卒 業 論 文

プログラミング演習における模範解答を用いた
テストケース評価基準の自動生成

English Title

指 導 教 員　　中村　正樹 准教授

富山県立大学工学部 電子・情報工学科

学 籍 番 号 : 1515015

氏 名　　尾崎　裕樹

提 出 年 月　2018 年 2 月

目次

第1章　はじめに

本研究では、プログラミング教育において、教員が学生にソフトウェアテストの方法を指導する際に使用できるシステムを作成する。

1.1　背景

プログラミングを学習する上では、仕様からのコーディングだけでなく、コーディングの後に行うソフトウェアテストの方法を学ぶことも重要である。適切なソフトウェアテストを行うには、適切なテストケースの設計が必要となる。そのためには、適切なテストケースを設計するための教育が求められる。その際に、学生が設計したテストケースが適切であるかの評価を自動で行うことによって、教員の負担を減らすことができる。しかし、テストケースの入力データを評価する場合には境界値以外の値は誰が設計しても同じ値になるとは限らない。そのため、教員が模範となるテストケースを用意しても、学生のテストケースとの単純な比較だけでは評価できない。

1.2　目的

そこで、文献 [1] ではテストケースに対する評価基準を用意し、その基準をどれだけパスできるか判定することによってテストケース評価の自動化が行われている。本研究では、教員の手間をさらに削減するため、このテストケース評価基準を自動生成することを目的とする。

1.3　論文の構成

・・・

第2章　プログラミング演習におけるテストケース評価システム

本章では本研究の関連研究 [1] について説明する。関連研究では、学生自身が適切なテストケースを設計できるようになるために、作成したテストケースを評価してアドバイスを行うシステムが提案されている。このシステムでは、教員によって演習問題毎にテストケースの評価基準とテストケースが不足していた場合に表示するアドバイスが記述される。演習問題毎にテストしなければならない値や入力のデータ数が異なるため、評価基準は演習問題を分析した上で、入力のデータ構造を定義してから記述される。

2.1　テストケース評価システム

関連研究では、教員があらかじめテストケース評価基準を作成し、学生が設計したテストケースが評価基準をどの程度パスできるかを判定することによって、テストケースの評価が行われている。評価基準をパスしない場合にアドバイスを表示することで、不足しているテストケースを把握させた。関連研究において、テストケースの評価は PHP で行われ、評価基準は次のようにタブ区切り形式でテキストとして記述された。

評価基準の記述形式

```
評価基準番号 TAB 判定条件 TAB アドバイス
```

「判定条件」はテストケースにおける入力データが満たすべき条件で、その条件を満たすテストケースがなかったときに、学習者に「アドバイス」が表示される。同値クラスを評価するための基準が複数ある場合などは、同じ「評価基準番号」を指定することで、評価基準がグループ化されている。例えば、年齢（整数値）を入力して、20 歳以上は成年、20 歳未満は未成年と表示する課題の評価基準は、入力データを変数 age として、成年、未成年、エラーの場合について次のように記述される。

評価基準の記述例 1

```
1 TAB age==20 TAB 成年の最低年齢
1 TAB age>=20 TAB 成年の場合
2 TAB age==19 TAB 未成年の最高年齢
2 TAB age==0 TAB 0 歳
2 TAB age<=0 && age<20 TAB 未成年の場合
3 TAB age<0 TAB 年齢がマイナスの場合
```

学生に作成したテストケースの網羅率を把握させるために、次の式により評価基準のテストカバレッジを計算して提示される。

評価基準のテストカバレッジ（%）

```
学習者のテストケースがパスした評価基準グループ数÷
教員が記述した評価基準グループ数× 100
```

判定条件の記述において、課題毎に入力データの型や個数が異なるため、テストケースの入力のデータ構造を定義し、そこで定義された変数を用いて判定条件が記述される。

2.2 入力のデータ構造の定義

データ構造は型と変数と出現回数から成る。入力のデータ構造の記述法はバッカス・ナウア記法（BNF）に類した形式で次のように設計された。なお、[ ] はグループ化、— は選択、? は 0 回か 1 回、+ は 1 回以上、* は 0 回以上の繰り返しを表す。

