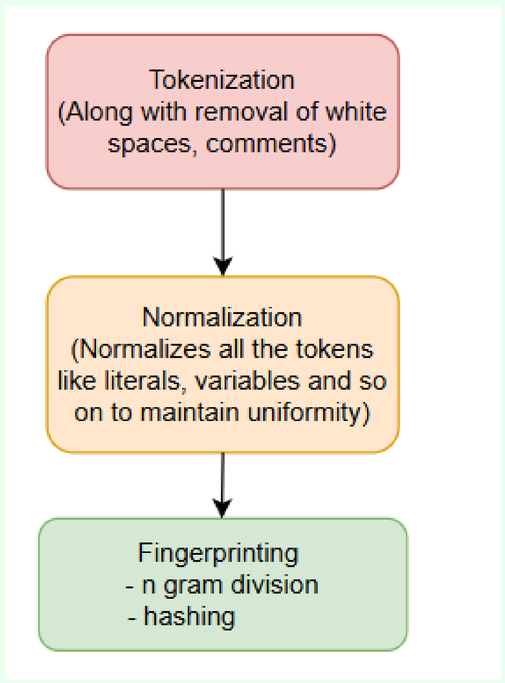
A better MOSS alternative?

Ruchika Shirsath, Dr. Uday Khedker
Department of Computer Science and Engineering, IIT Bombay

1. What is MOSS ?

- MOSS (Measure of Software Similarity) is a widely adopted plagiarism detection tool in academia, primarily used to identify similarities in source code



While effective, its reliance on surface-level analysis leaves room for circumvention

2. Why is MOSS being discussed ?

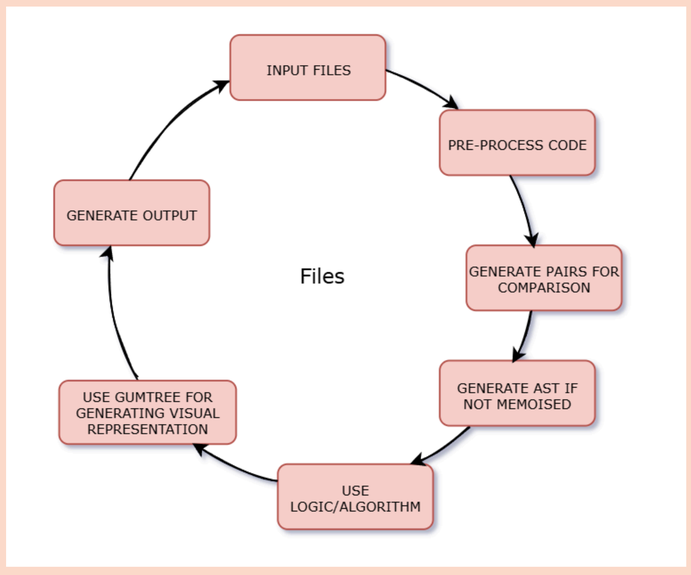
- Easy to Evade: Its reliance on n-gram analysis (in either string or tokenized form) makes it susceptible to simple obfuscation techniques.
- Limited Scope: It does not really account for the entirety of the source code and ignores deeper structural and semantic aspects.
- Need for Improvement: Loopholes exist, highlighting the need for more robust detection methods.

3. What does my tool do ?

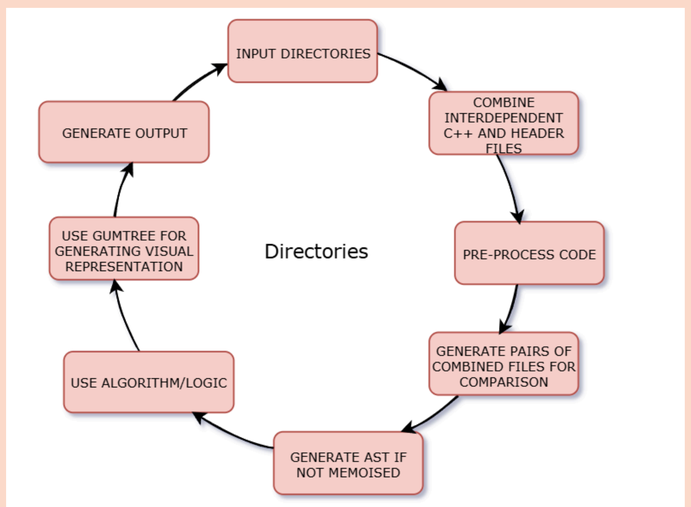
- AST-Based Analysis: The tool uses Abstract Syntax Trees (ASTs) of source codes for analysis at a structural level.
- Enhanced Similarity Detection: ASTs allow for a more comprehensive examination beyond surface-level comparisons.
- Syntax-Aware Differencing: Utilizes Gumtree, a syntax-aware diff tool, to visualize code differences effectively, highlighting changes such as insertions, additions, and deletions based on syntax structure.

4. Tool pipeline

- For files :



- For directories :



5. In what cases does my tool perform better than MOSS ?

Case	MOSS	SPDT
Copy case	93%	100%
Code Modification	91%	100%
Code Reordering	31%	84.62%
Redundant code addition	56%	100%
Changing control flow statements	--	87.5%
MOSSAD	--	75%

This comparison highlights cases where MOSS falls short in detection, while my tool demonstrates improved efficiency

6. Example

Code Reordering

Before reordering

```cpp
#include <iostream>

int multiply(int a, int b) {
    return a * b;
}

void printMessage() {
    std::cout << "Hello, ";
}

int add(int x, int y) {
    return x + y;
}

void printResult(int result) {
    std::cout << "the result is: " << result << std::endl;
}

int main() {
    int num1 = 5;
    int num2 = 3;

    printMessage();
    int product = multiply(num1, num2);
    int sum = add(num1, num2);
    printResult(product + sum);

    return 0;
}
```

After Reordering

```cpp
#include <iostream>

void printResult(int result) {
    std::cout << "the result is: " << result << std::endl;
}

void printMessage() {
    std::cout << "Hello, ";
}

int multiply(int a, int b) {
    return a * b;
}

int add(int x, int y) {
    return x + y;
}

int main() {
    int num1 = 5;
    int num2 = 3;

    printMessage();
    int product = multiply(num1, num2);
    int sum = add(num1, num2);
    printResult(product + sum);

    return 0;
}
```

Results

MOSS

Moss Results

Fri Jan 5 22:26:29 PST 2024

Options -l cc -m 10

[How to Read the Results | Tips | FAQ | Contact | Submission Scripts

File 1	File 2	Lines Matched
a1.cpp (31%)	a2.cpp (31%)	8

Any errors encountered during this query are listed below.

SPDT

```
root@LAPTOP-CVV9FMJL:~/changes# spdt t5.cpp t6.cpp
Checking files . . . .
Processing t5.cpp ...
Processing t6.cpp ...

Waiting for response . . . .
+--------+--------+----------------------+-------------------+
| File 1 | File 2 | Percentage Similarity |     Conclusion    |
+--------+--------+----------------------+-------------------+
| t5.cpp | t6.cpp |        84.62%        | Plagiarism found. |
+--------+--------+----------------------+-------------------+

Execution time: 1.1055538654327393 seconds.
root@LAPTOP-CVV9FMJL:~/changes#
```

7. MOSSAD

MOSS - Adversarial or MOSSAD is a tool meant for evading MOSS, it adds random variable declaration and initialization to tamper with the n-gram formation. MOSS fails with MOSSAD whereas my tool captures this to a good extent !