ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

School of Computer and Communication Sciences

Dirk Beyer

BLAST: A Software Verification Tool for C programs

Problem: Software errors

Input:
- C program *p*, and
- Property *e*

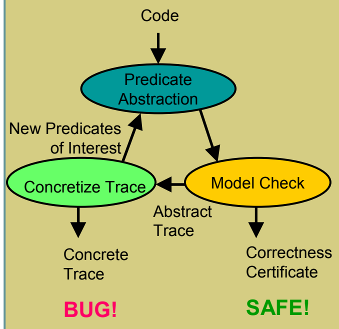
Output:
- *e* holds in *p* + certificate
- *e* is violated + example path

Solution: Model checking

Based on:
- Predicate abstraction
- Lazy abstraction
- Craig interpolation

Features:
- Test case generation
- Query language
- Memory safety

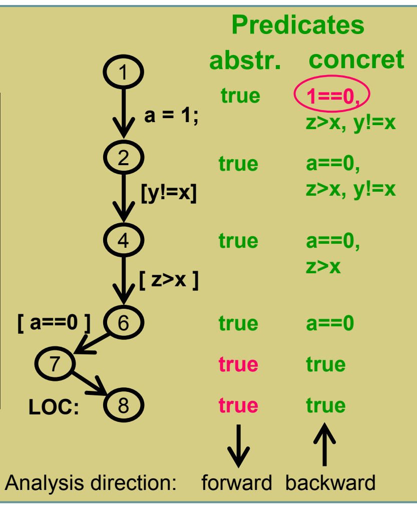
Lazy Predicate Abstraction



The procedure starts keeping track of no predicates at all. During a cycle in the model-check-refine loop, it checks the abstract program, and finds an abstract trace violating the property (if not, it terminates with SAFE). Then it concretizes that trace and finds out that the trace is infeasible (if not, it terminates with BUG). The infeasibility of the trace is due to missing predicates. Using Craig interpolation, new predicates are found and added to the abstraction, i.e., to refine the abstract program.

Example:

```
   int
   f (int x, int y, int z){
1:   int a = 1;
2:   if (y == x)
3:      y++;
4:   if (z <= x)
5:      y++;
6:   if (a == 0)
7:      LOC: // abort();
8:   return a;
   }
```



The Blast Query Language

- (Possibly Infinite-State) ***Monitor Automata*** for Reachability Queries over Program Locations
- First-Order Imperative ***Scripting Language*** for Combining Relations over Program Locations

Benefits of Two-Level Specifications

1. Separates properties from programs, while keeping a familiar syntax for writing properties
2. Treats a program as a database of facts that can be queried, and supports macros for traditional temporal-logic specifications
3. Supports the formulation of decomposition strategies for verification tasks
4. Supports the incremental maintenance of properties during program development

Example: Impact Analysis

Monitor:

```
GLOBAL int defined;
INITIAL { defined = 0; }
EVENT {
         PATTERN { j = $1; }
         ACTION { defined ++ ; }
}
FINAL { defined == 1 }  <-- else REJECT
```

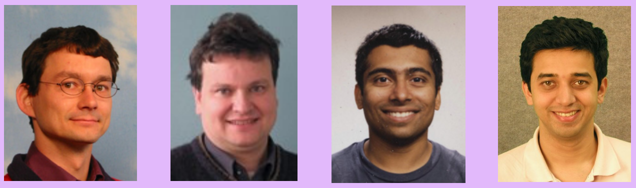
Query:

```
affected(l1,l2) :=
ACCEPT(LOC_LHS(l1,"j"),LOC_RHS(l2,"j"),monitor);
PRINT affected(l1,l2);
```

Blast Research Team

http://www.eecs.berkeley.edu/~blast

mailto:blast@eecs.berkeley.edu


Dirk Beyer Tom Henzinger Ranjit Jhala Rupak Majumdar