BRAFAR: Bidirectional Refactoring, Alignment, Fault Localization, and Repair for Programming Assignments

Thi Van Anh Dau

Limitation of previous works

- CLARA and SARFGEN
  - fail to provide feedback with unique control-flow structures
- Refactory
  - randomly applies refactoring and mutation operations to correct buggy programs

    ⟹ low success rate, complicate the original repair task
  - generate repairs for each basic block

    ⟹ produce unnecessary repairs

Limitation of previous works

Table 1: Comparison of BRAFAR against the most related feedback generation approaches.

Approach	Control-flow Repair	Minimal Control-flow Repair	Minimal Block Repair	Minimal Repair
CLARA [13]	✗	✗	✗	✗
SARFGEN [38]	✗	✗	✓	✓
Refactory [16]	✓	✗	✓	✗
BRAFAR	✓	✓	✓	✓

Motivation Examples

```
1  def search(x, seq):
2    for i,elem in enumerate(seq):
3      if seq == False:
4        return 0
5      elif x <= elem:
6        return i
7      elif i == (len(seq)-1):
8        return i+1
9      else:
10       continue
```

(a) A buggy program.

```
def search(x, seq):
  for i,elem in enumerate(seq):
    if elem < x:
      continue
    if elem == x:
      return i
    elif elem > x:
      return i
  return len(seq)
```

(b) A reference program.

```
1  def search(x, seq):
2    for i,elem in enumerate(seq):
3      if seq == False:
4        return 0
5      if elem == x:
6        return i
7      elif elem > x:
8        return i
9    return len(seq)
10
11
```

(c) *Refactroy*'s repair result.

```
def search(x, seq):
  for i,elem in enumerate(seq):
    if seq == False:
      return 0
    elif x <= elem:
      return i
    elif i == (len(seq)-1):
      return i+1
    else:
      continue
  return len(seq)
```

(d) BRAFAR's repair result.

Figure 1: Motivation example of real student submissions.

Introduction to BRAFAR

- Fully automated program repair tool
- Bidirectional refactoring algorithm: align two different control-flow structures with small modifications
- Coarse-to-fine fault localization: reduce unnecessary repairs

An overview of BRAFAR

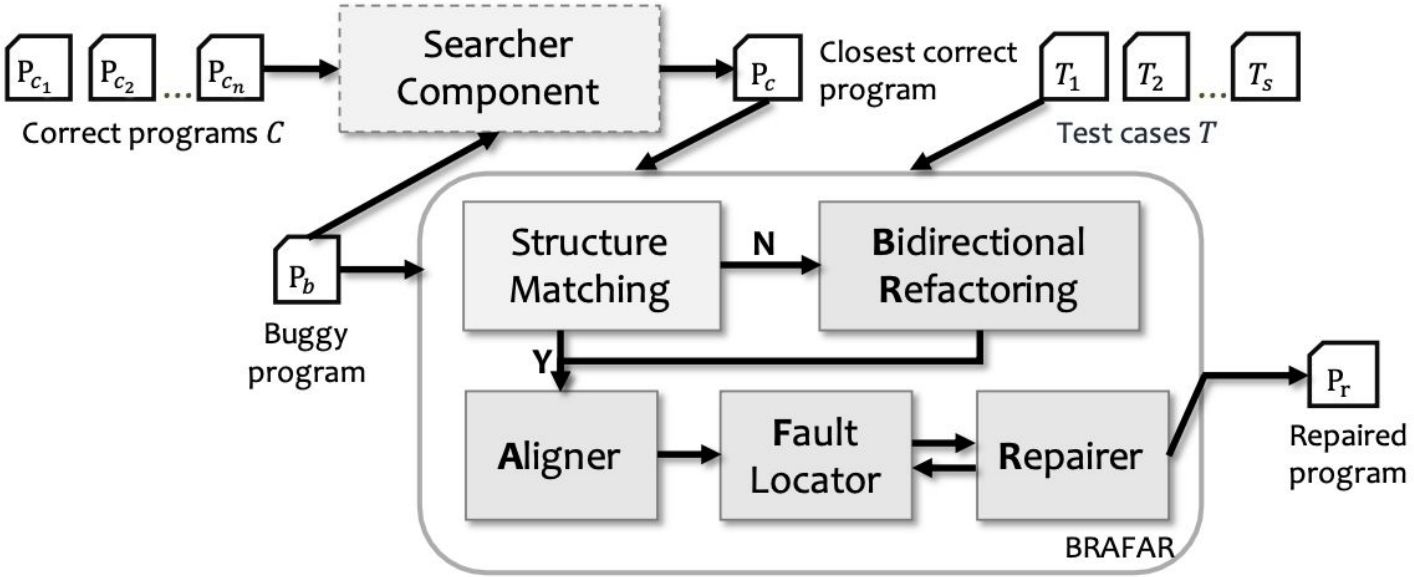
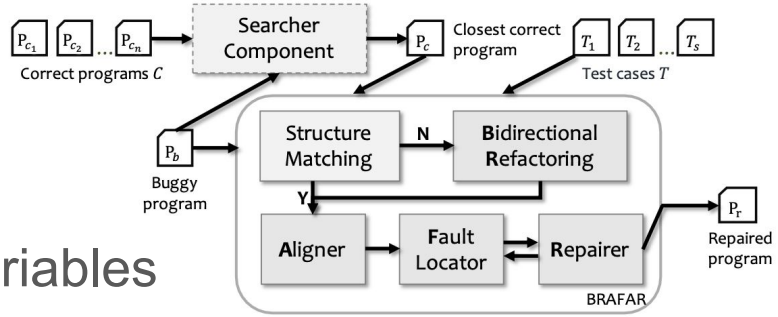


Figure 2: Overview of our approach.

An overview of BRAFAR

- Input: Correct program, Buggy program, Testcase.
- Step 1: Search the closet correct program
- Step 2: Bidirectional refactoring
  - align their control-flow structures.
- Step 3: Program repair
  - Aligner: aligns the basic blocks and variables
  - Fault Locator: locates the suspicious basic block
  - Repairer: repair until the generated program passes all testcases



Tool Demonstration

Thanks for listening