UCL DEPARTMENT OF
COMPUTER SCIENCE

Research Note

RN/13/06

Theoretical Analysis of GP-Evolved Risk Evaluation Formulas for Spectrum Based Fault Localisation

28 February 2013

Xiaoyuan Xie[1], *Fei-Ching Kuo*[1], *Tsong Yueh Chen*[1], *Shin Yoo*[2], *Mark Harman*[2]

Affiliation: Swinburn University[1] University College London[2]

E-Mail: xxie@swin.edu.au, dkuo@swin.edu.au, tychen@swin.edu.au

shin.yoo@ucl.ac.uk, mark.harman@ucl.ac.uk

Abstract

Risk evaluation formulas convert program spectrum data from test executions into suspiciousness score, according to which statements are ranked to aid debugging activities. Designing such formulas remained largely a manual task until Genetic Programming has been recently applied: resulting formulas showed promising performance in empirical evaluation. We investigate the GP-evolved formulas theoretically and prove that GP has produced four maximal formulas that had not been known before. More interestingly, some of the newly found maximal formulas show characteristics that may seem inconsistent with human intuition. This is the first SBSE result with provable human competitiveness.

1 Introduction

As a promising automatic fault localisation technique, Spectrum-Based Fault localisation (SBFL) has been proposed and widely studied for years. SBFL uses risk evaluation formulas to convert program spectrum data collected during the test execution into relative suspiciousness score for each program statement. Statements are then ranked according to this score: the faulty statement should ideally be ranked at the top. Designing an effective risk evaluation formula has been one of the most widely studied aspects of SBFL: known formulas in the literature include Tarantula [Jones et al., 2002], Ochiai [Abreu et al., 2006], Naish1 and Naish2 [Naish et al., 2011], etc.

However, designing risk evaluation formulas has remained a manual task that involves a great deal of human intelligence and efforts. Recently, Genetic Programming (GP) has been successfully applied to automatic design of risk evaluation formulas [Yoo, 2012]. Empirical results showed that among the 30 GP-evolved formulas, six are very effective and can outperform some formulas designed by human.

While most of the studies on SBFL focused on empirical evaluation of formulas, Xie et al. developed a framework to support the theoretical analysis of risk evaluation formulas [Xie et al., 2012]. Xie et al. analysed 30 manually designed risk evaluation formulas and identified a hierarchy between formulas. The results of the theoretical analysis showed that there exist two maximal groups of formulas, namely ER1 and ER5, for single-fault scenarios.

In this paper, we apply the same theoretical framework to the 30 GP-evolved formulas reported by Yoo [Yoo, 2012]. The results show that, among these 30 GP-evolved formulas, four formulas, namely GP02, GP03, GP13, GP19, have been identified to be maximal: GP13 is proven to be equivalent to ER1, while the remaining three formulas form three distinct and newly discovered maximal groups on their own respectively. Interestingly, some GP-evolved maximal formulas display characteristics that can be considered as inconsistent with human intuition. This finding is the first report of SBSE results that are provably human competitive.

2 Backgrounds

2.1 Spectrum-Based Fault Localisation (SBFL)

SBFL uses testing results and program spectrum to do fault localisation. The testing result is whether a test case is *failed* or *passed*. While the program spectrum records the run-time profiles about various program entities for a specific test suite. The program entities could be statements, branches, paths, etc.; and the run-time information could be the binary coverage status, the execution frequency, etc. The most widely used program spectrum involves statement and its binary coverage status in a test execution [Jones et al., 2002; Abreu et al., 2006].

Consider a program $PG=<s_1, s_2, ..., s_n>$ with n statements and a test suite of m test cases $TS = \{t_1, t_2, ..., t_m\}$. Figure 1 shows the information required by SBFL. ***RE*** records all the testing results, in which p and f indicate *passed* and *failed*, respectively. Matrix ***MS*** represents the program spectrum, where the (i^{th}, j^{th}) element represents the coverage information of statement s_i, by test case t_j, with 1 indicating s_i is executed, and 0 otherwise. In fact, the j^{th} column represents the *execution slice* of t_j.

For each statement s_i, its relevant testing result can be represented as a vector $A_i=<e_f^i, e_p^i, n_f^i, n_p^i>$, where e_f^i and e_p^i represent the number of test cases in ***TS*** that execute it and return the testing result of *failure* or *pass*, respectively; n_f^i and n_p^i denote the number of test cases that do not execute it, and return the testing result of *failure* or *pass*, respectively. A risk evaluation formula R is then applied on each statement s_i to calculate a real value indicating its risk of being faulty. A commonly adopted intuition in designing risk evaluation formulas is that statements associated with more *failed* or less *passed* testing results should have higher risks. Formulas that comply with this intuition include Tarantula [Jones et al., 2002], Jaccard [Chen et al., 2002], Ochiai [Abreu et al., 2006], Naish1 and Naish2 [Naish et al., 2011] etc.

