TATA CONSULTANCY SERVICES
Experience certainty.

Cause Points Analysis for Effective Handling of Alarms

Tukaram Muske, Uday P. Khedker

TRDDC, Tata Consultancy Services, India
Indian Institute of Technology Bombay, India

ISSRE 2016, Ottawa, Oct 23-27

Copyright © 2013 Tata Consultancy Services Limited

Motivating Example

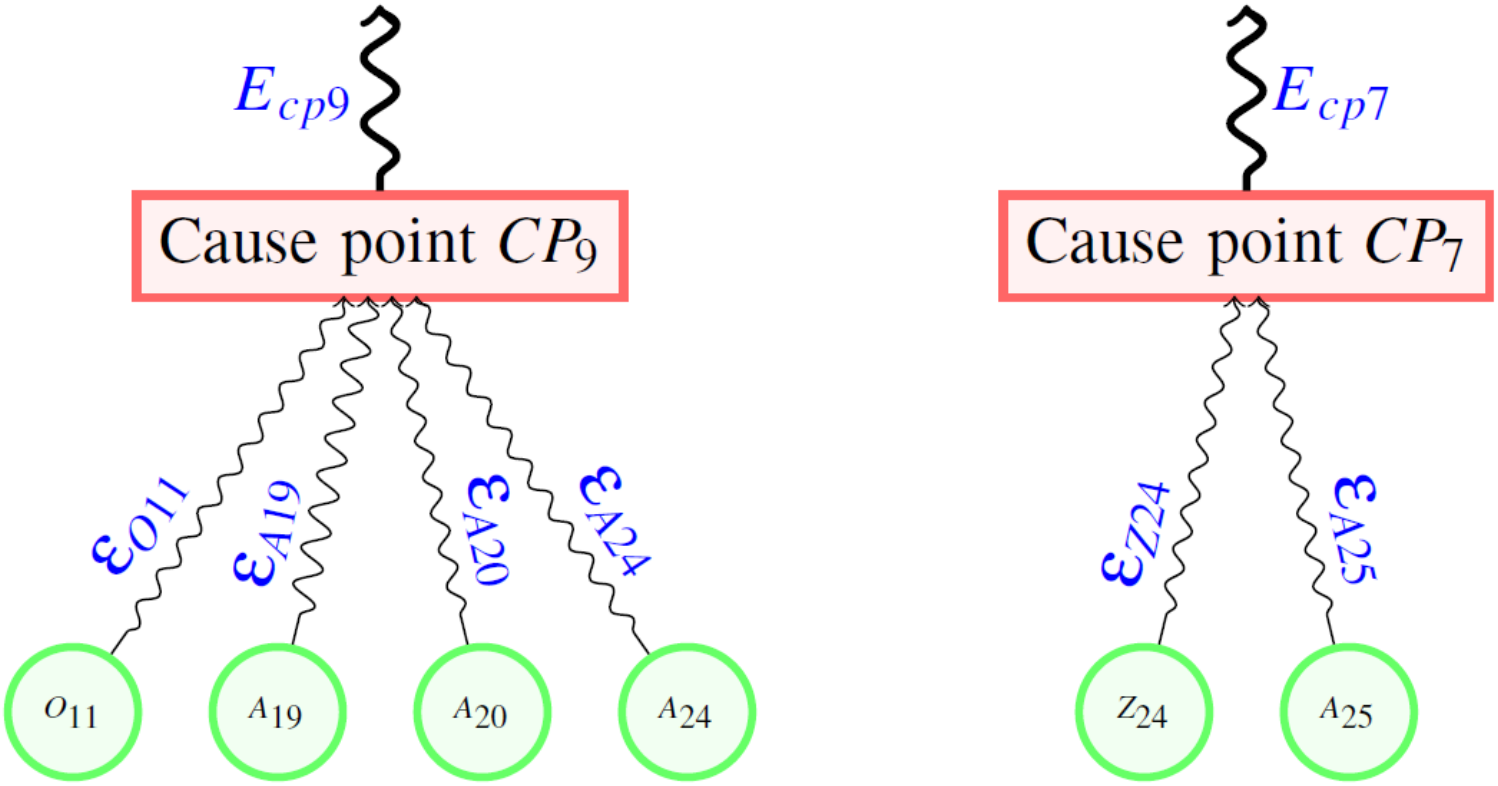
```
1  const int arr1[]={0,3,7,9,14,22,34};
2  char arr2[35], str[20], bound, tmp;
3
4   void foo(){
5   unsigned int i, j, k, length;
6    ... // some code
7   scanf("%s",str);  //Cause point CP7
8   if (i < 7 && j < i)
9      bound=arr1[i]-arr1[j]; //Cause point CP9
10
11   for(k = 0; k <= bound; k++){ //OFUF
12     f1(k);
13   }
14    length = strlen(str);
15    f2(bound, length);
16 }
17
```

```
18  void f1(int p){
19   if(nondet()) arr2[p] = 0; //AIOB
20   else arr2[p] = 1; //AIOB
21  }

22
23 void f2(int p, unsigned int q){
24 arr2[p - 1] = 100 / q; //AIOB, ZD
25 tmp = str[q]; //AIOB
26 }
```

