An idea of the Advanced Program Structure Formalized Method and Analysis Example of Sorting Programs

KISHIMOTO YORINORI
Department of Information Systems
Tokyo University of Information Sciences
4-1 Onari-dai, Wakaba-ku, Chiba
JAPAN
kisimoto@rsch.tuis.ac.jp

Abstract: - In the copyright protection of a program, the checking similarity and/or differences of programs are very important things. However, similarity and/or differences of programs are difficult to show clearly. So we propose the advanced program structure formalized method. In this idea, a program structure could be divided into the program structure elements and the program structure framework. In addition, program mechanisms could be divided into the flow control mechanisms and flow implement mechanisms. Thus, unique points of programs can be shown in these four categories. In this paper, we proposed the idea of the advanced program structure formalized method which divide the program structure into the program structure elements and the program structure framework, and which divide the program mechanisms into the flow control mechanisms and the flow implement mechanisms. As a result of applying this procedure to example which appeared in books, it was shown that unique points of programs can classify the program structure elements, the program structure framework and the flow implement mechanisms.

Key-Words: - Software Engineering, Formal Approach, Program structure, Copyright protection of program, similarity of programs, Static analysis of program

1 Introduction

In the copyright protection of a program, the checking similarity and/or differences of programs are very important things. However, similarity and/or differences of programs are difficult to show clearly. It had been proposed the Code Clone method that comparing same statement of programs [1]. This method could be checking similarity of two programs based on same statement of program source code. However, this method is just checking program source code statements similarity, but this method is out of consideration of the algorithms similarity. It requires to the method that can be checking unique points of algorithms for the copyright protection of a program.

So, we focused on the program structure formalized method [2][3][4]. The basic idea of this method is the program consisting of the program structure and the program mechanisms. The program structure is non-deterministic to represent basically the scheme of a program by neglecting the conditions in the selections and the iterations, and removing all program constructs which compose the mechanisms for all the conditions. The program structure represents the maximum framework of a program which is specified by the program mechanisms for the propositions.

In addition, this technology proposed a program structure could be represented by regular expression and it shows traceability analysis between a program structures represented by regular expression to it's specifications.

Based on this idea, unique points of programs derived from same specifications can be divided into a program structure's unique point and program mechanism's unique points. For example, even if programs have same program structure derived from same specifications, these programs are different because each program has different program mechanisms. In this case, unique points of program appear into program mechanisms. Thus, this method can divide program source code similarity into two elements, and it can provide compartmentalized discussion about similarity of programs.

But, it is difficult to separate program mechanisms from program source code, because, clear program mechanisms are only selections and iterations conditions, and many program mechanisms are slipping into program structure elements.
So we propose the advanced program structure formalized method. In this idea, a program structure could be divided into the program structure elements and the program structure framework. In addition, program mechanisms could be divided into the flow control mechanisms and flow implement mechanisms. Thus, unique points of programs can be shown in these four categories.

In this paper, we proposed the idea of the advanced program structure formalized method which divide the program structure into the structure elements and the structure framework, and which divide the program mechanisms into the flow control mechanisms and the flow implement mechanisms.

In addition, we proposed the method that how to implement this idea to practical field. And we show an example of application of this method and its analysis example.

### 2. Basic idea

#### 2.1 Program structure formalized method

The construct of a program can be divided into the maximum framework of a program (program structure), and a control of the flow that gives a deterministic (program mechanism). Program mechanism is process elements which define a program flow such as the conditions of selections and/or the conditions of iterations.

A program structure is the maximum framework of a non-deterministic process extracted from a program by neglecting the program mechanism. Three kinds of operators in regular expression can be used to represent the program construct sequence; "(AND) as concatenation, (OR) as selection, and Kleene Closures (\( \ast \), \( '+' \)) as iterations."

A program structure cannot be discussed about the stringency of a program. However, it can be discussed about a framework of a program, and the correspondence analysis between a program and specifications using formula transformation.

#### 2.2 Advanced Program Structure Formalized method

The program structure is non-deterministic framework of a program. And, it has the two parts.

One of the elements is program structure elements. These elements are implement specifications function and each elements corresponding to its specifications. For example, the statement "Value X and Y are swapped" appeared in specifications of sorting program. Commonly in this case, implementation of a program is "a=temp;a=b;b=temp;". Program structure elements are these program statements, and it shown in the program structure represented by regular expression is variables of regular expression.

