Java に対するループインバリアントを含む Daikon 生成アサーションの妥当性評価

宮本　敬三[†]　　岡野　浩三[†]（正員）
楠本　真二[†]（正員）

Adequacy Evaluation of Assertions for Java Including Loop Invariants Generated by Daikon

Keizo MIYAMOTO[†], *Nonmember*, Kozo OKANO[†], and Shinji KUSUMOTO[†], *Members*

[†] 大阪大学大学院情報科学研究科，豊中市
Graduate School of Information Science and Technology, Osaka University, Toyonaka-shi, 560–8531 Japan

あらまし　Daikon が生成する事前・事後条件の表明を ESC/Java2 で評価すると再現率，適合率ともに高いという研究結果がある．小規模のプログラムに対し，事前・事後条件，ループ不変式の再現率，適合率を ESC/Java2 及び人手で行った．その結果を示し，考察を与える．

キーワード　Java，表明，Daikon，JML，ESC/Java2

1. まえがき

DbC [1] はクラスとクラスの利用側の間における仕様決定を契約とみなすことでソフトウェアの信頼性，保守性を高めることを目的としている．この契約を記述する表明の自動生成ツールの一つである Daikon は，テストケース上でプログラムを実行し動的解析を行うことで表明を自動生成する [2]．しかし，Daikon が生成できるのはメソッドの事前条件，事後条件のみである．だが，ループ不変式は繰返し処理の停止性や妥当性に関連するプログラムの可読性，保守性の向上，検証自動化という観点から重要である [1]．一方，Daikon の表明生成能力については文献 [3] で ESC/Java2 [4] で再現率，適合率を検証すると，ともに高い結果（約 90%）が得られたと報告されている．しかし，表明自動生成の実用的観点からは，ループ不変式の評価や人の観点からの評価も大事である．

そこで，本研究では Java プログラムに対し，事前・事後条件，ループ不変式を Daikon で生成し，ESC/Java2 [4] と人手で，生成表明の妥当性を評価した．その結果，人手で評価した場合，Daikon が生成したメソッドとループ部の表明の妥当性は現状ではやや低いということを確認した．

2. 準　　備

JML, Daikon，ESC/Java2 について簡単に述べる．

JML（Java Modeling Language）[5] は Java プログラムコード中にコメント形式で表明を記述するための形式的記述言語である．

Daikon は生成した表明を JML 形式など様々な形式で出力できる．また，生成された表明を JML 形式で対象プログラムに挿入する等，利便性の高い機能も有している．

ESC/Java2 [4] はプログラム中に JML で記述した表明を静的解析により判定するツールである．表明を満たすか否かの情報とともに，満たさない場合は理由を出力する．

3. 評　　価

3.1 評価の目的と基準

評価実験の目的はメソッド，ループ部に対して生成された表明の妥当性の検証である．Daikon が動的解析により生成した表明の適合率と再現率，F 値を求めることにより生成された表明の妥当性を評価する．なお，ループ不変式は筆者らが開発したツールを用いて，間接的に Daikon に生成させた．適合率 (P)，再現率 (R)，F 値 (F) は，Daikon により生成された全表明数 (A)，ESC/Java2 において正しいと判定されたアサーション数 (E)，人が記述した必要な表明数 (C)，人が記述した必要な表明と Daikon により生成された表明の共通集合の要素数 (N) を用いて以下のように定義される．

- 適合率：P ($P = E/A$)
- 再現率：R ($R = N/C$)
- F 値：F ($F = \frac{2 \times P \times R}{P+R}$)

この F 値が高いほど妥当性が高いとする．

また，生成された表明の再現率を評価するための基準として，必要な表明，不要な表明を以下のように定義する．

- 必要な表明：プログラムの動作内容，ソースコードの内容を理解した人間が記述した表明
- 不要な表明：特定のテストケースに依存し一般的には成立しない表明

例えば，入力された配列に対してインサーションソートを行うメソッドに対して要素数 10 の配列 a をテストケースとして Daikon で表明を出力させる．このときループカウンタとして用いられている変数 i に関する事後条件として [ensure i==10] という表明が出力される．この表明は動作結果としては正しいが入力される配列の要素数は固定ではないので，一般的には成立しない．このような表明を不要な表明とする．

表 1　対象プログラム
Table 1　Target programs.

No.	プログラム名	No.	プログラム名
1	Sum.java	4	iSorting.java
2	Min.java	5	qSorting.java
3	bSorting.java	6	calc.java

3.2　評価対象のあらまし

評価対象として表 1 に示す 6 種類のプログラムを用いた．プログラム名が内容を表している．calc.java は逆ポーランド記法の数式を計算するプログラムである．これらのプログラムはループがアルゴリズム中で重要な役割を果たしている．また，条件分岐，配列変数などループ以外にもプログラムに重要な要素を含んでいるという点でも評価対象に適していると考える．

3.3　結　　果

表 1 の対象プログラムに対し生成された表明のうち，ループ部に対して生成されたものの結果を表 2 に示す．また，人手で適合率を求めた場合のメソッドとループ部の表明の比較結果を表 3 に示す．

4.　考　　察

結果は記載しないが，ループ部の事前条件，事後条件，不変式ごとの適合率の平均を比較すると同程度になった．また，再現率，F 値に関しても同様の結果が得られた．Daikon がプログラムに対して表明を生成できるのはメソッド部に限られているために，直接ループ部に対して事前条件，事後条件，ループ不変式を生成することはできない．そこで，本論文では，ループ部を抽出し新たなメソッドとし，このメソッドに対して表明を生成することでループ部の事前条件，事後条件を生成した．また，ループ内に空のメソッドを追加し，このメソッドに対して表明を生成することでループ不変式を生成した．このため，ループ部の事前条件，事後条件の適合率，再現率，F 値はメソッド部のものと同程度になると考えられる．実験の結果得られた適合率，再現率，F 値は全体的に低いという結果になったが，Daikon を実際に利用する上での課題が分かったという点で意味があると考えられる．また，Daikon はそもそも事前条件，事後条件の生成が主目的であり，ループ不変式は比較的考慮されていない．そのため，Daikon をループ不変式生成に使えるか否かの評価も必要となる．結果として，表 2 に示したループ部の適合率，再現率は文献 [3] のものより低くなることが分かった．適合率の低下の原因として，メソッドとループ部の表明を求める際に考慮すべき条件の違いが

表 2　ループ部における適合率，再現率，F 値
Table 2　The results of loop parts.