入力データ構造の定義

```
データ構造 ::== [ ‘(’ 変数リスト ‘)’[出現回数]? ]+
変数リスト ::== 型 変数名 [ ‘,’ 型 変数名]*
出現回数 ::== ‘{’ ‘+’ | ‘*’ |式 ‘}’
式 ::== 変数名 | 数値 | 式 演算子 式
演算子 :== ‘+’ | ‘*’
```

2.3　関数定義

複数の入力データに対する処理や、頻繁に用いられる処理を記述するための関数を定義する方法が設計された。複数の身体データを読み込んで、身長の最大値を表示する課題の場合、最大値が複数存在するテストの評価基準は、引数のリストの最大値の個数を返す関数 countmax() を定義して、次のように記述される。なお、実引数 height は全身長データのリストである。

関数を用いた評価基準の記述例

```
1 TAB countmax(height)>=2 TAB 身長の最大値が複数の場合
```

Listing 2.1　countmax 関数

```
function countmax($list){

        $lmax = max($list);

        return count(array_keys($list, $lmax));

}
```

2.4　評価基準の具体例

入力が比較的複雑になる課題として、ボウリングの得点計算が例示されている。なおデータ構造の uint は 0 と正の整数の型である。

ボウリングの点の計算の課題

ボウリングは、１フレーム目から１０フレーム目までの倒れたピンの数によって得点が決まる。フレームごとに倒したピンの数を読み込み、各フレームごとの得点を計算して表示すること。なお、ガーターとミスは 0 と表示するものとし、ダブルがあった場合はメッセージを表示する。

ボウリングのデータ構造

```
(uint thr1, uint thr2){9}
(uint last1, uint last2, uint last3)
```

ボウリングの評価基準

```
1 TAB thr1 == 10 TAB ストライクのスコア計算
2 TAB thr1 != 10 && thr1+thr2 == 10 TAB スペアのスコア計算
3 TAB last1 == 10 TAB ストライクの通常加算
3 TAB last1 != 10 && last1+last2 == 10 TAB スペアの通常加算
4 TAB double_strike(thr1) || all_strike([last1, last2]) TAB ダブル
4 TAB last_strike(thr1) == 1 && last1 == 10 TAB 最終 F をまたぐダブル
```

ストライクとスペアだった場合のスコア計算が正しいかをテストする必要がある（評価基準番号 1, 2)。最終フレームではストライク、スペアでも通常の加算になるので、それらのテストも必要である（評価基準番号 3)。評価基準番号 4 はダブルについてのテストである。ダブルがあれば true を返す関数 double_strike()、 リストがすべて 10 ならば true を返す関数 all_strike()、リストの最後の要素から 10 が何回連続しているかを返す関数 last_strike() が定義されている。

2.5 システムの評価

参考書の例題のテストケースのデータ構造が評価システムに則した形式で記述できるか調査が行われ、実行時に入力のデータ構造が変化する共用体などの問題以外ではデータ構造を記述できることが確認された。

また、評価システムにより、学生がテストケースを作成できるようになるのかを確認するために、プログラミングを学習済みの学生 7 人がボウリングの課題のテストケースを作成した。その結果最終的に 6 人の学生がカバレッジ 100% を達成したことが確認された。

2.6 システムの課題

関連研究では、教員が問題文を分析した上で、評価基準を記述しているので、システムを運用する上で教員の負担が大きくなってしまうことが考えられる。そこで、教員の負担を減らすために、評価基準を自動で生成する方法を考える。

第3章　テストケース評価基準の自動生成

3.1　背景

第4章　検証

4.1　背景

第5章　まとめ

謝 辞

参 考 文 献

(1) 蜂巣吉成，小林悟，吉田敦，阿草清磁： プログラミング演習におけるテストケース評価システム，コンピュータソフトウェア第 34 巻第 4 号，2017，pp.54-60