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
Complex	4	1	0	0	3	10	6	60
SeqFFT	2	1	0	0	1	3	1	33
Client	2	1	0	0	0	3	0	0
FFTUtility	3	1	0	0	1	6	1	17
Total Classes=4	11	4	0	0	5	22	8	36

Contents

1 Complex

Table 2: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
Complex	$\surd$
plus	$\surd$
minus	$\surd$
times	$\surd$

Table 3: State Transition Matrix

	alive
alive	$\uparrow$

Table 4: Methods Concurrency Matrix

	Complex	plus	minus	times
Complex	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
plus	$\nparallel$	$\parallel$	$\parallel$	$\parallel$
minus	$\nparallel$	$\parallel$	$\parallel$	$\parallel$
times	$\nparallel$	$\parallel$	$\parallel$	$\parallel$

2 SeqFFT

Table 5: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
SeqFFT	$\surd$
sequentialFFT	$\surd$

Table 6: State Transition Matrix

	alive
alive	$\uparrow$

Table 7: Methods Concurrency Matrix

	SeqFFT	sequentialFFT
SeqFFT	$\nparallel$	$\nparallel$
sequentialFFT	$\nparallel$	$\parallel$

3 Client

Table 8: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
Client	√
main	√

Table 9: State Transition Matrix

	alive
alive	↑

Table 10: Methods Concurrency Matrix

	Client	main
Client	∦	∦
main	∦	∦

4 FFTUtility

Table 11: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
FFTUtility	√
createRandomComplexArray	√
show	√

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

	FFTUtility	createRandomComplexArray	show
FFTUtility	∦	∦	∦
createRandomComplexArray	∦	∦	∦
show	∦	∦	∥

5 Abbreviation

Table 14: 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

6 Annotated version of the input program generated by Sip4J

```
1  package outputs;
2  import edu.cmu.cs.plural.annot.*;
3
4  @ClassStates({@State(name = "alive")})
5  class Complex {
6  @Perm(ensures="unique(this) in alive")
7  Complex() {   }
8
9  @Perm(requires="immutable(this) in alive",
10 ensures="immutable(this) in alive")
11  public Complex plus(Complex b) {
12  return null;
13
14 }
15 @Perm(requires="immutable(this) in alive",
16 ensures="immutable(this) in alive")
17  public Complex minus(Complex b) {
18  return null;
19
20 }
21 @Perm(requires="immutable(this) in alive",
22 ensures="immutable(this) in alive")
23  public Complex times(Complex b) {
24  return null;
25
26 }
27
28 }ENDOFCLASS
29
30 @ClassStates({@State(name = "alive")})
31
32 class SeqFFT {
33 @Perm(ensures="unique(this) in alive")
34 SeqFFT() {   }
35
36 @Perm(requires="pure(this) in alive",
37 ensures="pure(this) in alive")
38   Complex[] sequentialFFT(Complex[] x) {
39  return null;
40
41 }
42
43 }ENDOFCLASS
44
45 @ClassStates({@State(name = "alive")})
46
47 class Client {
48 @Perm(ensures="unique(this) in alive")
49 Client() {   }
50
51 @Perm(requires="unique(this) in alive",
52 ensures="unique(this) in alive")
53   void main(String[] args) {
54
55 }
56
57 }ENDOFCLASS
58
59 @ClassStates({@State(name = "alive")})
60
61 class FFTUtility {
62 @Perm(ensures="unique(this) in alive")
63 FFTUtility() {   }
64
65 @Perm(requires="unique(this) in alive",
66 ensures="unique(this) in alive")
67   Complex[] createRandomComplexArray(Complex[] x, int n) {
68  return null;
69
70 }
71 @Perm(requires="pure(this) in alive",
72 ensures="pure(this) in alive")
73   void show(Complex[] x, String title) {
74
75 }
76
77 }ENDOFCLASS
```