Other element of program structure is the program structure framework. It is the non-deterministic framework of program structure. It shown in the program structure represented by regular expression are concatenations, selections (\( '+' \)), iterations (Kleene Closures \( \ast \), \( '+' \)).

Thus, a program structure can be divided into program structure elements and a program structure framework.

The program mechanisms are flow control of the program. And it has two parts. The program mechanism gives deterministic to program structure. Commonly, implementations of these in a program are conditions in selections and iterations. However, it is difficult and/or impossible to implement a program flow control just only using conditions. In program flow control, commonly it used the flags and/or the counters. So, the program mechanisms can be divided into the flow control mechanisms (conditions in selections and iterations) and the flow implement mechanisms (flags and counters).

Fig. 1 shows conceptual scheme of the advanced program structure formalized method.

![Conceptual Scheme of the Advanced Program Structure Formalized Method](image)

However, it is difficult to classify program structure elements and flow implement mechanisms. So, the program structure elements and the flow implement mechanisms were implemented program statements. Because, only checking program source code cannot decide these differences. So, we focused on the traceability analysis between a program and its specifications. Program statements corresponding to specifications assume program structure elements, and other elements assume flow implement mechanisms.
In this way, it can divide clearly into program structure elements and flow implement mechanisms.

3. Procedure
This method consists of the following Procedure.

(1) Reshaping of a program
At the conditions of selections and/or iterations, a count and a process such as a call of a subprogram may be included. In this case, a program is corrected so that the conditions of selections and/or iterations may become only a logical operation.

(2) Separation of a program structure element
A program-structure element is extracted from a source code. The conditions of iterations and selections in a program are ignored.

(3) Program structure formalization
Variables corresponding with the regular expression are assigned to each program elements. A regular expression shows a program structure based on these variables and three kinds of operators in regular expression that can be used to represent the program construct sequence; • (AND) as concatenation, + (OR) as selection, and Kleene Closures ((/)*, (/)) as iterations.

(4) Check specifications
The functional portion in a program is extracted from specifications. Variables are given to the function corresponding to a regular expression at a traceability analysis.

(5) Traceability analysis
The function of specifications and the variable name of a corresponding program structure are replaced by the variable obtained by the specifications element extraction.

(6) Program structure element and flow implement mechanism separation
At a program structure, the regular expression variables which correspond to specifications are program structure elements, and others are flow implement mechanisms.

4. An application example
It shows application example of this method.

(1) Modification of a program
At the conditions of selections and/or iterations, a count and a process such as a call of a subprogram may be included. In this case, a program is corrected so that the conditions of selections and/or iterations may become only a logical operation. Fig. 2 shows this example [5]. This example has the increment that is included in the control of the "for statement". So, this example is rewritten in a "while statement."

(2) Separation of a program structure element
A program structure element is extracted from a source code. The conditions of iterations and selections in a program are ignored.

Before
public static void sort(int[] a){
    int n=a.length;
    for(int i = 0 ; i < n ; i++){
        for(int j = n ; j > i ; j--){
            if(a[j-1] > a[j]){  int temp = a[j];
                a[j] =a[j-1];
                a[j-1] = temp;
            }
        }
    }
}

After
public static void sort(int[] a){
    int n=a.length;
    int i=0;
    while( i < n ){
        int j=n;
        while( j > i ){
            if(a[j-1] > a[j]){  int temp = a[j];
                a[j] =a[j-1];
                a[j-1] = temp;
            }j--;
        }
    i++;
}

Fig.2 An example of the program modification

public static void sort(int[] a){
    int n=a.length;
    int i=0;
    while( i < n ){
        int j=n;
        while( j > i ){
            if(a[j-1] > a[j]){  int temp = a[j];
                a[j] =a[j-1];
                a[j-1] = temp;
            }j--;
        }
    i++;
}

Fig.3 Program Structure and its variable names
### Table 1. Classifications of Sorting programs based on Advanced Program Structure Formalized Method