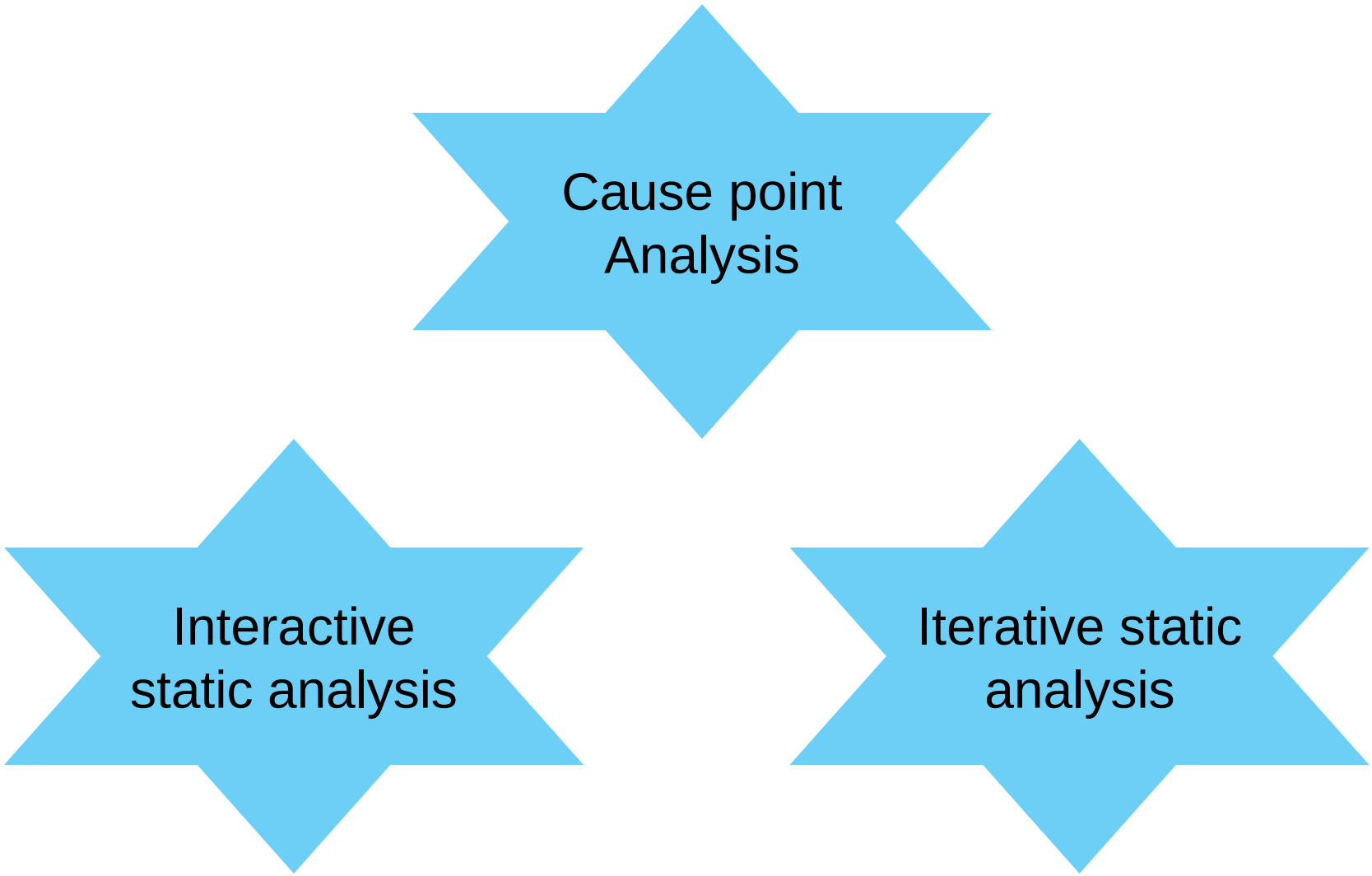
6 alarms due to
2 cause points

Manual Effort



$$\varepsilon_{orig} = ((\varepsilon_{O11}+E_{cp9})+\varepsilon_{A19}+\varepsilon_{A20}+\varepsilon_{A24}) + ((\varepsilon_{Z24}+E_{cp7})+\varepsilon_{A25})$$

Proposed Approach



Motivating Example

```
1  const int arr1[]={0,3,7,9,14,22,34};
2  char arr2[35], str[20], bound, tmp;
3
4   void foo(){
5   unsigned int i, j, k, length;
6    ... // some code
7   scanf("%s",str);  //Cause point CP7
8   if (i < 7 && j < i)
