 random

Table 20: Methods Requires Clause Satisfiability

Method	Satisfiability
random	√
seed	√
update	√

Table 21: State Transition Matrix

	alive
alive	↑

Table 22: Methods Concurrency Matrix

	random	seed	update
random	∦	∦	∦
seed	∦	∦	∦
update	∦	∦	∦

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

```
package outputs;
import edu.cmu.cs.plural.annot.*;

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

}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) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void addOpsToTimer(String name, double count) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void startTimer(String name) {
}
@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 printTimer(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 JGFMolDynBenchSizeA {
@Perm(ensures="unique(this) in alive")
JGFMolDynBenchSizeA() {   }

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

}ENDOFCLASS

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

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

@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 JGFsetsize(int size) {
}
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void JGFinitialise() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void JGFapplication() {
}
@Perm(requires="pure(this) in alive",
ensures="pure(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 md {
@Perm(ensures="unique(this) in alive")
md() {   }

@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void initialise() {
```

```
157 }
158 @Perm(requires="full(this) in alive",
159 ensures="full(this) in alive")
160 public void runiters() {
161 }

163 }ENDOFCLASS

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

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

171 @Perm(requires="full(this) in alive",
172 ensures="full(this) in alive")
173 public void domove(double side) {
174 }
175 @Perm(requires="full(this) in alive",
176 ensures="full(this) in alive")
177 public void force(double side, double rcoff, int mdsize, int x) {
178 }
179 @Perm(requires="full(this) in alive",
180 ensures="full(this) in alive")
181 public double mkekin(double hsq2) {
182  return 0;
183 }
184 @Perm(requires="full(this) in alive",
185 ensures="full(this) in alive")
186 public double velavg(double vaverh, double h) {
187  return 0;
188 }
189 @Perm(requires="full(this) in alive",
190 ensures="full(this) in alive")
191 public void dscal(double sc, int incx) {
192 }

194 }ENDOFCLASS

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

198 class random {
199 @Perm(ensures="unique(this) in alive")
200 random() {   }

202 @Perm(requires="full(this) in alive",
203 ensures="full(this) in alive")
204 public double seed() {
205  return 0;
206 }
207 @Perm(requires="full(this) in alive",
208 ensures="full(this) in alive")
209 public double update() {
210  return 0;
211 }

213 }ENDOFCLASS
```