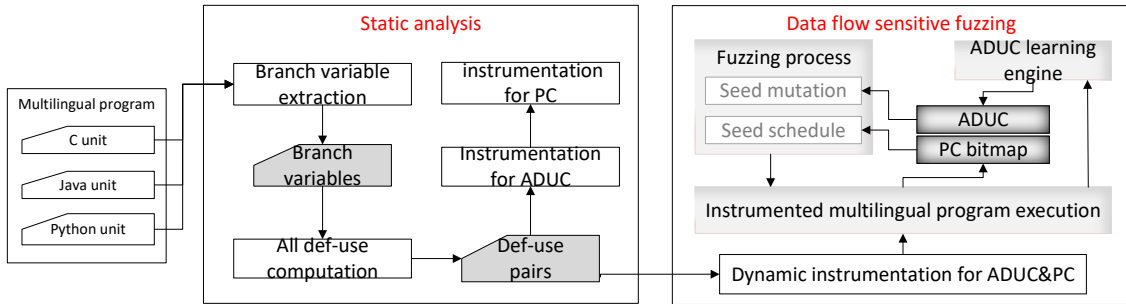


Overview



PC: path coverage, ADUC: all def-use coverage

Example

```
1 int Process (int List) {  
2   int Len = len (List);  
3   if (Len == 0) {  
4     return 0;  
5   }  
6   int Res = 1;  
7   for (int i = 0; i < Len; i++) {  
8     int Val = List[i];  
9     if (Val == 1) {  
10      Res += Val;  
11    }  
12    else if (Val == 2) {  
13      Res *= Val;  
14    }  
15    else  
16    {  
17      Res += Res/Val;  
18    }  
19  }  
20  return Res;  
21 }
```

Branch variables:

Len:
Def (s2) -> P-Use (s3)

i:
Def (s7) -> P-Use (s7)

Val:
Def (s8) -> P-Use (s9)
-> C-Use (s10)
-> P-Use (s12)
-> C-Use (s13)
-> C-Use (s17)

List0 = [1,2,3,4,...,100]
-> Val = 100

Mutation by item:

List1 = [**11**,2,3,4,...,100]
-> Val = 100

List2 = [1,**21**,3,4,...,100]
-> Val = 100

.....

List1023 = [1,2,3,4,...,**991**, 100]
-> Val = 100

List1023 = [1,2,3,4,...,99, **1001**]
-> Val = **1001**

=> Val is only determined by List[99]

List0 = [1,2,3,4,...,100]
-> Val = $\sum \text{Hash}(\text{List1}[i])$

Mutation by item:

List1 = [**11**,2,3,4,...,100]
-> Val = $\sum \text{Hash}(\text{List1}[i])$

List2 = [1,**21**,3,4,...,100]
-> Val = $\sum \text{hash}(\text{List1}[i])$

.....

List1023 = [1,2,3,4,...,**991**, 100]
-> Val = $\sum \text{Hash}(\text{List1}[i])$

List1023 = [1,2,3,4,...,99, **1001**]
-> Val = $\sum \text{Hash}(\text{List1}[i])$

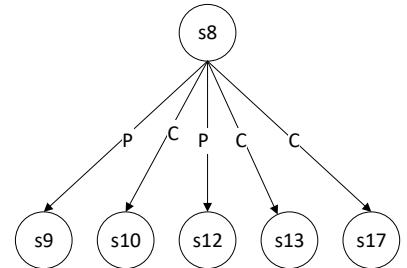
=> Val is determined by each List[i]

=> ADUC change -> seed schedule

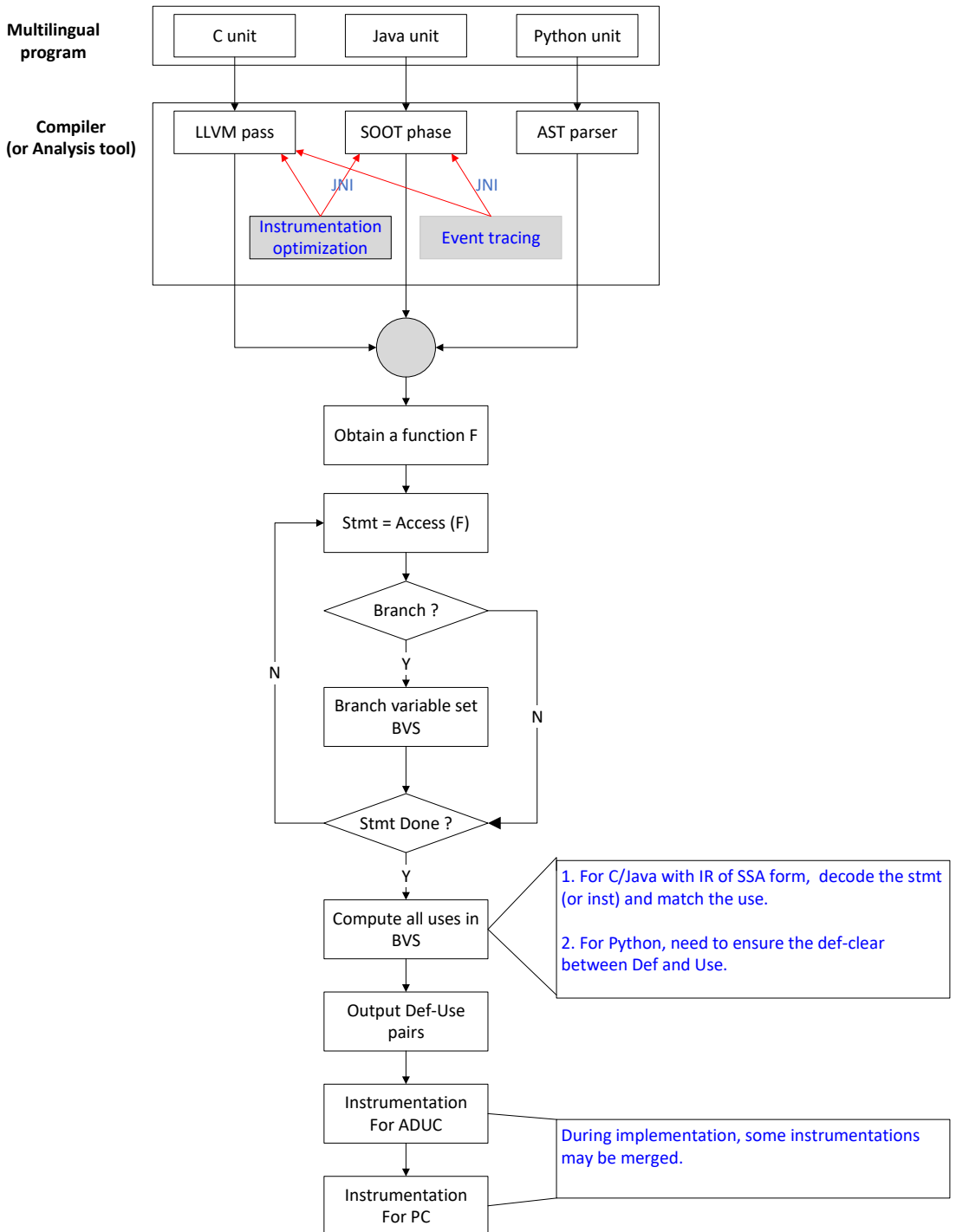
=> P-Use -> seed mutation

=> C-Use -> directed fuzzing

Value = hashed bitmap



Static phase



Runtime

