language=C++, xleftmargin=.05numbersep=2pt, numbers=left, numberstyle=, basicstyle=, breaklines=true, keywordstyle=, showstringspaces=false, numbers=left, numbersymbol=., firstnumber=0, keepspaces=true, moredelim=[is][]++, commentstyle=, morekeywords=try, return, include, printf, extern, true, false, logic

Wonderful : A Terrific Application and Fascinating Paper

Your N. Here
Your Institution

Second Name
Second Institution

Abstract

Your Abstract Text Goes Here. Just a few facts. Whet our appetites.

1 Introduction

A paragraph of text goes here. Lots of text. Plenty of interesting text.

More fascinating text. Features[1] galore, plethora of promises.

2 Alg.

3 Alg. example

4 This is Another Section

Some embedded literal typset code might look like the following :

```
#include <iostream>
using namespace std;
main()
{
cout << "Hello world \n";
return 0;
}
```

Now we're going to cite somebody. Watch for the cite tag. Here it comes [1].

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

References

[1] EINSTEIN, A. Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]. *Annalen der Physik 322*, 10 (1905), 891–921.

Algorithm 1: Quick fix locations searching and patches generation

Input: Satisfiable program execution paths set $S_{Paths} := \{s_k | 0 \leq k \leq n, \forall\ n \geq 0\}$
Output: Refactorings set $R_{set} := \{r_j | 0 \leq j < 2\}$

```
1  W_set := {w_k | 0 ≤ k ≤ n, ∀ n ≥ 0 }; // set of working lists, k'th list
2  N_set := {n_t | 0 ≤ t ≤ n, ∀ n ≥ 0 }; // set of nodes
3  N_set := ∅; W_set := ∅; // initializing both nodes set and working list set to empty set
4  countBP :=0; countGQF :=0; // init.  counters, count buggy paths and generated fixes
5  R_set :=∅;
6  while ((Sat_paths.hasNext)) do
7      if (hasBug(s_k)) then
8          countBP := countBP + 1; // count the buggy paths
9          i := startIndex(s_k); // set the start index of the path
10         w_k := setWorkList(s_k); // set the detected buggy path into the work list
11         NLocs := 1; // number of quick fix locations
12         C := 0; // quick fix locations counter
13         // if the work list length greater than 0 else skip path
14         if (getLength(w_k) > 0) then
15             n_t := initNode(w_k); // the node at which the bug was detected
16             N_set :=N_set ∪ {n_t}; // add a node for the in-place fix
17             r_j := refact(n_t); // create a new bug refactoring
18             R_set :=R_set ∪ {r_j}; // add new refactoring to the set R
19             while (i >0 ∧ C <NLocs) do
20                 fNode := {w_k,i}; // get next node from work list located at index i
21                 if (isQuickFixNode(fNode)) then
22                     n_t+1 := fNode; // store current node
23                     N_set:=N_set ∪ {n_t+1}; // add the node for a not in-place fix
24                     setConsObject(w_k); // store constraint
25                     if (notAffectedPaths(S_Paths, n_t+1)) then
26                         pLoc := probLoc(n_t+1);
27                         putMarker(pLoc); // put new marker
28                         r_j+1 := refact(n_t+1); // create a new bug refactoring
29                         R_set :=R_set ∪ {r_j+1}; // add refactoring
30                         countGQF := countGQF + 1; // count the generated fixes
31                     end
32                     C := C + 1; // increase not in-place quick fix locations counter
33                 end
34                 i := i - 1; // go one step backwards on the path
35             end
36         end
37         k := k + 1; // get next satisfiable program execution path
38     end
39 end
```

Notes

[1] Remember to use endnotes, not footnotes!