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
ArrayCollection	7	1	0	0	4	28	10	36
ObjectClass	2	1	0	0	0	3	0	0
Client	2	1	0	0	0	3	0	0
Total Classes=3	11	3	0	0	4	34	10	29

Contents

1 ArrayCollection

Table 2: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
ArrayCollection	$\surd$
printColl	$\surd$
computeStat	$\surd$
isSorted	$\surd$
findMax	$\surd$
incrColl	$\surd$
tidyupColls	$\surd$

Table 3: State Transition Matrix

	alive
alive	$\uparrow$

Table 4: Methods Concurrency Matrix

	ArrayCollection	printColl	computeStat	isSorted	findMax	incrColl	tidyupColls
ArrayCollection	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
printColl	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\nparallel$	$\nparallel$
computeStat	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\nparallel$	$\nparallel$
isSorted	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\nparallel$	$\nparallel$
findMax	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\nparallel$	$\nparallel$
incrColl	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
tidyupColls	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$

2 ObjectClass

Table 5: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
ObjectClass	$\surd$
manipulateObjects	$\surd$

Table 6: State Transition Matrix

	alive
alive	$\uparrow$

Table 7: Methods Concurrency Matrix

	ObjectClass	manipulateObjects
ObjectClass	$\nparallel$	$\nparallel$
manipulateObjects	$\nparallel$	$\nparallel$

3 Client

Table 8: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
Client	$\surd$
main	$\surd$

Table 9: State Transition Matrix

	alive
alive	$\uparrow$

Table 10: Methods Concurrency Matrix

	Client	main
Client	$\nparallel$	$\nparallel$
main	$\nparallel$	$\nparallel$

4 Abbreviation

Table 11: 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

5 Annotated version of the input program generated by Sip4J

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

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

@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public void printColl(Integer[] coll) {

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public void computeStat(Integer[] coll) {

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
  boolean isSorted(Integer[] coll) {
 return 0;

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
  Integer findMax(Integer[] coll) {
 return null;

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void incrColl(Integer[] coll) {

}
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
 public void tidyupColls(Integer[] coll) {

}

}ENDOFCLASS

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

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

@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void manipulateObjects(Client p1, Client p2) {

}

}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[] a) {

}

}ENDOFCLASS
```