	Loop		
No.	P	R	F
1	0.76	0.29	0.41
2	0.87	0.57	0.69
3	0.88	0.67	0.76
4	0.92	0.64	0.76
5	0.90	0.56	0.69
6	0.94	0.46	0.62

表 3　人手で求めた適合率
Table 3　Metrics evaluated by human.

	Method			Loop		
No.	P	R	F	P	R	F
1	0.22	0.20	0.29	0.53	0.29	0.37
2	0.50	0.20	0.21	0.43	0.57	0.49
3	0.05	0.20	0.08	0.56	0.67	0.61
4	0.06	0.40	0.10	0.28	0.64	0.39
5	0.07	0.20	0.10	0.15	0.56	0.23
6	0.11	0	0	0.13	0.46	0.20

考えられる．メソッドの事前条件，事後条件はクラスのフィールド変数，メソッドの仮引数，返り値の間に成立する条件を求めることになる．一方，ループ部の事前条件，事後条件の場合はフィールド変数，メソッドの仮引数，ローカル変数の間に成立する条件を求めることになる．対象プログラムには，ローカル変数は複数存在しているために，ループ部の事前条件，事後条件の方がメソッド部より考慮すべき条件が複雑になる．この結果，適合率が文献 [3] の値より低下したと考えられる．また，再現率が低下の原因としては，判定方法の違いが考えられる．文献 [3] では再現率を求める際に ESC/Java2 による判定を用いている．しかし，本論文では人手による判定を用いた．この結果，再現率が文献 [3] の値より低下したと考えられる．また，表 2 の結果をプログラム別に見ると calc.java の適合率，再現率，F 値が他のプログラムよりも低いことが分かる．これは，他の五つのプログラムへのテストケースは Java の乱数生成メソッドを用いて自動生成したが，calc.java へのテストケースは自動生成ではなく人手で作成したために入力データ数が少なくなってしまった，逆ポーランド記法の数式の値を計算するという処理内容が他のプログラムに比べて複雑であるために変数の値を監視するだけでは正しい表明を生成しにくかったためと考えられる．他のプログラムは入力，出力データともに int 型の配列変数であり変数の値の監視だけで必要な表明を生成しやすいと考えれら

れる．

表 3 の適合率は表 2 に比べ低下している．本研究では再現率も人手により求めている．これらの結果は，適合率，再現率をともに ESC/Java2 の判定により求めた既存の結果 [3] と比較するとかなり悪い．これは，ESC/Java2 としては正しいが人間が見て不要と思われる表明が大量にでたことが理由である．このことから，ESC/Java2 による表明の判定よりも人手による判定の方が厳しいといえる．なお，結果は紙面の関係上記載しないが，メソッドの事前条件と事後条件では，事後条件の方が適合率，再現率ともに事前条件のものより良くなる傾向があった．以上の結果をまとめると，現段階での Daikon が生成した表明の妥当性は ESC/Java2 で機械的に計測した場合，メソッド，ループ部ともに同程度である．しかし，適合率を人手で判定した場合，ESC/Java2 で適合率を求めた場合に比べ，表明の F 値は表 3 に示したように総じて低く妥当性はメソッド，ループ部ともに十分とはいえない．妥当性を高めるためには適合率，再現率の向上が必要である．

5. む す び

本研究では，ループ構造が主体となり，アルゴリズムの実装が主となる比較的小規模のプログラム群に対して Daikon が生成した表明の適合率を ESC/Java2 による判定と人手による判定の 2 通りの方法で求めた．その結果，適合率は ESC/Java2 による判定では高いが，人手による判定では低いことを確認した．今後の課題として，表明自動生成の観点から，必要な表明の生成精度の向上，無駄な表明の削除が挙げられる．これらを解決するためには表明の生成，削減手法の改善だけでなく，テストケースの作成方法の改良が必要である．また，静的解析と Daikon の動的解析を組み合わせるということも考えられる．

文 献

[1] B. Meyer, Object-Oriented Software Construction, 2nd ed., Prentice-Hall, 2000.

[2] M.D. Ernst, J.H. Perkins, P.J. Guo, S. Mccamant, C. Pacheco, M.S. Tschantz, and C. Xiao, “The daikon system for dynamic detection of likely invariants,” Science of Computer Programming, vol.69, no.1-3, pp.35–45, 2007.

[3] J.W. Nimmer and M.D. Ernst, “Automatic generation of program specifications,” Proc. 2002 Int. Symp. on Software Testing and analysis (ISSTA), pp.232–242, 2002.

[4] C. Flabagan, K. Rustan, M. Leino, M. Lillibridge, G. Nelson, J.B. Saxe, and R. Stata, “Extended static checking for java,” Proc. ACM SIGPLAN 2002, pp.234–245, 2002.

[5] G.T. Leavens, A.L. Baker, and C. Ruby, “Preliminary design of JML: A behavioral interface specification language for java,” ACM SIGSOFT, vol.31, no.3, pp.1–35, 2006.

（平成 20 年 4 月 16 日受付，6 月 18 日再受付）