C コンパイラの最適化のリグレッションテストのための アセンブリコード比較法

Comparison of Assembly Codes for Regression Test of C Compiler Optimization

北浦 幸太　Kota Kitaura
岩辻 光功　Mitsuyoshi Iwatsuji
石浦 菜岐佐　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

コンパイラには, 信頼性とともに高い最適化の性能が要求されるため, 意図通りの最適化が行われているかどうかはコンパイラの重要なテスト項目になる. 文献 [1] では, 2 つのコンパイラが生成するアセンブリコードの比較によって一方の最適化の不足を検出する手法を提案している. しかし, その比較は単純に命令数だけに基づいて行っていたため, 誤判定が生じていた. そこで本稿では, C コンパイラの最適化のリグレッションテストを対象に, アセンブリコードの比較法の改良を提案する.

2　差分テストに基づくコンパイラの最適化不足の検出

文献 [1] の手法は, ランダムに生成したプログラムを 2 つの異なるコンパイラでコンパイルし, 生成されるアセンブリコードを比較することによって一方の最適化不足を検出する. 同じコンパイラの異なるバージョンでテストを行えば, 最適化のリグレッションをテストすることもできる. プログラムの生成に用いる Orange3 [2] は, C コンパイラの算術最適化を対象としたランダムテストシステムであり, 図 1 のようなプログラムを生成する. 式数, 式の形, 演算子, 変数の型および初期値は設定に基づきランダムに決定される.

[1] では 2 つのアセンブリコードの命令数の比が閾値以下の場合に差があると判定していたが, 目視確認するとその中には, 性能に差があると判定できないものが含まれることが課題になっていた.

3　アセンブリコード比較法

本稿では, C コンパイラの最適化のリグレッションテストを対象に, より精度の高いアセンブリコードの比較法を提案する. 本手法は, アセンブリコード中の不一致部分の抽出と, 不一致命令の重み和の比較に基づく.

比較方法を図 2 に示す. アセンブリコードが k 命令以上連続して一致している部分を手がかりに, 不一致命令列の対 (S_i, S_i') を抽出する. 不一致命令列対のいずれかに有意な差があれば, アセンブリコードには差があると判定する. (S_i, S_i') は, 対応が取れる命令を削除した後, 残った命令の重みの和を求め, その比が閾値以下であれば差があると判定する. 重みは, 乗除算や分岐命令が ALU 演算命令よりも大きくなるように設定する.

4　実験結果

本手法に基づくテストシステムを Perl5 で実装し, GCC-5.2.1 を比較対象に GCC-6.0.0 (-O3 オプション) のテストを行った. 結果を表 1 に示す. 比較の閾値は, 従来法は 0.9, 本手法は 0.8 とした. 従来手法では検出した差分 181 件のうち, 10 件が誤判定であったのに対し, 本手法はより多くの差分を検出し, 誤判定はなかった. 従来手法で検出した 171 件は本手法でも全て検出していた. 本手法により検出した GCC-7.0.0 (開発版) の最適化のリグレッション 2 件を開発者に報告した [1].

```
 1: #include <stdio.h>
 2: #define OK() printf("@OK@\n")
 3: #define NG(fmt,val) printf("@NG@ (test="fmt")\n",val)
 4: const signed int k8 = 144011145;
 5: int main (void)
 6: {
 7:   signed short x1 = 6;
 8:   static unsigned short x2 = 1U;
 9:   static signed long x4 = 4542636934L;
10:   volatile signed short x7 = -1;
11:   signed long t0 = -1261917469307207940L;
12:   signed long t1 = -42079927921L;
13:   t0 = (x2^(x4+((x7+k8)>>x2)));
14:   t1 = ((x4<<(x4/t0))/x1);
15:   if (t0 == 4614642507L) {OK();} else {NG("%d", t0);}
16:   if (t1 == 757106155L) {OK();} else {NG("%lld", t1);}
17:   return 0;
18: }
```

図 1　Orange3 が生成するテストプログラムの例

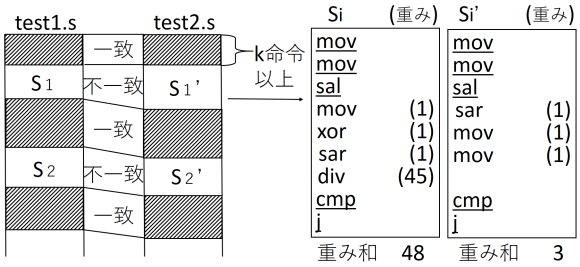

図 2　アセンブリコードの比較法

表 1　実験結果

method	テスト数	差分検出数	
		正判定	誤判定
従来手法 [1]	30,000	171	10
本手法		764	0

5　むすび

本稿では, C コンパイラの最適化のリグレッションテストを対象としたアセンブリコード比較法を提案した. 異なるコンパイラでの比較方法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] 岩辻, 橋本, 石浦：“差分ランダムテストに基づくコンパイラの最適化機会の検出,” 電子情報通信学会ソサイエティ大会, A-3-2 (Sept. 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).

[1] https://gcc.gnu.org/bugzilla/ id=71521, id=71563