$$
\begin{array}{cc}
 & TS:(t_1 \quad t_2 \quad \ldots \quad t_m) \\
PG:\begin{pmatrix} s_1 \\ s_2 \\ \cdot \\ \cdot \\ \cdot \\ s_n \end{pmatrix} & MS:\begin{pmatrix} 1/0 & 1/0 & \ldots & 1/0 \\ 1/0 & 1/0 & \ldots & 1/0 \\ & & \cdot & \\ & & \cdot & \\ & & \cdot & \\ 1/0 & 1/0 & \ldots & 1/0 \end{pmatrix} \\
 & RE:(p/f \quad p/f \quad \ldots \quad p/f)
\end{array}
$$

Figure 1: Information for conventional SBFL

2.2 Theoretical framework

With the development of more and more risk evaluation formulas, people began to investigate their performance. Xie et al. [2012] have recently developed a theoretical framework to analysis the performance between different formulas. Since we will apply this theoretical framework in this paper, thus we briefly describe it before presenting the analysis on GP-evolved formulas.

Definition 2.1. Given a program with n statements $PG=<s_1, s_2, ..., s_n>$, a test suite of m test cases $TS=\{t_1, t_2, ..., t_m\}$, and a risk evaluation formula R, which assigns a risk value to each program statement. For each statement s_i, a vector $A_i=<e_f^i, e_p^i, n_f^i, n_p^i>$ can be constructed from TS, and $R(s_i)$ is a function of A_i. For any faulty statement s_f, following three subsets are defined.

$$
\begin{aligned}
S_B^R &= \{s_i \in S | R(s_i) > R(s_f), 1 \leq i \leq n\} \\
S_F^R &= \{s_i \in S | R(s_i) = R(s_f), 1 \leq i \leq n\} \\
S_A^R &= \{s_i \in S | R(s_i) < R(s_f), 1 \leq i \leq n\}
\end{aligned}
$$

That is, S_B^R, S_F^R and S_A^R consist of statements of which the risk values are higher than, equal to and lower than the risk value of s_f, respectively.

In practice, a tie-breaking scheme may be required to determine the order of the statements with same risk values. The theoretical analysis only investigates consistent tie-breaking schemes, which are defined as follows.

Definition 2.2. Given any two sets of statements S_1 and S_2, which contain elements having the same risk values. A tie-breaking scheme returns the ordered statement lists O_1 and O_2 for S_1 and S_2, respectively. The tie-breaking scheme is said to be consistent, if all elements common to S_1 and S_2 have the same relative order in O_1 and O_2.

The effectiveness measurement is referred to as Expense metric, which is the percentage of code that needs to be examined before the faulty statement is identified [Yoo, 2012]. Obviously, a lower Expense of formula R indicates a better performance.

Let E_1 and E_2 denote the Expenses with respect to the same faulty statement for risk evaluation formulas R_1 and R_2, respectively. We define two types of relations between R_1 and R_2 as follows.

Definition 2.3 (Better)**.** R_1 is said to be *better* than R_2 (denoted as $R_1 \rightarrow R_2$) if for any program, faulty statement s_f, test suite and consistent tie-breaking scheme, we have $E_1 \leq E_2$.

Definition 2.4 (Equivalent)**.** R_1 and R_2 are said to be *equivalent* (denoted as $R_1 \leftrightarrow R_2$), if for any program, faulty statement s_f, test suite and consistent tie-breaking scheme, we have $E_1 = E_2$.

It is obvious from the definition that $R_1 \rightarrow R_2$ means R_1 is more effective than R_2. As a reminder, if $R_1 \rightarrow R_2$ holds but $R_2 \rightarrow R_1$ does not hold, $R_1 \rightarrow R_2$ is said to be a strictly "*better*" relation. In the theoretical framework, there are several assumptions, which are listed as follows.

1. A testing oracle exists, that is, for any test case, the testing result of either *fail* or *pass*, can be decided.
2. We have the assumption of perfect bug detection that the fault can always be identified once the faulty statement is examined.
3. We exclude the omission faults, because SBFL is designed to assign risk values to the existent statements.
4. The test suite is assumed to have 100% statement coverage, that is, for any s_i, we have $e_f^i + e_p^i > 0$. Also assumed is that the test suite contains at least one passed test case and one failed test case, that is, for any s_i, we have $e_p^i + n_p^i > 0$ and $e_f^i + n_f^i > 0$.

For readers who are interested in the justifications, validity and impacts of the above assumptions, please refer to [Xie et al., 2012].

Given a test suite TS, let T denote its size, F denote the number of *failed* test cases and P denote the number of *passed* test cases. Immediately after the definitions and the above assumptions, we have $1 \leq F < T$, $1 \leq P < T$, and $P+F=T$, as well as the following lemmas.

Lemma 2.5. For any $A_i = <e_f^i, e_p^i, n_f^i, n_p^i>$, we have $e_f^i + e_p^i > 0$, $e_f^i + n_f^i = F$, $e_p^i + n_p^i = P$, $e_f^i \leq F$ and $e_p^i \leq P$.

Lemma 2.6. For any faulty statement s_f with $A_f = <e_f^f, e_p^f, n_f^f, n_p^f>$, if s_f is the only faulty statement in the program, we have $e_f^f = F$ and $n_f^f = 0$.

