P2-03

プログラム挙動可視化システム TEDViT における アルゴリズム可視化機能の改良

Improving algorithm visualization function in the program behavior visualization system TEDViT

鈴木達稀[*1]，鈴木啓太[*1]，小暮悟[*1]，野口靖浩[*1]，山下浩一[*2]，山本頼弥[*2]，小西達裕[*1]，伊東幸宏[*3]
Tatsuki SUZUKI[*1], Keita SUZUKI[*1], Satoru KOGURE[*1], Yasuhiro NOGUCHI[*1], Koichi YAMASHITA[*2], Raiya YAMAMOTO[*2], Tatsuhiro KONISHI[*1] & Yukihiro ITOH[*3]
[*1] 静岡大学情報学部
[*1]Faculty of Informatics, Shizuoka University
[*2] 常葉大学経営学部
[*2]Faculty of Business Administration, Tokoha University
[*3] 静岡大学
[*3]Shizuoka University
Email: suzuki.tatsuki.20@shizuoka.ac.jp

あらまし: 初学者プログラミングを学習する際，プログラムやアルゴリズムの挙動をイメージできることが重要だとされているが，初学者がプログラムの挙動を正確にイメージすることは困難である．そこで，プログラムの挙動を教師が指定した可視化ルールに基づいて自由に可視化できる C 言語プログラム挙動可視化システム TEDViT が開発されている. 本 TEDViT ではルールの誤りを指摘する機能が一部のプロパティにしか対応していない，コネクタの表現方法が不十分である，2 次元配列を描画できないという問題を持っていた．本稿では，これらの問題を解決する手法を提案し，その効果を検証した．
キーワード： アルゴリズム理解，プログラミング言語教育，プログラム挙動可視化

1. 背景と目的

プログラミングを学習する際，プログラム・アルゴリズムの挙動をイメージすることが重要だとされている．しかし，プログラムの挙動を初学者が正確にイメージすることは困難である．この問題を解決するために，Jeliot3 [(1)] や PROVIT [(2)] など多くのプログラミング挙動可視化システムが多く開発さている．しかし，これらのツールでは開発者が定めた一定の方法でしか可視化できず，教師が可視化を制御できない．

この問題を解決するために我々はこれまでに C 言語プログラム挙動可視化システム TEDViT [(3)] を開発している．TEDViT では，教師が変数に対応する各オブジェクトのプロパティを自由に設定することで，それらのオブジェクトの座標や大きさなどを教師が意図したとおりに描画でき，教師が作成したプログラムを教師が指定した方法で可視化できる．

前述の通り，TEDViT は教師側が指定したルールでプログラムの挙動を 1 ステップごとに確認可能なため，アルゴリズムの学習にも役立つ．しかし，TEDViT ではアルゴリズムを学習する際に以下のような問題点がある．

問題点１　ルール誤りの発見が困難である
問題点２　コネクタの表現方法が不十分である
問題点３　２次元配列を描画することができない

この 3 つの問題点を解決することが本研究の目的である．解決手法については 3 章で述べる．

2. 関連研究と先行研究

2.1 関連研究

Jeliot3(for Java) [(1)] や PROVIT [(2)] は，学習者が自由に編集した Java プログラムの挙動を Step ごとに視覚的に再現できるツールである．学習者が入力したソースコードに対し，教師側が定めた描画ルールに基づいてデータの流れを示す矢印やアニメーションを用いて視覚的にプログラムの挙動を提示する．しかし，これらのツールは可視化の方法を教師の説明意図に応じて適切に変化させることはできない．

2.2 先行研究

TEDViT [(3)] は，C 言語を学習するプログラミング初学者を対象とした学習支援ツールである．TEDVIT のメイン画面は，実行するソースコードを表示させるソースコード部，実行時の変数などのメモリ情報を表示させる実装ビュー，プログラムの詳細な挙動を可視化する概念ビューからなる. TEDViT では実行中のプログラムの各変数の値が，それぞれの変数に対応するオブジェクトに自動で表示される．教師は CSV で記述する可視化ルールにより各変数に対応する各オブジェクトの背景色や文字色などを変化させたり，好きなタイミングでメッセージを表

示できる.

3.　問題点の改善策

3.1 問題点 1 について

TEDViT には，ルールの誤り箇所を出力する機能が一部プロパティにしか対応していない．そこでより多くのプロパティでルールの誤り箇所を出力する機能を実装することで解決する．ルールに誤りがあった場合に，誤りのあるルールの描画データとそのルールが記述されている CSV ファイル内の行数, プロパティ，正しいルールの記述の記述方法をコンソール画面に出力する. その出力を確認することで，ルールが記述されている CSV ファイル内から誤りがあるルールとプロパティを探し出せる．また，TEDViT のルールについて理解が進んでいない教師でもその出力を確認することでルール修正が可能となる.