9     bound=arr1[i]-arr1[j]; //Cause point CP9
10
11  for(k = 0; k <= bound; k++){ //OFUF
12    f1(k);
13  }
14   length = strlen(str);
15   f2(bound, length);
16 }
17
```

Get values for unknowns from user

Re-analyze the code using inputs

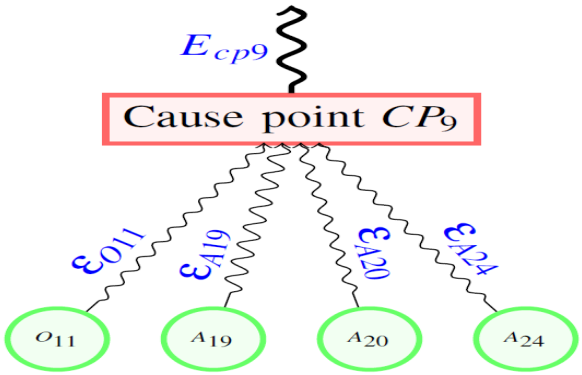
No alarm is generated

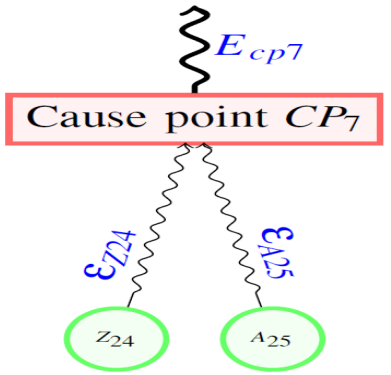
```
18  void f1(int p){
19   if(nondet()) arr2[p] = 0; //AIOB
20   else arr2[p] = 1; //AIOB
21  }

22
23 void f2(int p, unsigned int q){
24 arr2[p - 1] = 100 / q; //AIOB, ZD
25 tmp = str[q]; //AIOB
26 }
```