<table>
<thead>
<tr>
<th>Program Structure Elements</th>
<th>with Program Structure Framework</th>
<th>with Fow Implement Mechanisms</th>
<th>Algorithm Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>(((C_1 C_2 C_3 + \varepsilon)^\ast))</td>
<td>(a_i b_i (c_i (C_1 C_2 C_3 + \varepsilon) c_d) b_j^*)</td>
<td>Bubble Sort[5]</td>
<td></td>
</tr>
<tr>
<td>(((\varepsilon + \varepsilon)^\ast C_1 C_2 C_3))</td>
<td>(a_{i1} b_{i1} (c_{i1} (C_1 C_2 C_3) d_1) b_j^*)</td>
<td>Bubble Sort[6]</td>
<td></td>
</tr>
<tr>
<td>(((\varepsilon + \varepsilon)^\ast C_1 C_2 C_3))</td>
<td>(a_{i1} b_{i1} (c_{i1} (C_1 C_2 C_3) d_2) b_j^*)</td>
<td>Bubble Sort[7]</td>
<td></td>
</tr>
<tr>
<td>(((\varepsilon + \varepsilon)^\ast C_1 C_2 C_3))</td>
<td>(a_{i1} b_{i1} (c_{i1} (C_1 C_2 C_3) d_3) b_j^*)</td>
<td>Bubble Sort[8]</td>
<td></td>
</tr>
<tr>
<td>(((\varepsilon + \varepsilon)^\ast C_1 C_2 C_3))</td>
<td>(a_{i1} b_{i1} (c_{i1} (C_1 C_2 C_3) d_4) b_j^*)</td>
<td>Bubble Sort[9]</td>
<td></td>
</tr>
</tbody>
</table>

(3) **Program structure formalized**

Variables corresponding with the regular expression are assigned to each program elements. It shows in Fig. 3. And a regular expression derived from Fig. 3 as follows;

\[
a_{i1} b_{i1} (c_{i1} (C_1 C_2 C_3)^\ast) b_j^*
\]

(4) **Specifications element extract**

The functional portion in a program is extracted from specifications. Variables are given to the function corresponding to a regular expression at a traceability analysis. A function of "swap of values" derived from specifications because these examples are sort algorithm programs. So, variable name \(C\) was given to this function.

(5) **Traceability analysis**

The function of specifications and the variable name of a corresponding program structure are replaced by the variable obtained by the specifications element extraction. In this example, swap of the value in a program is implemented by three, the interim storage \((\text{temp}=a[])\) of a value, the copy \((a=b)\) of a value, and the copy \((b=\text{temp})\) of a keeping value. Then, these elements are program structure elements and these set variables as \(C_1, C_2, \text{ and } C_3\).

\[
a_{i1} b_{i1} (c_{i1} (C_1 C_2 C_3 + \varepsilon) c_d) b_j^*
\]

(6) **Program structure element and flow implement mechanism separation**

At a program structure, the regular expression variables which correspond to specifications are program structure elements, and others are flow implement mechanisms. Variables of flow implements mechanisms are small capital alphabets. The results of having applied this method to the practical examples in books are shown in Table 1 [5][6][7][8][9].

In this example, in order to discuss simply, the subscript of the following was given to the variable of the regular expression.

Increment is "I",
Decrement is "D",
Set of a value is "S".

In addition, the same statement has given the same variable.
5. Consideration
In Table 1, it shows each example of program structure elements have same program structure elements. Program structure frameworks are classified in 10 types. Flow implements mechanisms are classified in 16 types. In this example, it shows the program unique points can be separate in program structure elements, program structure framework, and flow implement mechanisms.

And in this example, it shows deference of programs derived from same specifications is gradually different by the advanced program structure formalized method. Table 2 shows this sequence.

Table 2. Sequence of the differences of programs

| 1st: | the program structure elements |
| 2nd: | the program structure framework |
| 3rd: | flow implement mechanisms |

It would appear that differences of programs derived from step of designing programs, and these steps corresponds to sequence in Table 2.

6. Conclusion
In this paper, we proposed the advanced program structure formalized method. This method is classifying program structure into the program structure elements and the program structure framework, and it classifies program mechanisms into the flow control mechanisms and the flow implement mechanisms.

We also proposed the procedure of this idea apply practical examples. As a result of applying this procedure to example which appeared in books, it was shown that unique points of programs can classify the program structure elements, the program structure framework and the flow implement mechanisms.

References: