プログラム間の類似性の定量化手法

Method of Similarity Quantification between Program Codes

小田 悠介[*1], 上村 康輔[*2], 若林 茂[*3]
Yusuke ODA[*1], Kosuke KAMIMURA[*2], Shigeru WAKABAYASHI[*3]
[*1] 神戸市立工業高等専門学校 専攻科 電気電子工学専攻
[*1]Kobe City College of Technology Advanced Course
[*2*3] 神戸市立工業高等専門学校 電子工学科
[*2*3]Department of Electronics, Kobe City College of Technology

あらまし：プログラミング入門教育時に学生から提出される課題プログラムをその手法（アルゴリズム）に基づいて分類する．そのためにはプログラム間の類似性を定量化する必要がある．以前の研究では，2つのプログラムを比較するときの比較レベル（比較感度）を設定し，それを順次変化させて初めて一致したレベル値をプログラム間の距離として定義した．今回その類似度の見直しを行い，新しい手法でのプログラムを開発したので報告する．
キーワード：プログラミング教育，プログラムの類似性，カーネル法，クラスタリング

1. はじめに

プログラミング教育の現場では，教師は学生から提出される大量のプログラムを目視により妥当性を判断し，それぞれのプログラムに適した評価を行う必要がある．これは多くの場合，ほとんど差異のないプログラムを何度も評価することになるため，非常に冗長な作業である．プログラム間の類似性を機械的に評価してグループ化ができれば，教師の負担は軽減し，また評価の自動化を行うための足掛かりとなる．これまでの研究では，プログラムの比較レベル（比較感度）を定義し，それぞれの比較レベル上でプログラム同士を比較する方法を採用してきた[(1)]．本稿ではこの方法を段階的感度比較法と呼ぶこととし，最初に解説を行う．次に，段階的感度比較法に起因する不具合を述べ，今回新たに導入したカーネル法に基づく類似性の評価法を示す．最後に実際の解析結果および考察を述べる．

2. 段階的感度比較法

2.1. 概説

2 つのプログラムを比較する方法には，例えばソースコードの先頭から末尾まで文字が完全に一致するか調べる方法，構文解析の結果得られる構文構造が一致するか調べる方法，同じ入力に対する出力が一致するかどうかだけ調べる方法など，いくつかの方法が考えられる．これら複数の比較法を，厳密性が高い（比較の感度が高い）と考えられるものから順に適用してゆくと，どこかで一致することになる．プログラム同士が初めて一致した比較レベルをプログラム間の距離とする．以前の研究では，次に示す 5 種類の比較法を比較レベルとして定義した．

比較レベル 0: 一文字ごとの完全比較

比較レベル 1: トークン列の比較

比較レベル 2: 変数名，計算式を標準化して比較

比較レベル 3: 制御構造を標準化して比較

比較レベル 4: 入出力だけの比較

これらの関係を図 1 に示す．

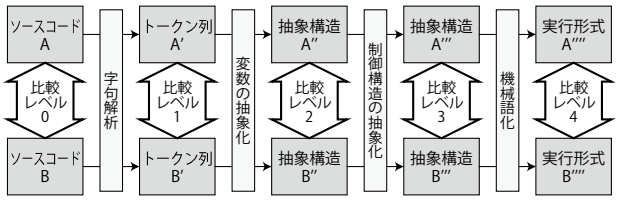


図 1: 段階的感度比較法

2.2. 問題点

段階的感度比較法にはいくつかの不具合がある．前節で示した比較レベルを例に取ると，まず，レベル 0 とレベル 1 の差と，レベル 1 とレベル 2 の差が同一であるという根拠がない．またレベル 2 とレベル 3 に論理的な順序関係が存在しない（変数名，計算式は同じで制御構造のみが違う場合）ことが挙げられる．つまり，比較レベルは間隔尺度ではなく，また順序尺度としても一部疑問点が残る．

3. カーネルを用いた類似性の定量化

3.1. カーネル法

本稿で述べるのは，上記の段階的感度比較法とは基本的に異なる定量化法であり，類似性を求める操作としてカーネル法を導入する．カーネル法とは，データ構造間の内積に相当するスカラ値を定義することによって，内積空間上の種々のアルゴリズムを一般のデータ構造へ適用できるようにする方法である．この内積を求める関数をカーネルと呼ぶ．プログラムはラベル付き木として表現できるので，木に関する既知のカーネルを使用することを考える．以下ではこれらの説明と，得られたカーネルからデータ間の類似性を推定する方法を述べる．

3.2. 共通ルート部分木カーネル

木の末端を適当な組み合わせで取り除いてできる部分木について，比較対象の木T_1, T_2の間に同じ形のものがいくつ存在するかを数え上げたものを共通ルート部分木カーネル(2)と呼び，式(1)(2)で与えられる．

$$\kappa_R(T_1,T_2)=\prod_{i=1}^{d(v_r)}\left(\kappa_R\left(\tau\left(\mathrm{ch}_i(v_{r1})\right),\tau\left(\mathrm{ch}_i(v_{r2})\right)\right)+1\right),\quad \text{if } \mathrm{d}(v_{r1})\neq 0\wedge \mathrm{d}(v_{r1})=\mathrm{d}(v_{r2})\wedge \mathrm{lbl}(v_{r1})=\mathrm{lbl}(v_{r2}) \tag{1}$$

$$\kappa_R(T_1,T_2)=0,\text{otherwise} \tag{2}$$

ここで，$\mathrm{d}(v)$はノードvの出次数，$\mathrm{ch}_i(v)$はvのi番目の子，$\tau(v)$はvを根とする部分木，$\mathrm{lbl}(v)$はvのラベルである．またv_{r1}, v_{r2}はそれぞれT_1, T_2の根である．プログラムの構文木の場合，ラベルは演算子や識別子，構文などを識別するための文字列や数値となる．

