C コンパイラ用テストスイート CF3 の検出能力向上

Enhancing Detection Ability of C Compiler Test Suite CF3

池尾 弘史 石浦菜岐佐
Hirofumi Ikeo Nagisa Ishiura

関西学院大学 理工学部 School of Science and Technology, Kwansei Gakuin University

1 はじめに

コンパイラはソフトウェア開発において非常に重要な役割を担っており, その信頼性を確保するためにテストスイートやランダムテストによる徹底的なテストが行われる. CF3 [1] は, C コンパイラの算術最適化を対象としたテストスイートであり, 3 演算からなる算術式に対するコードが正しく生成されるかどうかをテストする. CF3 はテスト効率の点では優れているが, 一部の演算子を含む式に関する不具合の検出数が少ないことが課題となっていた. そこで本稿では, 変数の初期値に用いる値の集合を変更することにより, CF3 の検出能力を向上させる手法を提案する.

2 C コンパイラ用テストスイート CF3

C コンパイラの算術最適化を対象とするランダムテスト Orange3 [2] は GCC や Clang/LLVM で多くの不具合を検出しているが, エラーを引き起こすプログラムを最小化すると, その半数は演算数 3 以下の算術式を 1 つだけ含む単純なものになる [1]. CF3 は, これに着目し, この規模の算術式を含むプログラムを集めることによりテストの効率化を図ったものである. 図 1 は, `((x1>>x2)<(x3|x4))` という形の式 (111, 120 行目) に対するテストプログラムの例であり, 1 ファイル中に変数の型や初期値が異なる 100 の テストケース (`f0`∼`f99`) を収容している. CF3 は, 同様のファイルを 3 演算で構成可能な 10,985 通り全ての式の形に対して集めたものである. 変数の初期値には, 最小化したエラープログラム中で出現頻度の高かった 1, 2, 31, および min, max (変数の型の最小値, 最大値) を使用している. 変数の型や初期値の組合せの網羅は非現実的であるため不具合の見逃しは避けられないが, 一部の演算, 特にシフト演算や剰余算を含む式に関する不具合の検出数が少ないことが課題となっていた.

```
   1: #include<stdio.h>
   2: #include<stdint.h>
   3: #include<stdlib.h>
   4: #define NG() print("@NG@ %s", __func__);
   5: static uint8_t x2 = 0U;
                ...
 106: void f0(void) {
 107:   static uint64_t x1 = 2U;
 108:   int16_t x3 = 1U;
 109:   volatile int8_t x4 = 1;
 110:   int32_t t0 = 10;
 111:   t0 = ((x1>>x2)<(x3|x4));
 112:   if(t0 != 0){NG();} else {;}
 113: }
 114: void f1(void) {
 115:   int16_t x5 = 2U;
 116:   uint8_t x6 = 2U;
 117:   uint8_t x7 = 1U;
 118:   uint16_t x8 = INT16_MAX;
 119:   static int32_t t2 = -829136;
 120:   t2 = ((x5>>x6)<(x7|x8));
 121:   if(t2 != 1){NG();} else {;}
 122: }
      ...
1308: int main(void){
1309:   f0();f1();...;f99();
1411:   return 0;
1142: }
```

図 1 CF3 のテストプログラムの例 (test1946.c)

3 変数の初期値集合の変更による検出率の向上

本稿では, CF3 のテストプログラムの初期値に用いる値の集合を変更することにより, この課題の解決を図る. Orange3 が不具合を検出したプログラムの分析に基づき, 変数の初期値として固定値 1, 2, 31 の代わりに乱数を用いるとともに, −1 を追加する. ただし乱数には, 対数スケールで一様な乱数を使用する.

4 実験結果

x86_64 用の GCC と Clang/LLVM の 9 バージョンを従来と本稿の CF3 でテストする実験を行った. 本稿の CF3 における変数の初期値として, 乱数, min, max, −1 が出現する確率はそれぞれ 46%, 22%, 22%, 10%とした. 検出エラー数を表 1 に示す. 「時間」は本稿の CF3 によるテストに要した時間 (8 スレッドで実行) である. 本稿の CF3 は従来に比べて全てのコンパイラでエラー検出数が向上した. 検出したエラープログラムを分析すると, シフト演算や剰余算の出現数が増加していた.

表 1 検出エラー数

compiler	従来	本稿 (時間 [h])	
GCC-4.5	28	33	(1.69)
GCC-4.4	28	34	(1.64)
GCC-4.3	38	41	(1.61)
GCC-4.2	28	56	(1.83)
Clang-3.4	39	202	(0.88)
Clang-3.3	46	212	(0.81)
Clang-3.2	58	214	(0.82)
Clang-3.1	57	216	(0.75)
Clang-3.0	54	209	(0.79)

CPU: Intel Core i7-870@2.93GHz×8

5 むすび

本稿では, 変数の初期値集合の変更により CF3 の検出能力を向上させる手法を提案した. 現在の CF3 が網羅している式のパターン集合の中には冗長なものが含まれているので, これを削除してテストスイートのサイズを削減することが今後の課題である.

謝辞 本研究は一部 JSPS 科研費 25330073 の助成による.

参考文献

[1] 日比野, 石浦: “C コンパイラの算術最適化を対象としたテストスイート CF3,” 信学技報, VLD2014-130 (Jan. 2015).

[2] E. Nagai, A. Hashimoto, and N. Ishiura: “Reinforcing random testing of arithmetic optimization of C compilers by scaling up size and number of expressions,” IPSJ Trans. SLDM, vol. 7, pp. 91–100 (Aug. 2014).