Utility Interfaces for Advanced Program Analysis in ROSE
Sriram Aananthakrishnan*, Markus Schordan†, Dan Quinlan†
*University of Utah †Lawrence Livermore National Laboratory

1 Motivation
- Utility interfaces collect useful information about the program
- Compiler analyses are very expensive and can utilize this information to manage the complexity of the analysis
- We designed and implemented two utility interfaces:
  - [1] def-use variable info for expressions
  - [2] address taken info

2 Def-Use Variable Info for Expressions
- Identify all variables that are modified by an expression
- Identify all variables used by an expression
- Identify if an expression modifies/use any memory location that does not correspond to variables in the expression

3 Interpret the Operator to Compute Def-Use Info
\[ \text{def set} : \text{set of variables defined by the expression} \]
\[ \text{use set} : \text{set of variables used by the expression} \]
\[ (x? a : b) = y + z \]
\[ \text{def set} : \{a, b\} \]
\[ \text{use set} : \{x, y, z\} \]

4 Collect Pointers and Function Calls in an Expression with Use Semantics
Flag the def/use set to indicate that the memory locations defined/used are not represented by any variables in the expression

Examples:
\[ \text{arr}[*p] = x + y \]
\[ \text{def set: } \{\text{arr}\} \]
\[ \text{use set: } \{\text{x, y, p}\} \]
\[ \text{arr}[i+1][j=0] = b[j][k=1]+c[k+1][j-1] \]
\[ \text{def set: } \{\text{arr, j, k}\} \]
\[ \text{use set: } \{\text{b, c, i, j, k}\} \]

5 Address Taken Info
- A pointer can only modify arrays or variables whose address can be taken
- Collect all variables that are of pointer, reference and array types
- Collect all variables whose address can be taken
- Variables not in the address taken set cannot be modified by pointers

6 Full C++ Support
- Handle named C++ objects, reference types, overloaded operators, exceptions and casting
- Unhandled constructs: None
- Computes exactly what variables are defined/used in expressions, pointers and function calls in expressions are also determined
- User friendly interface

7 Measurements
- Def-Use variable info and address taken info can be computed on demand in linear time

<table>
<thead>
<tr>
<th>Problem</th>
<th>LOC</th>
<th>Def-Use Info (ms)</th>
<th># Expr</th>
<th>Address Taken Info (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem34</td>
<td>107k</td>
<td>581</td>
<td>504929</td>
<td>549</td>
</tr>
<tr>
<td>Problem35</td>
<td>128k</td>
<td>1112</td>
<td>834605</td>
<td>959</td>
</tr>
<tr>
<td>Problem36</td>
<td>130k</td>
<td>1109</td>
<td>791874</td>
<td>949</td>
</tr>
</tbody>
</table>

8 Conclusion
- Def-Use info can be used to test if a function has side-effects
- A naïve pointer analysis for C can be implemented using def-use and address taken interface
- Other pointer analysis should not violate this result
- The address taken info can also help decide on the pointer analysis required for the program