A necessary condition for the equivalence between two risk evaluation formulas is as follows.

Theorem 2.7. Let R_1 and R_2 be two risk evaluation formulas. If we have $S_B^{R_1} = S_B^{R_2}$, $S_F^{R_1} = S_F^{R_2}$ and $S_A^{R_1} = S_A^{R_2}$ for any program, faulty statement s_f and test suite, then $R_1 \leftrightarrow R_2$.

Xie et al. [2012] have applied the above theoretical framework on 30 manually designed formulas, identifying two groups of most effective formulas for single-fault scenario, namely the maximal groups of formulas. And our definition of "maximal formula" is as follows.

Definition 2.8. A risk evaluation formula R_1 is said to be a maximal formula of a set of formulas, if for any element R_2 of this set of formulas, $R_2 \rightarrow R_1$ implies $R_2 \leftrightarrow R_1$.

3 Theoretical analysis of GP-evolved risk evaluation formulas

3.1 Risk evaluation formulas generated by GP

Yoo [2012] has generated 30 GP-evolved formulas. As mentioned in Section 1, there are 10 out of the 30 formulas which need unreasonable additional assumptions and hence are excluded in this study. Therefore, our investigation will focus on the remaining 20 formulas (namely, GP01, GP02, GP03, GP06, GP08, GP11, GP12, GP13, GP14, GP15, GP16, GP18, GP19, GP20, GP21, GP22, GP24, GP26, GP28 and GP30). As a reminder, the following analysis is under single-fault scenario.

In [Xie et al., 2012], we have proved the maximality of formula groups ER1 (consists of Naish1 and Naish2) and ER5 (consists of Wong1, Russel & Rao and Binary) under single-fault scenario. By using the theoretical framework above, we are able to prove that among the 20 GP-evolved formulas, GP02, GP03, GP13 and GP19 are maximal formulas under single-fault scenario. More specifically, GP02, GP03 and GP19 are distinct maximal formulas to ER1 and ER5; while GP13 is equivalent to ER1. In the following discussion, the group which consists of Naish1, Naish2 and GP13 will be referred to as New_ER1. We have also proved that New_ER1 is strictly better than all the other remaining 16 GP-evolved formulas under investigation. However, since the focus of this paper is to identify the maximal (that is, maximally effective) GP-evolved formulas, we will only provide the detailed proofs for the maximality of GP02, GP03, GP13 and GP19. Definitions of the involved formulas are listed in Table 1.

Table 1: Investigated formulas

Name		Formula expression
New_ER1	Naish1	$\begin{cases} -1 & \text{if } e_f{<}F \\ P-e_p & \text{if } e_f{=}F \end{cases}$
	Naish2	$e_f - \frac{e_p}{e_p+n_p+1}$
	GP13	$e_f(1+\frac{1}{2e_p+e_f})$
ER5	Wong1	e_f
	Russel & Rao	$\frac{e_f}{e_f+n_f+e_p+n_p}$
	Binary	$\begin{cases} 0 & \text{if } e_f{<}F \\ 1 & \text{if } e_f{=}F \end{cases}$
GP02		$2(e_f+\sqrt{n_p})+\sqrt{e_p}$
GP03		$\sqrt{\lvert e_f^2-\sqrt{e_p}\rvert}$
GP19		$e_f\sqrt{\lvert e_p-e_f+n_f-n_p\rvert}$

3.2 Maximal GP-evolved risk evaluation formulas

Before presenting our proof, we need the following lemmas for ER1 (consists of Naish1 and Naish2) and GP13.

Lemma 3.1. For Naish1 and Naish2 which are shown to be equivalent [Xie et al., 2012], we have $S_B^{N1}{=}S_B^{N2}{=}X^{Op}$, $S_F^{N1}{=}S_F^{N2}{=}Y^{Op}$ and $S_A^{N1}{=}S_A^{N2}{=}Z^{Op}$, where

$$X^{Op}{=}\{s_i|e_f^i{=}F \text{ and } e_p^f{>}e_p^i, 1{\leq}i{\leq}n\} \tag{1}$$
$$Y^{Op}{=}\{s_i|e_f^i{=}F \text{ and } e_p^f{=}e_p^i, 1{\leq}i{\leq}n\} \tag{2}$$
$$Z^{Op}{=}S\backslash X^{Op}\backslash Y^{Op} \tag{3}$$

Lemma 3.2. For GP13, we have $S_B^{GP13}{=}X^{Op}$, $S_F^{GP13}{=}Y^{Op}$ and $S_A^{GP13}{=}Z^{Op}$, respectively.