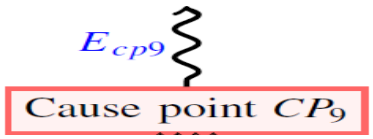
Manual Effort Reduction

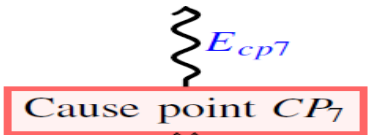
ε_{orig} =



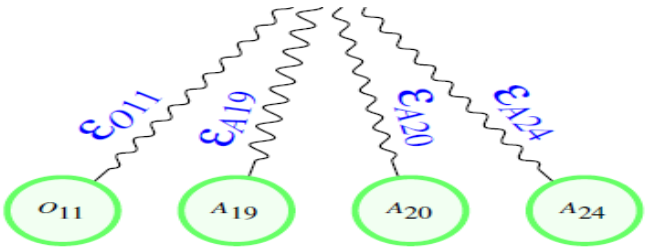


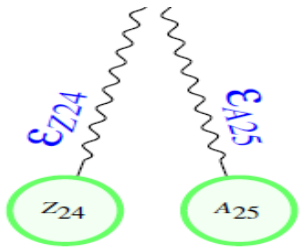
ε_{new} =



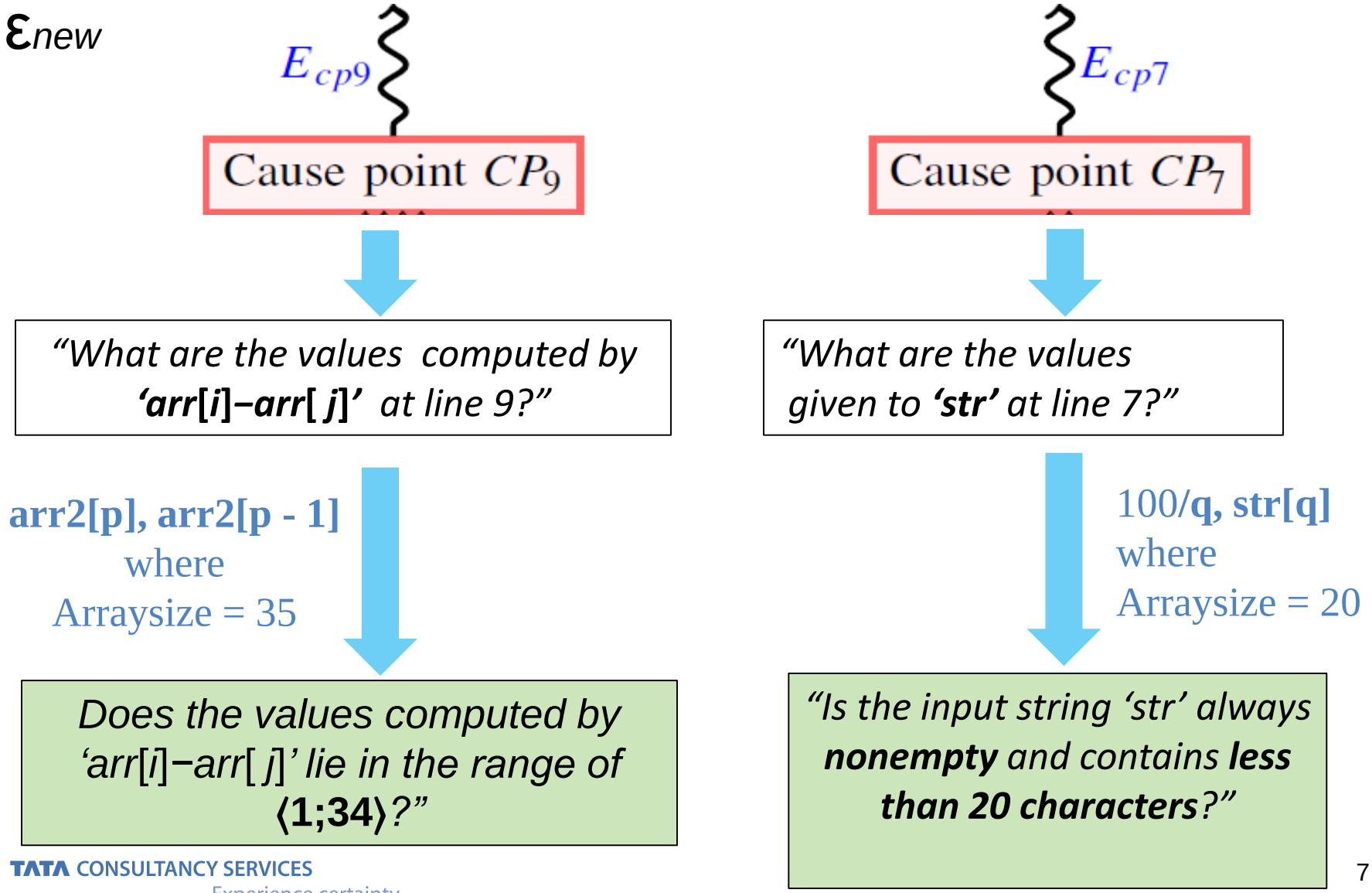


ε_{saved} =





Cause Points-specific Queries



Modeling of Cause Points

- Unknowns of various types
  - *i-unknowns* e.g. scanf("%s", str);
  - *c-unknowns* e.g. ratio * a * b + size
  - *loop-unknowns* e.g. for(i=0; i < 10 || foo(); i++){…}
  - *ds-unknowns* e.g. arr[i], pop();
  - *lib-unknowns* e.g. lib();
  - *p-unknowns*
- Cause point = Unknown + program point + unknown-type

Ranking of Cause Points

1. Unknown types

   input > library > path > loop > computational > data-structure