3.2 問題点 2 について

TEDViT ではコネクタを直線でしか表現できず，可視化方法に制限があった．そこで，コネクタを折線でも表現できる機能を追加することで問題点 2 を解決する. 折線は 1 回, または 2 回屈折可能である. コネクタの方向は上下左右の 4 方向から指定することができ，意図した方向にコネクタを折線で描画できる.

また，コネクタが他のオブジェクトと重なってしまう問題や，コネクタの始点・終点となるオブジェクトを指定する際や同時に可視化するラベルを記述する際にルールの記述量が増加し，時間がかかってしまう問題が発生していたため，手動で位置を調整できる機能，自動で位置を調整する機能，コネクタにラベルを表示する機能，コネクタの表示順番を指定する機能についても設計・実装した.

3.3 問題点 3 について

TEDViT では 2 次元配列を描画することができず，2 次元配列を使ったアルゴリズムの可視化ができなかった．そこで 2 次元配列を描画できるよう改良した．また，表の複数行表示, 表見出しの表示についても実装を行い，動的計画法等の 2 次元配列を使用するアルゴリズムの可視化が可能になった.

4.　評価実験

問題点 1, 2 を解決できたことを確認するために評価実験を実施した．問題点 1 に対しては従来システムよりルールの修正時間が短縮できていることを確認するため，教員 2 名，TA 経験のある情報系修士学生 1 名，学部学生 1 名の計 4 名を被験者とし，実験を実施した．提示した手本と同じ動作をするよう，従来システムと新システムそれぞれでルールの修正を実施してもらい，修正にかかる時間を計測した．それぞれのシステムを使用した時の平均修正時間は，従来システムで 7 分 35 秒, 新システムで 5 分 5 秒であった．アンケートで新旧システムのどちらがルールを修正しやすいかを聞いたところ，新システムの方がルール修正をしやすいという意見が多く，教師の修正コストを削減できたと考えられる．今回の研究ではルール誤り箇所の出力をすべてのプロパティに対応させることができず，アンケート結果でもあったように一部プロパティではすべての記述方法に対応できていない．TEDViT で指定可能なプロパティや記述方法すべてに対応することでさらに教師の修正時間を短縮できると考えられる.

問題点 2 に対しては従来システムより理解度が上昇していることを確認するため，情報系学部生 7 名を被験者とし, 実験を実施した．新旧システムそれぞれでアルゴリズムの学習をしてもらい，その前後で理解度を測るテストを行った. 最高点 10 点のテストで，旧システム(4.5 点)の方が新システム(1.9 点)よりも点数の向上が良かったという結果となり，新システムによる学習効果は確認できなかった．しかし，新システムのコネクタは見やすいか，というアンケートには見やすいという意見が多く，視認性は向上したと考えられる.

問題点 3 に対しては新たに 2 次元配列を学習できるようシステムの拡張を行ったため，従来 TEDViT と比較しても効果は自明であるため，評価実験は実施していない.

5.　今後の展望

本研究では CSV ファイル内のルール誤り箇所をコンソール画面に出力する機能を作成した．またコネクタの記述箇所を一部簡略化できるようにした．これらによって，ルール修正時間の短縮できたと考える．しかし，現状のルール記述量でも一定以上の時間を要してしまうため，さらにルールの記述コストを削減する方法を考えていきたい.

また，プログラミング以外の分野のアルゴリズムを可視化できるように追加の描画機能の実装を現在進めている．今後はそれらのシステムの効果を検証しシステムの更なる改良していく予定である.

参考文献

(1) Mereno, A., Myller, N., Sutinen, E.: “Visualizing programs with Jeliot 3”, AVI 04: Proceedings of the working conference on Advanced visual interfaces, pp.373-376 (2004.5).
(2) 松村和哉, 渡部治郎, 寺内俊, He Aiguo, “PROVIT ソフトウェア可視化手法を用いた初心者向け C 言語教育ツール”, 電子情報通信学会技術研究報告，教育工学, 109(268), pp.41-46, (2009.11).
(3) Yamashita, K., Fujioka, R., Kogure, S., Noguchi, Y., Konishi, T. and Itoh, Y., “Classroom Practice for Understandings Pointers using Learning Support System for Visualizing Memory Image and Target Domain World”, Research and Practice in Technology Enhanced Learning (RPTEL), Vol. 12, No. 17, pp.116 (2017.9).