3.3. 全部分木カーネル

それぞれの木に含まれるノードの全ての組み合わせに対する共通ルート部分木カーネルの総和を全部分木カーネル(2)と呼び，式(3)で表される．

$$\kappa_A(T_1,T_2)=\sum_{v_1\in T_1}\sum_{v_2\in T_2}\kappa_R(\tau(v_1),\tau(v_2)) \tag{3}$$

3.4. 余弦類似度

カーネルの定義から，2 つのデータ構造に対応する空間上のベクトルがなす角を調べることができる．式(4)はその余弦を表し，これを余弦類似度と呼ぶ．

$$\mathrm{sim}_{\cos}(T_1,T_2)=\frac{\kappa(T_1,T_2)}{\sqrt{\kappa(T_1,T_1)\kappa(T_2,T_2)}} \tag{4}$$

余弦類似度は，直感的にはT_1とT_2がどの程度似ているかを表しており，$T_1=T_2$であれば 1 となる．

3.5. カーネル重心法

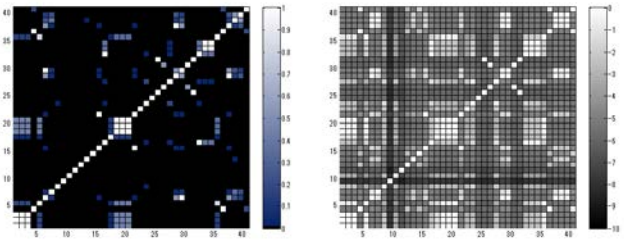
カーネルにより定義される空間上で，重心法によるクラスタリングを行うことを考える．クラスタG, Hの重心間の距離$d_{G,H}$は，余弦定理にカーネルを導入して整理すると式(5)で表される．

$$d_{G,H}{}^2=\frac{1}{N_G{}^2}\sum_{i=1}^{N_G}\sum_{j=1}^{N_G}\kappa(g_i,g_j)+\frac{1}{N_H{}^2}\sum_{i=1}^{N_H}\sum_{j=1}^{N_H}\kappa(h_i,h_j)-\frac{2}{N_G N_H}\sum_{i=1}^{N_G}\sum_{j=1}^{N_H}\kappa(g_i,h_j) \tag{5}$$

ここで，g_i, h_iは各クラスタを構成する要素，N_G, N_Hは各クラスタの要素数である．

4.　学生のプログラムの解析

3.に示した手法を実際のデータに適用する．ここで用いたデータは，神戸市立高専電子工学科 2 年のプログラミング演習課題で集められた学生のプログラム 40 個である．図 2 に全部分木カーネルによる余弦類似度，およびその常用対数を示す．対数を示す理由は，式(1)によりカーネルの値がプログラムのサイズに対して指数関数的に増加するためである．



(a) 余弦類似度　　(b) 余弦類似度の常用対数

図 2: 学生のプログラム同士の余弦類似度
（全部分木カーネルによる）

縦軸および横軸は学生の出席番号である．またマスの色は余弦類似度の大きさを表し，色が薄いほど類似度が大きい，つまり学生同士が同じようなプログラムを書いていると推測できる．また，これらのグラフは対角について対称である．

次に，図 2(b)に関してカーネル重心法を用いてクラスタリングを行い，生成されたクラスタの順に並べ替えたものを図 3 に示す．

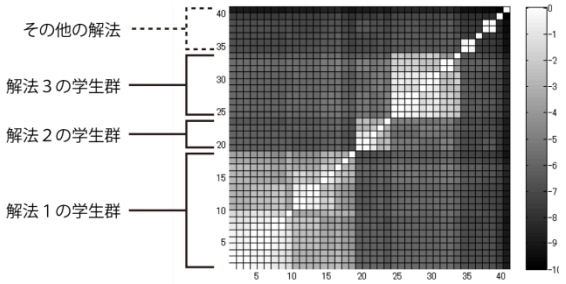


図 3: カーネル重心法によるクラスタリング結果

図 3 を見ると，類似度の高い値が矩形状に集合しているのが観察できる．これは，同じプログラムへの類似度が高いプログラム同士も類似度が高くなるという事実が現れており，同じ矩形に属する学生は同じ解法を採用しているものと推測できる．図 3 には比較的大きな矩形が 3 個観察できることより，課題に対する学生の回答が大きく 3 パターンに分類されていることが分かる．

5.　おわりに

従来法によるプログラム間の類似性の定義を見直し，カーネル法に基づく類似度および重心法クラスタリングを導入した．その結果，プログラム間の類似性を定量的に評価できるようになった．また学生のプログラムをいくつかのグループとして観察することが可能となった．

参考文献

(1) 井上 晴喜, 若林 茂: プログラム間の類似性に関する研究, JSiSE 第 31 回全国大会講演論文集, pp.401-402 (2006)
(2) J. Shawe-Taylor, N. Cristianini: Kernel Methods for Pattern Analysis, 共立出版, pp.472-482 (2010)