2. Grouping of Cause points

   A. *Lexical Similarity-based Grouping* e.g. malloc();

   B. *Proximity-based Grouping* e.g. same file or function

3. Contribution score-based ranking

   $tc\text{-}score(cp) = k * fc\text{-}score(cp) + pc\text{-}score(cp)$

Approach Benefits

Manual review effort reduction

Reviewing as many as **alarms** possible in a given time

Identifying as many as **errors** possible in a given time

Experimental Evaluation

RQ1: What is the reduction in the manual effort using the proposed approach?

RQ2: What is the contribution of cause points in generating the alarms and how are they distributed in practice?

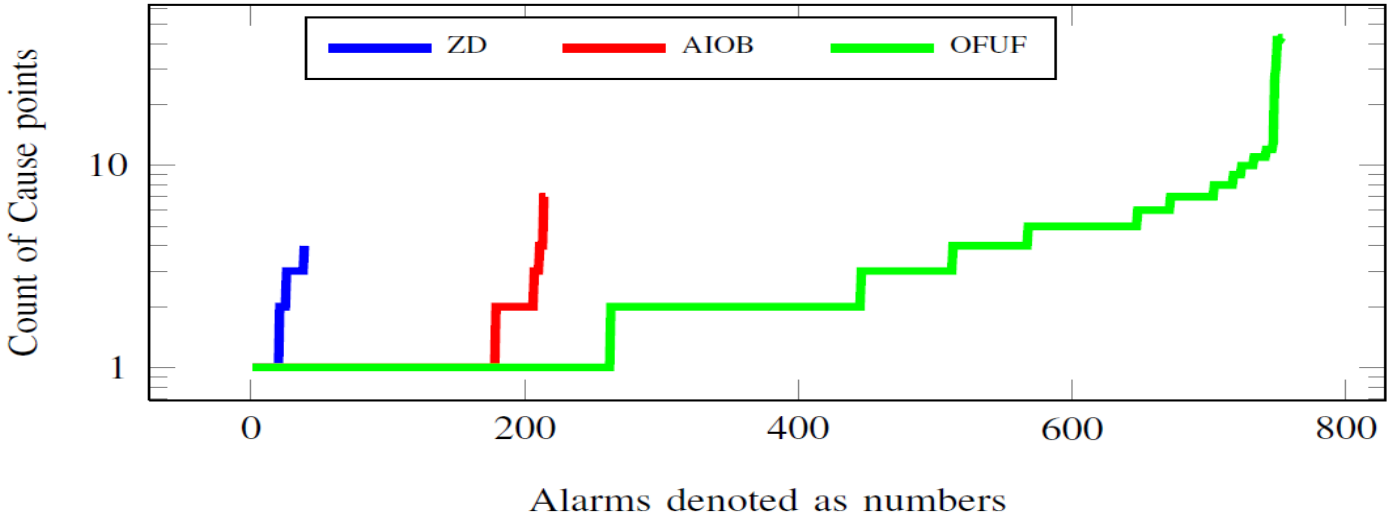
RQ3: How effective are the metrics used in the ranking of cause points?

