Software Re-Engineering

Lecture: 15

Dr. Syed Muazzam Ali Shah
Department of Software Engineering
National University of Computer & Emerging Sciences
muazzam.ali@nu.edu.pk

Sequence [**Todays Agenda**]

Content of Lecture

Reverse Engineering – Techniques

- Lexical Analysis
- Syntactic Analysis
- Control Flow Analysis
- Data Flow Analysis
- **Program Slicing**
- Visualization
- Program metrics

Reverse Engineering – Techniques

What program slicing actually tries to achieve?

Reverse Engineering – Techniques

Program Slicing

- It is a technique that is given a potentially large program and it extracts an interesting subset of that program.
- Given a potentially large program slicing tries to **extract** an **executable subset** of this program that (potentially) affects the **values** that exist at a particular **program location**.

Reverse Engineering – Techniques

Program Slicing

- Slicing criterion = program location + variable
- So given something called a slicing criterion which is essentially a point in the program so the program location and a particular variable or maybe a particular memory location that you really care about
    - Slicing wants to extract that subset of the program that is responsible for computing the value of that variable at this particular program location.

Reverse Engineering – Techniques

Program Slicing

- Example

```
var n = readInput();
var I = 1;
var sum = 0;
var prod = 1;
while (i<=n) {
sum = sum + 1;
prod = prod * i;
i = i + 1;
}
console.log(sum);
console.log(prod);
```

Reverse Engineering – Techniques

Program Slicing

Let's use this as an example of how slicing can further reduce this full program into something that is relevant for a particular slicing criterion.

Reverse Engineering – Techniques

Program Slicing

- Example

```
var n = readInput();
var I = 1;
var sum = 0;
var prod = 1;
while (i<=n) {
sum = sum + 1;
prod = prod * i;
i = i + 1;
}
console.log(sum);
console.log(prod);
```

Slice for the value of **sum** at this statement?

Reverse Engineering – Techniques

Program Slicing

- Example

```
var n = readInput();
var I = 1;
var sum = 0;
var prod = 1;
while (i<=n) {
sum = sum + 1;
prod = prod * i;
i = i + 1;
}
console.log(sum);
console.log(prod);
```

Slice for the value of **prod** at this statement?

Reverse Engineering – Techniques

Program Slicing

- Example

```
var n = readInput();
var I = 1;
var sum = 0;
var prod = 1;
while (i<=n) {
sum = sum + 1;
i = i + 1;
prod = prod * i;
i = i + 1;
}
console.log(sum);
console.log(prod);
```

Slice for the value of n at this statement?

Reverse Engineering – Techniques

Program Slicing

- Backward vs Forward slicing

Backward slicing:

- Statements that influencing the slicing criteria.
  - Starting from a slicing criterion we're interested in the statements that influence the slicing criterion.
  - So it's basically about which other statements make us have this particular value at this particular location in the code this backward slicing.

Reverse Engineering – Techniques

Program Slicing

- Backward vs Forward slicing

Forward slicing :

- Statements that are influenced by the slicing criteria.
  - Starting at a particular slicing criterion so a particular statement with a variable that has some value we are asking the question which other statements are influenced by this slicing criterion.

Reverse Engineering – Techniques

Program Slicing

❖ Static vs Dynamic slicing

⌗ **Static slicing:**

- Static slicing is a technique that do not execute a program but rather analyze its source code in order to extract a slice of a program.

⌗ **Dynamic slicing :**

- Dynamic slicing is a technique that execute a program and then from this execution extract an Interesting fragment of the larger program.

Reverse Engineering – Techniques

Program Slicing

- Why Do wee need slicing – various applications

Debugging:

- Focus on parts of program relevant for a bug.
  - If you are interested in a particular misbehavior of your program i.e, some bug, then slicing helps you to identify which parts of the program influence what you are seeing.
  - You know that at a particular point in the program some variable has the wrong value then the question you are typically asking is which parts of the program influence the value of that variable that I'm seeing here and slicing can help you to answer that question by basically focusing your attention on those parts of the program that are relevant for this bug.

Reverse Engineering – Techniques

Program Slicing

- Why Do wee need slicing – various applications

- **Program understanding:**
  - Which statements influence this statement?
    - You may just want to know which statements are actually influencing some other statements and the values that they are computed on and slicing can help you by doing.
    - This you can start from one statement and then either slice backward or forward and see how this statement gets influenced or influences future statements.

Reverse Engineering – Techniques

Program Slicing

❖ **Why Do wee need slicing – various applications**

⌗ **Change impact analysis:**

- Which parts of a program are affected by a change? What should be retested?

  - ➢ When a program is evolving then the code is changing then at some point you want to know which parts of the code are affected by a particular change
  - ➢ For example this is interesting to know if you want to decide which parts of a program should be retested or may be tested at all after a code change and because typically you do not have to re-test everything after you change something but only those parts of the program that are affected by a change and which parts of a program are affected by a change that's something you can compute with the help of slicing.

Reverse Engineering – Techniques

Program Slicing

- Why Do wee need slicing – various applications

Parallelization:

- Determine parts of program that can be computed independently of each other.
    - If you have a program that is sequential but you would like to run it in parallel to speed up the execution then you need some way to determine which parts of the program can be computed independently of each other and because slicing tells you about the dependencies of different statements and parts of your program it can also help you to basically compute individual slices that are independent of each other and then you can run these slices in parallel because they anyway do not influence each other.

Thank You!