Proof. Since $e_f^f{=}F$, it follows immediately from the definition of GP13 that

$$S_B^{GP13}{=}\{s_i|e_f^i(1{+}\frac{1}{2e_p^i+e_f^i}){>}F(1{+}\frac{1}{2e_p^f+F}), 1{\leq}i{\leq}n\} \tag{4}$$
$$S_F^{GP13}{=}\{s_i|e_f^i(1{+}\frac{1}{2e_p^i+e_f^i}){=}F(1{+}\frac{1}{2e_p^f+F}), 1{\leq}i{\leq}n\} \tag{5}$$

1. To prove that $S_B^{GP13} = X^{Op}$.

   (1) To prove $X^{Op}{\subseteq}S_B^{GP13}$.
   For any $s_i{\in}X^{Op}$, we have $F(1{+}\frac{1}{2e_p^i+F}){>}F(1{+}\frac{1}{2e_p^f+F})$ because $e_p^f{>}e_p^i$ and $F{>}0$. Since $e_f^i{=}F$, we have $e_f^i(1{+}\frac{1}{2e_p^i+e_f^i}){>}F(1{+}\frac{1}{2e_p^f+F})$, which implies $s_i{\in}S_B^{GP13}$. Thus, we have proved $X^{Op}{\subseteq}S_B^{GP13}$.

   (2) To prove $S_B^{GP13}{\subseteq}X^{Op}$.
   For any $s_i{\in}S_B^{GP13}$, we have $e_f^i(1{+}\frac{1}{2e_p^i+e_f^i}){>}F(1{+}\frac{1}{2e_p^f+F})$. Let us consider the following two exhaustive cases.

- Case (i) $e_f^i<F$. First, consider the sub-case that $e_f^i=0$. Then we have $e_f^i(1+\frac{1}{2e_p^i+e_f^i})=0$. It follows from the definition of S_B^{GP13} that $0>F(1+\frac{1}{2e_p^f+F})$, which is however contradictory to $F>0$ and $e_p^f\geq 0$. Thus, it is impossible to have $e_f^i=0$. Now, consider the sub-case that $0<e_f^i<F$. After re-arranging the terms, the expression $e_f^i(1+\frac{1}{2e_p^i+e_f^i})-F(1+\frac{1}{2e_p^f+F})$ becomes $(\frac{e_f^i}{2e_p^i+e_f^i}-\frac{F}{2e_p^f+F})-(F-e_f^i)$. Since $0<e_f^i<F$, this expression can be further re-written as $(\frac{1}{1+2\frac{e_p^i}{e_f^i}}-\frac{1}{1+2\frac{e_p^f}{F}})-(F-e_f^i)$. Since $\frac{e_p^i}{e_f^i}\geq 0$ and $\frac{e_p^f}{F}\geq 0$, we have $0<\frac{1}{1+2\frac{e_p^i}{e_f^i}}\leq 1$ and $0<\frac{1}{1+2\frac{e_p^f}{F}}\leq 1$. As a consequence, we have $(\frac{1}{1+2\frac{e_p^i}{e_f^i}}-\frac{1}{1+2\frac{e_p^f}{F}})<1$. Since both F and e_f^i are positive and non-negative integers, respectively, $e_f^i<F$ implies $(F-e_f^i)\geq 1$. Thus, we have $(\frac{1}{1+2\frac{e_p^i}{e_f^i}}-\frac{1}{1+2\frac{e_p^f}{F}})-(F-e_f^i)<0$, which however is contradictory to $e_f^i(1+\frac{1}{2e_p^i+e_f^i})>F(1+\frac{1}{2e_p^f+F})$. Therefore, it is impossible to have $0<e_f^i<F$. Therefore, we have proved that if $s_i\in S_B^{GP13}$, we cannot have $e_f^i<F$.
- Case (ii) $e_f^i=F$. Assume further $e_p^i\geq e_p^f$. Obviously, we have $F(1+\frac{1}{2e_p^i+F})\leq F(1+\frac{1}{2e_p^f+F})$, which can be re-written as $e_f^i(1+\frac{1}{2e_p^i+e_f^i})\leq F(1+\frac{1}{2e_p^f+F})$. However, this is contradictory to $F(1+\frac{1}{2e_p^i+F})>F(1+\frac{1}{2e_p^f+F})$. Thus, the only possible case is $e_p^f>e_p^i$.

Therefore, we have proved that if $s_i\in S_B^{GP13}$, then $e_f^i=F$ and $e_p^f>e_p^i$, which imply $s_i\in X^{Op}$. Therefore, $S_B^{GP13}\subseteq X^{Op}$.

In conclusion, we have proved $X^{Op}\subseteq S_B^{GP13}$ and $S_B^{GP13}\subseteq X^{Op}$. Therefore, $S_B^{GP13}=X^{Op}$.

2. To prove that $S_F^{GP13}=Y^{Op}$.

(1) To prove $Y^{Op}\subseteq S_F^{GP13}$.
For any $s_i\in Y^{Op}$, we have $e_f^i(1+\frac{1}{2e_p^i+e_f^i})=F(1+\frac{1}{2e_p^f+F})$ because $e_f^i=F$ and $e_p^f=e_p^i$. After the definition of S_F^{GP13}, $s_i\in S_F^{GP13}$. Thus, we have proved $Y^{Op}\subseteq S_F^{GP13}$.

(2) To prove $S_F^{GP13}\subseteq Y^{Op}$.
For any $s_i\in S_F^{GP13}$, we have $e_f^i(1+\frac{1}{2e_p^i+e_f^i})=F(1+\frac{1}{2e_p^f+F})$. Let us consider the following two exhaustive cases.

