# 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	3
2	$\mathbf{SeqFFT}$	4
3	Client	5
4	FFTUtility	6
5	Abbreviation	7
6	Annotated version of the input program generated by Sip4J	8

# 1 Complex

Table 2: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
Complex	
plus	
minus	
times	

Table 3: State Transition Matrix

	alive
alive	<b>↑</b>

Table 4: Methods Concurrency Matrix

	Complex	snld	minus	times
Complex	#	#	#	#
plus	#			
minus	#			
times	#			

# 2 SeqFFT

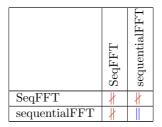
Table 5: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
SeqFFT	
sequentialFFT	

Table 6: State Transition Matrix



Table 7: Methods Concurrency Matrix



## 3 Client

 ${\it Table~8:~Method's~Satisfiability} ({\it Code~Reachability~Analysis}$ 

Method	Satisfiability
Client	
main	

Table 9: State Transition Matrix



Table 10: Methods Concurrency Matrix

	Client	main
Client	#	#
main	$\parallel$	#

## 4 FFTUtility

Table 11: Method's Satisfiability(Code Reachability Analysis

Method	Satisfiability
FFTUtility	$\checkmark$
createRandomComplexArray	$\sqrt{}$
show	$\checkmark$

Table 12: State Transition Matrix

	alive
alive	<b>↑</b>

Table 13: Methods Concurrency Matrix

	FFTUtility	create Random Complex Array	show
FFTUtility	#	#	$\parallel$
createRandomComplexArray	#	#	$\parallel$
show	#	#	

### 5 Abbreviation

Table 14: Used Abbreviation

Symbol	Meaning
	requires clause of the method is satisfiable
×	requires clause of the method is unsatisfiable
<b>↑</b>	The row-state can be transitioned to the column-state
×	The row-state cannot be transitioned to the column-state
	The row-method can be possibly executed parallel with the column-method
<b>H</b>	The row-method cannot be executed parallel with the column-method

#### 6 Annotated version of the input program generated by Sip4J

```
package outputs;
import edu.cmu.cs.plural.annot.*;
    @ClassStates({@State(name = "alive")})
    class Complex {
@Perm(ensures="unique(this) in alive")
Complex() {
}
   @Perm(requires="immutable(this) in alive",
ensures="immutable(this) in alive")
   ensures="immutable(this) in alive"
public Complex plus(Complex b) {
  return null;
    @Perm(requires="immutable(this) in alive",
    ensures="immutable(this) in alive"
public Complex minus(Complex b) {
                 'immutable(this) in alive")
    @Perm(requires="immutable(this) in alive",
    ensures="immutable(this) in alive")
public Complex times(Complex b) {
     return null;
   }
28 }ENDOFCLASS
30 @ClassStates({@State(name = "alive")})
   class SegFFT {
   @Perm(ensures="unique(this) in alive")
SeqFFT() { }
   @Perm(requires="pure(this) in alive",
   ensures="pure(this) in alive")
Complex[] sequentialFFT(Complex[] x) {
    return null;
41 }
43 }ENDOFCLASS
   @ClassStates({@State(name = "alive")})
   class Client {
   @Perm(ensures="unique(this) in alive")
Client() { }
   @Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
      void main(String[] args) {
55
57 }ENDOFCLASS
59  @ClassStates({@State(name = "alive")})
    class FFTUtility {
   @Perm(ensures="unique(this) in alive")
FFTUtility() {  }
   @Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
Complex[] createRandomComplexArray(Complex[] x, int n) {
    return null;
   OPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
void show(Complex[] x, String title) {
75 }
   }ENDOFCLASS
```