Summary

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

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
SeriesTest	6	1	0	0	5	21	9	43
JGFTimer	9	1	0	0	3	45	6	13
JGFInstrumentor	13	1	0	0	12	91	12	13
JGFSeriesBenchSizeB	2	1	0	0	0	3	0	0
JGFSeriesBench	7	1	0	0	1	28	1	4
Total Classes=5	37	5	0	0	21	188	28	15

Contents

1 SeriesTest

Table 2: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
SeriesTest	√
buildTestData	√
Do	√
TrapezoidIntegrate	√
thefunction	√
freeTestData	√

Table 3: State Transition Matrix

	alive
alive	↑

Table 4: Methods Concurrency Matrix

	SeriesTest	buildTestData	Do	TrapezoidIntegrate	thefunction	freeTestData
SeriesTest	∦	∦	∦	∦	∦	∦
buildTestData	∦	∦	∦	∥	∥	∦
Do	∦	∦	∦	∥	∥	∦
TrapezoidIntegrate	∦	∥	∥	∥	∥	∥
thefunction	∦	∥	∥	∥	∥	∥
freeTestData	∦	∦	∦	∥	∥	∦

2 JGFTimer

Table 5: Method's Satisfiability(Code Reachabiity Analysis

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

Table 6: State Transition Matrix

	alive
alive	↑

Table 7: Methods Concurrency Matrix

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

3 JGFInstrumentor

Table 8: Method's Satisfiability(Code Reachabiity Analysis

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

Table 9: State Transition Matrix

	alive
alive	↑

Table 10: 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	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦

4 JGFSeriesBenchSizeB

Table 11: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFSeriesBenchSizeB	√
main	√

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

	JGFSeriesBenchSizeB	main
JGFSeriesBenchSizeB	∦	∦
main	∦	∦

5 JGFSeriesBench

Table 14: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFSeriesBench	$\surd$
JGFrun	$\surd$
JGFsetsize	$\surd$
JGFinitialise	$\surd$
JGFkernel	$\surd$
JGFvalidate	$\surd$
JGFtidyup	$\surd$

Table 15: State Transition Matrix

	alive
alive	$\uparrow$

Table 16: Methods Concurrency Matrix

	JGFSeriesBench	JGFrun	JGFsetsize	JGFinitialise	JGFkernel	JGFvalidate	JGFtidyup
JGFSeriesBench	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
JGFrun	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
JGFsetsize	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
JGFinitialise	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
JGFkernel	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
JGFvalidate	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$	$\nparallel$
JGFtidyup	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$

6 Abbreviation

Table 17: 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

7 Annotated version of the input program generated by Sip4J

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

@ClassStates({@State(name = "alive")})
class SeriesTest {
@Perm(ensures="unique(this) in alive")
SeriesTest() {   }

@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
  void buildTestData() {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  void Do() {

}

 private double TrapezoidIntegrate(double x0, double x1, int nsteps, double omegan, int select) {
 return 0;

}

 private double thefunction(double x, double omegan, int select) {
 return 0;

}
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
  void freeTestData() {

}

}ENDOFCLASS

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

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

@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void reset() {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void start() {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void stop() {

}
@Perm(requires="share(this) in alive",
ensures="share(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="share(this) in alive",
ensures="share(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="share(this) in alive",
ensures="share(this) in alive")
  void addTimer(String name) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  void addOpsToTimer(String name, double count) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  void startTimer(String name) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  void stopTimer(String name) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  double readTimer(String name) {
 return 0;

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  void resetTimer(String name) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  void printTimer(String name) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  void printperfTimer(String name) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
  void storeData(String name, Object obj) {

}
@Perm(requires="share(this) in alive",
ensures="share(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 JGFSeriesBenchSizeB {
@Perm(ensures="unique(this) in alive")
```

```
160 JGFSeriesBenchSizeB() {   }

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

166 }

168 }ENDOFCLASS

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

172 class JGFSeriesBench {
173 @Perm(ensures="unique(this) in alive")
174 JGFSeriesBench() {   }

176 @Perm(requires="unique(this) in alive",
177 ensures="unique(this) in alive")
178  public void JGFrun(int size) {

180 }
181 @Perm(requires="share(this) in alive",
182 ensures="share(this) in alive")
183  public void JGFsetsize(int size) {

185 }
186 @Perm(requires="unique(this) in alive",
187 ensures="unique(this) in alive")
188  public void JGFinitialise() {

190 }
191 @Perm(requires="share(this) in alive",
192 ensures="share(this) in alive")
193  public void JGFkernel() {

195 }
196 @Perm(requires="pure(this) in alive",
197 ensures="pure(this) in alive")
198  public void JGFvalidate() {

200 }
201 @Perm(requires="unique(this) in alive",
202 ensures="unique(this) in alive")
203  public void JGFtidyup() {

205 }

207 }ENDOFCLASS
```