- Case (i) $e_f^i<F$. First, consider the sub-case that $e_f^i=0$. Then we have $e_f^i(1+\frac{1}{2e_p^i+e_f^i})=0$. It follows from the definition of S_F^{GP13} that $0=F(1+\frac{1}{2e_p^f+F})$, which is however contradictory to $F>0$ and $e_p^f\geq 0$. Thus, it is impossible to have $e_f^i=0$. Now, consider the sub-case that $0<e_f^i<F$. Similar to the above proof of $S_B^{GP13}\subseteq X^{Op}$, we can prove that $(\frac{1}{1+2\frac{e_p^i}{e_f^i}}-\frac{1}{1+2\frac{e_p^f}{F}})<(F-e_f^i)$, which is however contradictory to $e_f^i(1+\frac{1}{2e_p^i+e_f^i})=F(1+\frac{1}{2e_p^f+F})$. Therefore, it is impossible to have $0<e_f^i<F$. Therefore, we have proved that if $s_i\in S_F^{GP13}$, then we cannot have $e_f^i<F$.
- Case (ii) $e_f^i=F$. Assume further $e_p^i\neq e_p^f$. Obviously, we have $F(1+\frac{1}{2e_p^i+F})\neq F(1+\frac{1}{2e_p^f+F})$, which can be re-written as $e_f^i(1+\frac{1}{2e_p^i+e_f^i})\neq F(1+\frac{1}{2e_p^f+F})$. However, this is contradictory to $e_f^i(1+\frac{1}{2e_p^i+e_f^i})=F(1+\frac{1}{2e_p^f+F})$. Thus, the only possible case is $e_p^f=e_p^i$.

We have proved that if $s_i\in S_F^{GP13}$, then $e_f^i=F$ and $e_p^f=e_p^i$, which imply $s_i\in Y^{Op}$. Therefore, $S_F^{GP13}\subseteq Y^{Op}$.

In conclusion, we have proved $Y^{Op} \subseteq S_F^{GP13}$ and $S_F^{GP13} \subseteq Y^{Op}$. Therefore, we have $S_F^{GP13} = Y^{Op}$.

3. To prove that $S_A^{GP13} = Z^{Op}$.

   After Definition 2.1, we have $S_A^{GP13} = S \setminus S_B^{GP13} \setminus S_F^{GP13}$ and $Z^{Op} = S \setminus X^{Op} \setminus Y^{Op}$, where S denotes the set of all investigated statements. Since we have proved $S_B^{GP13} = X^{Op}$ and $S_F^{GP13} = Y^{Op}$, it is obvious that $S_A^{GP13} = Z^{Op}$.

□

Now, we are ready to prove that GP13, Naish1 and Naish2 belong to the same group of equivalent formulas (referred to as New_ER1).

Proposition 3.3. GP13 $\leftrightarrow$ Naish1 and GP13 $\leftrightarrow$ Naish2.

Proof. Refer to Lemma 3.1 and Lemma 3.2, we have $S_B^{N1} = S_B^{N2} = S_B^{GP13}$, $S_F^{N1} = S_F^{N2} = S_F^{GP13}$ and $S_A^{N1} = S_A^{N2} = S_A^{GP13}$, respectively. After Theorem 2.7, GP13 $\leftrightarrow$ Naish1 and GP13 $\leftrightarrow$ Naish2. □

Apart from GP13, we have three new maximal GP-evolved formulas for single-fault scenario, namely, GP02, GP03 and GP19. Unlike GP13, these three formulas do not belong to New_ER1 or ER5.

Proposition 3.4. GP02, GP03, GP19, New_ER1 and ER5 are distinct maximal formulas (or groups of equivalent formulas).

Proof. To prove this, we will demonstrate that neither $R_1 \rightarrow R_2$ nor $R_2 \rightarrow R_1$ is held, where R_1 and R_2 are any two of these five formulas (or groups of equivalent formulas). Consider the following two program PG_1 and PG_2 as shown in Figure 2 and Figure 3, respectively. Suppose two test suites $TS1_1$ and $TS1_2$ are applied on PG_1 and two test suites $TS2_1$ and $TS2_2$ are applied on PG_2. Vector A_i with respect to these test suites and programs are listed in Table 2.