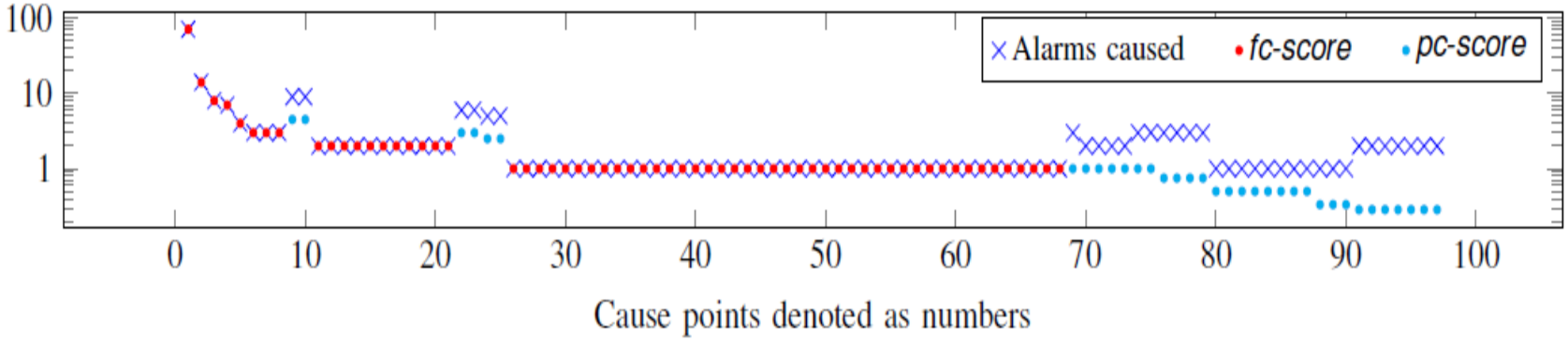
- Implementation using TCS ECA analysis framework

Effort Reduction (RQ1)

App.	Tool used	Verification Property & alarms	Reviewers		Manual Effort (Hrs)		% ε_{saved}
			Original	Proposed	ε_{orig}	ε_{new}	
A1(C)	TCS ECA	AIOB(215)	R1#	R3#	2.41	1.30	**46.05**
	TCS ECA	AIOB (215)	R6	R6	6.83	3.48	**49.04**
	TCS ECA	AIOB+OFUF+ZD(1000)	R2#	R2#	9.15	6.36	**30.49**
A2(C)	TCS ECA	AIOB(196)	R5 + R6	R5 + R6	3.29	2.53	**23.10**
A3(C)	TCS ECA	AIOB+ZD (243)*	R7	R8	12.15	7.50	**38.27**
A4(C)	TCS ECA	IDP (2000)*	R1	R2	2.74	1.24	**54.74**
A5 (C++)	Polyspace Code Prover	AIOB+ZD (85)	R4	R4	3.53	2.40	**32.01**
A6 (Java)	NPEDetector	NDP (555)*	R3#	R2#	5.68	1.58	**72.18**

Distribution/Contribution (RQ2)





Ranking Effectiveness (RQ3)

$$\text{Error cause rate (t)} = \frac{\textit{Errors uncovered in alarms caused by Cause Points of type t}}{\textit{Total alarms caused by Cause Points of type t}}$$

	Cause point (unknown) types					
	inputs	Library	Path	Loop	Computation	Data Structure
No. of cause points	7	7	6	16	10	11
Alarms caused	11	7	6	56	12	21
Errors caused	11	7	3	20	0	2
% Error cause Rate	**100**	**100**	**50**	**35**	**0**	**9**

Summary

Motivation

Simplifying manual inspections of alarms

Proposed Approach

Cause point Analysis

Iterative static analysis

Thank You!

Approach Benefits

Manual review effort reduction

Reviewing as many as **alarms** possible in a given time

Identifying as many as **errors** possible in a given time

Experimental Evaluation

RQ1: What is the reduction in the manual effort using the proposed approach?
Answer: 42 %

RQ2: What is the contribution of cause points in generating the alarms and how are they distributed in practice?
Answer: 80-20 principle in causing the alarms

RQ3: How effective are the metrics used in the ranking of cause points?
Answer: Validated our hypothesis in cause points ranking