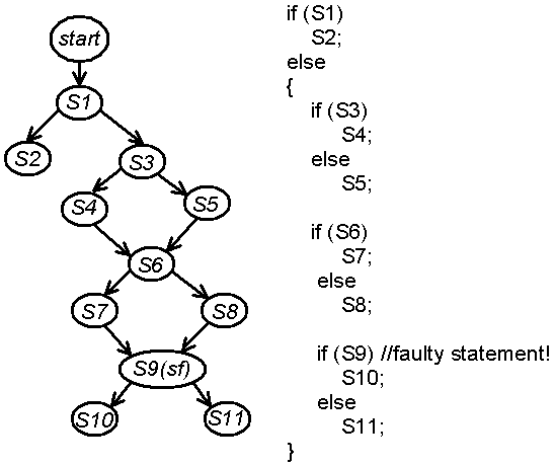

Figure 2: Program PG_1

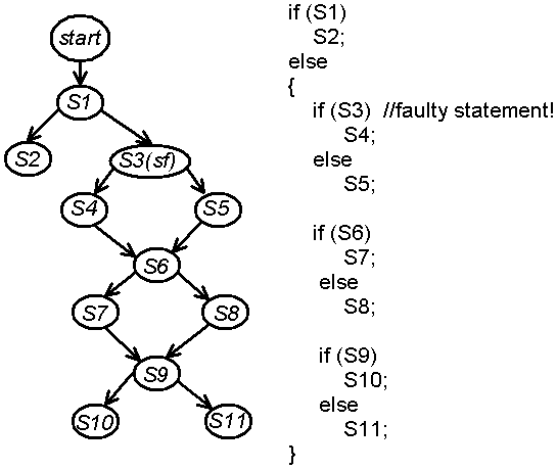

Figure 3: Program PG_2

Table 3 lists the statement divisions for these five formulas with respect to $TS1_1$ and $TS1_2$ applied on PG_1. and Table 4 lists the statement divisions for these five formulas with respect to $TS2_1$ and $TS2_2$ applied on PG_2.

Suppose we adopt the "ORIGINAL ORDER" as the tie-breaking scheme. Then the corresponding rankings of the faulty statement for these five formulas are as Table 5. From this table, we have demonstrated that

- With $TS1_2$ New_ER1 $\rightarrow$ GP02 does not hold; with $TS2_1$ GP02 $\rightarrow$ New_ER1 does not hold.

Table 2: A_i for PG_1 and PG_2 with different test suites

Statement	A_i=<$e_f^i, e_p^i, n_f^i, n_p^i$>			
	$TS1_1$	$TS1_2$	$TS2_1$	$TS2_2$
s_1	<1, 6, 0, 0>	<1, 8, 0, 0>	<2, 15, 0, 0>	<10, 15, 0, 0>
s_2	<0, 1, 1, 5>	<0, 6, 1, 2>	<0, 1, 2, 14>	<0, 1, 10, 14>
s_3	<1, 5, 0, 1>	<1, 2, 0, 6>	<2, 14, 0, 1>	<10, 14, 0, 1>
s_4	<1, 4, 0, 2>	<1, 1, 0, 7>	<1, 7, 1, 8>	<9, 0, 1, 15>
s_5	<0, 1, 1, 5>	<0, 1, 1, 7>	<1, 7, 1, 8>	<1, 14, 9, 1>
s_6	<1, 5, 0, 1>	<1, 2, 0, 6>	<2, 14, 0, 1>	<10, 14, 0, 1>
s_7	<1, 4, 0, 2>	<1, 1, 0, 7>	<1, 8, 1, 7>	<5, 6, 5, 9>
s_8	<0, 1, 1, 5>	<0, 1, 1, 7>	<1, 6, 1, 9>	<5, 8, 5, 7>
s_9	<1, 5, 0, 1>	<1, 2, 0, 6>	<2, 14, 0, 1>	<10, 14, 0, 1>
s_{10}	<1, 4, 0, 2>	<1, 1, 0, 7>	<1, 9, 1, 6>	<1, 12, 9, 3>
s_{11}	<0, 1, 1, 5>	<0, 1, 1, 7>	<1, 5, 1, 10>	<9, 2, 1, 13>

- With $TS1_2$ ER5 $\rightarrow$ GP02 does not hold; with $TS2_1$ GP02 $\rightarrow$ ER5 does not hold
- With $TS1_1$ New_ER1 $\rightarrow$ GP03 does not hold; with $TS1_2$ GP03 $\rightarrow$ New_ER1 does not hold.
- With $TS1_1$ ER5 $\rightarrow$ GP03 does not hold; with $TS1_2$ GP03 $\rightarrow$ ER5 does not hold.
- With $TS1_1$ New_ER1 $\rightarrow$ GP19 does not hold; with $TS1_2$ GP19 $\rightarrow$ New_ER1 does not hold.
- With $TS1_1$ ER5 $\rightarrow$ GP19 does not hold; with $TS1_2$ GP19 $\rightarrow$ ER5 does not hold.
- With $TS1_1$ GP02 $\rightarrow$ GP03 does not hold; with $TS1_2$ GP03 $\rightarrow$ GP02 does not hold.
- With $TS1_1$ GP02 $\rightarrow$ GP19 does not hold; with $TS1_2$ GP19 $\rightarrow$ GP02 does not hold.
- With $TS2_1$ GP03 $\rightarrow$ GP19 does not hold; with $TS2_2$ GP19 $\rightarrow$ GP03 does not hold.

In summary, we have proved that for any two of these five formulas (or groups of equivalent formulas) R_1 and R_2, neither $R_1 \rightarrow R_2$ nor $R_2 \rightarrow R_1$ is held. Therefore, GP02, GP03, GP19, New_ER1 and ER5 are five distinct maximal formulas (or groups of equivalent formulas). □

4 Discussion

Yoo [2012] used a small number of programs and faults to evolve new risk evaluation formulas, more precisely, four subject programs and 20 mutants for evolution. As suggested by Yoo [2012], "the results should be treated with caution" since "there is no guarantee that the studied programs and faults are representative of all possible programs and faults".

In this paper, we use the theoretical framework recently proposed by Xie et al. [2012] to analyze Yoo's GP-evolved risk evaluation formulas for single-fault scenario. Among Yoo's formulas, four have been proved to be maximal, namely, GP02, GP03, GP13 and GP19, where GP13 forms a new maximal group of equivalent formulas with Naish1 and Naish2. This new maximal group is referred to as New_ER1); while GP02, GP03 and GP19 are distinct to New_ER1 and ER5. Moreover, New_ER1 is strictly better than the remaining 16 GP-evolved formulas under investigation.

Results in this paper are exempt from the inherent disadvantages of experimental studies, and hence are definite conclusions for any program and fault under the assumptions that are commonly adopted by the

Table 3: Statement division for PG_1 with $TS1_1$ and $TS1_2$

Statement	$TS1_1$	$TS1_2$
New_ER1	$S_B^R = \{s_4, s_7, s_{10}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2, s_5, s_8, s_{11}\}$	$S_B^R = \{s_4, s_7, s_{10}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2, s_5, s_8, s_{11}\}$
ER5	$S_B^R = \emptyset$ $S_F^R = \{s_1, s_3, s_4, s_6, s_7, s_9, s_{10}\}$ $S_A^R = \{s_2, s_5, s_8, s_{11}\}$	$S_B^R = \emptyset$ $S_F^R = \{s_1, s_3, s_4, s_6, s_7, s_9, s_{10}\}$ $S_A^R = \{s_2, s_5, s_8, s_{11}\}$
GP02	$S_B^R = \{s_4, s_7, s_{10}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2, s_5, s_8, s_{11}\}$	$S_B^R = \emptyset$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$
GP03	$S_B^R = \{s_1\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$	$S_B^R = \{s_1, s_2, s_5, s_8, s_{11}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_4, s_7, s_{10}\}$
GP19	$S_B^R = \{s_1\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$	$S_B^R = \{s_1, s_4, s_7, s_{10}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_2, s_5, s_8, s_{11}\}$

SBFL community. It is a surprise that without exhausting all possible programs and faults, GP can still deliver maximal formulas. Moreover, such a process is totally automatic and does not involve any human intelligence. Thus, the cost of designing effective risk evaluation formulas can be significantly reduced.

Apart from generating new maximal formulas, GP actually discloses new intuitions or insights of maximal formulas. First, as a new member of an existing maximal group of equivalent formulas, GP13 has enriched the knowledge of this group and provided new insights to maximal formulas. By analyzing the definition of GP13, we find that it complies with the commonly adopted intuition that statements associated with more *failed* or less *passed* testing results should have higher risks. However, the other three GP-evolved formulas, though maximal, do not comply with this intuition, as explained below.

For GP02, given two statements s_1 and s_2,

- if $a_{ep}^1{=}a_{ep}^2$, then $a_{ef}^1{>}a_{ef}^2$ implies GP02(s_1)>GP02(s_2), which is consistent with the commonly adopted intuition;
- if $a_{ef}^1{=}a_{ef}^2$, then $a_{ep}^1{<}a_{ep}^2$ does not necessarily imply GP02(s_1)>GP02(s_2). For example, $a_{ef}^1{=}a_{ef}^2{=}1$, $P{=}8$, $a_{ep}^1{=}1$ and $a_{ep}^2{=}2$, then we have GP02(s_1)$=2(1+\sqrt{8-1})+1$, which is less than GP02(s_2)$=2(1+\sqrt{8-2})+\sqrt{2}$ Obviously, this does not comply with the commonly adopted intuition.

For GP03, given two statements s_1 and s_2,

- if $a_{ep}^1{=}a_{ep}^2$, then $a_{ef}^1{>}a_{ef}^2$ does not necessarily imply GP03(s_1)>GP03(s_2). For example, $a_{ep}^1{=}a_{ep}^2{=}25$, $a_{ef}^1{=}2$ and $a_{ef}^2{=}1$, then we have GP03(s_1)$=1$, which is less than GP03(s_2)$=2$. Obviously, this does not comply with the commonly adopted intuition.
- if $a_{ef}^1{=}a_{ef}^2$, then $a_{ep}^1{<}a_{ep}^2$ does not necessarily imply GP03(s_1)>GP03(s_2). For example, $a_{ef}^1{=}a_{ef}^2{=}1$, $a_{ep}^1{=}16$ and $a_{ep}^2{=}25$, then we have GP03(s_1)$=\sqrt{3}$, which is less than GP03(s_2)$=2$. Obviously, this does not comply with the commonly adopted intuition.

For GP19, given two statements s_1 and s_2,

Table 4: Statement division for PG_2 with $TS2_1$ and $TS2_2$

Statement	$TS2_1$	$TS2_2$
New_ER1	$S_B^R = \emptyset$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$	$S_B^R = \emptyset$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$
ER5	$S_B^R = \emptyset$ $S_F^R = \{s_1, s_3, s_6, s_9\}$ $S_A^R = \{s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$	$S_B^R = \emptyset$ $S_F^R = \{s_1, s_3, s_6, s_9\}$ $S_A^R = \{s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$
GP02	$S_B^R = \{s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2\}$	$S_B^R = \{s_4, s_{11}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2, s_5, s_7, s_8, s_{10}\}$
GP03	$S_B^R = \{s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1\}$	$S_B^R = \emptyset$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_1, s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$
GP19	$S_B^R = \{s_1\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_2, s_4, s_5, s_7, s_8, s_{10}, s_{11}\}$	$S_B^R = \{s_1, s_4, s_{11}\}$ $S_F^R = \{s_3, s_6, s_9\}$ $S_A^R = \{s_2, s_5, s_7, s_8, s_{10}\}$

Table 5: Rankings of faulty statement for five formulas

Statement	PG_1 (s_f=s_9)		PG_2 (s_f=s_3)	
	$TS1_1$	$TS1_2$	$TS2_1$	$TS2_2$
New_ER1	6	6	1	1
ER5	6	6	2	2
GP02	6	3	7	3
GP03	4	8	8	1
GP19	4	7	2	4

- if a_{ep}^1=a_{ep}^2, then a_{ef}^1>a_{ef}^2 does not necessarily imply GP19(s_1)>GP19(s_2). For example, P=20, a_{ep}^1=a_{ep}^2=10; F=4, a_{ef}^1=2 and a_{ef}^2=1, then we have GP19(s_1)=0, which is less than GP19(s_2)=$\sqrt{2}$. Obviously, this does not comply with the commonly adopted intuition.

- if a_{ef}^1=a_{ef}^2, then a_{ep}^1<a_{ep}^2 does not necessarily imply GP19(s_1)>GP19(s_2). For example, F=2, a_{ef}^1=a_{ef}^2=1; P=10, a_{ep}^1=8 and a_{ep}^2=9, then we have GP19(s_1)=$\sqrt{6}$, which is less than GP19(s_2)=$\sqrt{8}$. Obviously, this does not comply with the commonly adopted intuition.

Generally speaking, formulas defined by human beings are usually confined to the perceived intuition and background of the proposers. Thus, some maximal formulas may be overlooked by human beings. However, GP does not suffer from this problem. As explained in the above examples for GP02, GP03 and GP19, GP is able to define maximal formulas based on intuitions that would never be endeavoured by human beings.

In summary, GP has the advantage of being unbiased and hence has no predefined intuition nor unacceptable intuition to design the formulas. This study provides an evidence to show the advantage of using GP rather than human beings to design formulas. In addition to the delivery of new maximal formulas which would not be generated by human beings, these maximal formulas provide new perspectives and insights to enrich our knowledge of effective formulas.

5 Conclusion

Search-based techniques have been widely used in software engineering, such as testing, maintenance, etc. [Harman and Jones, 2001; Harman, 2010]. Recently, Yoo [2012] has successfully utilized a search-based technique, namely, Genetic Programming, to generate effective risk evaluation formulas for SBFL. In this paper, by using the recently developed theoretical framework [Xie et al., 2012] on Yoo's GP-evolved formulas under single-fault scenario, we have demonstrated that four formulas are maximal, namely, GP02, GP03, GP13 and GP19. Obviously, our results provide a strong support that "Genetic Programming can be an ideal tool for designing risk evaluation formulas". Moreover, these four maximal formulas may help to mine new insights and intuitions of effective formulas, which somehow have been ignored by human beings.

References

Abreu, R., Zoeteweij, P., and van Gemund, A. J. C. (2006). An evaluation of similarity coefficients for software fault localization. In *Proceedings of the 12th Pacific Rim International Symposium on Dependable Computing*, pages 39–46, Riverside, USA.

Chen, M., Kiciman, E., Fratkin, E., Fox, A., and Brewer, E. (2002). Pinpoint: problem determination in large, dynamic internet services. In *Proceedings of the 32th IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 595–604, Washington DC, USA.

Harman, M. (2010). The relationship between search based software engineering and predictive modeling. In *Proceedings of the 6th International Conference on Predictive Models in Software Engineering*, pages 1:1–1:13, Timişoara, Romania.

Harman, M. and Jones, B. (2001). Search based software engineering. *Information and Software Technology*, 43(14):833C839.

Jones, J. A., Harrold, M. J., and Stasko, J. (2002). Visualization of test information to assist fault localization. In *Proceedings of the 24th International Conference on Software Engineering*, pages 467–477, Florida, USA.

Naish, L., Lee, H. J., and Ramamohanarao, K. (2011). A model for spectra-based software diagnosis. *ACM Transactions on Software Engineering and Methodology*, 20(3):11:1–11:32.

Xie, X. Y., Chen, T. Y., Kuo, F.-C., and Xu (2012). A Theoretical Analysis of the Risk Evaluation Formulas for Spectrum-Based Fault Localization. *Accepted by the ACM Transactions on Software Engineering and Methodology*.

Yoo, S. (2012). Evolving human competitive spectra-based fault localisation techniques. In *Proceedings of the 4th International Symposium on Search-Based Software Engineering*, Trento, Italy.