ロボットアシスタントを用いた教育支援システムの開発とプログラミング初学者のコーディング傾向の分析

平成 24 年 4 月 26 日受付

荻野晃大*
玉田春昭*
上田博唯*

要　旨

本稿では，大学におけるプログラミング初学者のプログラミング教育を支援するために開発したロボットシステム：フィノケーションの仕組みを述べる。今回のフィノケーションには，新たに導入した初学者のプログラミング傾向を定量的に分析する仕組みを追加したので，その仕組みについて述べる。また，フィノケーションを用いて収集した初学者のソースコードを，熟練者の目視によりプログラミング傾向を定性的に分析した結果を述べる。

キーワード：コミュニケーションロボット，プログラミング教育支援，ヒューマン・コンピュータインタラクション，メトリクス，リポジトリマイニング

1. はじめに

我々は，人とロボットとのインタラクションを行う一つのアプリケーションとして開発したプログラミングの教育を支援するロボットシステム：フィノケーション[1,5,6,7]を開発している。本稿では，フィノケーションに導入したプログラミングの初学者の陥りやすい誤りを把握する仕組みと，フィノケーションを用いて収集した結果の定性的な分析結果について述べる。

コンパイル作業を明示的に行う C 言語や Java 言語に関するプログラミングの初学者は，プログラミング言語の文法やライブラリ関数に関する理解度が低いため，頻繁にコンパイルエラーを起こす。またプログラミングの経験の少ない初学者においてコンパイルエラーは，エラーの真の原因と初学者の考える原因と乖離していることがあるため，なかなかエラーを解消することができない。その結果として初学者は，プログラミングの学習から敬遠してしまう恐れもある。そのため初学者がこのような誤りに陥っている場合，指導者は誤りを早急に把握し，ヒントを出したり，解説し直したりする必要がある。

一方，プログラミング演習では 1 人の教員と数名のティーチングアシスタントによって，多数の

* 京都産業大学コンピュータ理工学部

受講生を指導することが多い。そのため，全受講生の状況を逐次把握し，学習状況に応じた対応を行うことは困難であることが多い。初心者のプログラミングの内容を分析し，初心者の陥りやすい誤りの傾向を分析することは，指導者が適切に教育を行っていくうえで有益である。また，初学者に対して，事前に初学者の陥りやすい誤りの傾向を提示することは，初学者自身により，誤りを回避することもできる。

そこで我々は，プログラミングの初学者を対象に，初学者の理解度を推測と初学者の直感的な理解を支援する方法として，初学者の出すコンパイルエラーを収集・分析し，エラー内容を分かり易い言葉に翻訳して，ロボットを通して提示する仕組みとして，フィノケーションを開発している[1,5,6,7]。我々の開発しているフィノケーションでは，以下の 3 つの仕組みを持つ。

a) 初学者がシステムの利用に関して，特別の技術や知識を習得することなく，普段通りに使える仕組み
b) ロボットを用いて初学者のプログラミングに関する問題を解決する仕組み
c) 初学者のエラーの内容やその頻度等から初学者の理解度を分析し，教師に提示する仕組み

本稿では，フィノケーションシステムの(c)を改良する仕組みとして，ソースコードの編集に関わるメトリクス計測[2]を導入し，初学者のプログラミングの傾向を定量的に分析する仕組みを開発したので，その仕組みについて述べる。

また，フィノケーションの仕組みを用いて収集した初学者のコーディング過程を定性的に分析し，その傾向を明らかにする。分析にあたっては，初学者を対象としたプログラミング演習の中で，「ファイル保存時」や「ソースコードのコンパイル時」といった単位でのソースコードのスナップショットを取得する。そして，このスナップショットを対象に定性的分析を行った。定性的な分析は，コンパイル時のエラーメッセージとその際のソースコードを元に，初学者が課題の要求に沿った編集を行っていたかを目視で分析した。

2. フィノケーション

本節では，プログラミングの学習経験の少ない初学者へのロボットによる学習支援のシステム：フィノケーション[1,5,6,7]の仕組みと，今回の研究において改良した点について述べる。

(1) シナリオ

はじめに，フィノケーションを使った学生の学習シナリオを図 1 に示す。下記の数字と図 1 中の数字は対応している。

通常のプログラミングの実習は，(1)教師が初学者に対して課題の説明を行い，その後，(2)初学者は各自課題に取り組むという流れで進められる。この流れの中で，初学者は，(3)　課題のソースコードの編集，コンパイル，実行を行う。一般的に，初学者はコンパイルに成功するまでに何度もソースコード編集を繰り返すことになる。このときフィノケーションは，(4)初学者が出したコンパイル

エラーを収集し，(5)収集されたデータの分析を行う。そして，(6)全体の統計値を分析の結果として教師に提示する。教師はそれを受けて，(7)初学者に対して更なるヒントを与えることも可能である。また一方で，初学者個人に対する分析も行い，必要に応じて(8)ロボットに対して学生の支援のためのコントロール命令を送る。(9)ロボットはコンパイルエラーの翻訳を行い，解決手段を学生個人に対して提示する。その初学者が再度ソースコード編集，コンパイルを試み，(10)コンパイルに成功したとき，ロボットは初学者を褒める。

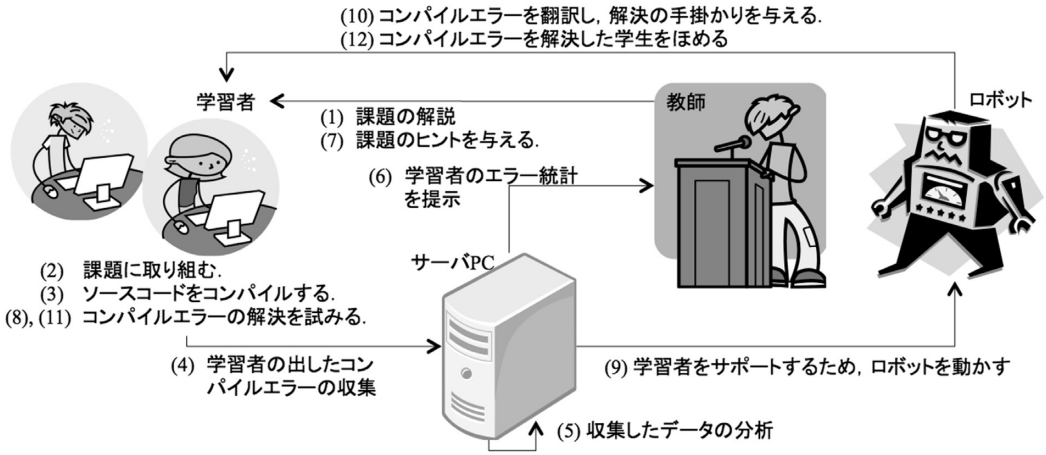

図1　フィノケーションによるプログラミングの学習支援の流れ

図1に示したような学習支援を実現するために，フィノケーションでは以下のような仕組みを持つシステムとして図2に示すようなシステムとしてデザインした。

(2) システム

初学者に対してプログラミングへの学習意欲を維持するためには，プログラムとは直接関係ない「英語」というハードルを下げ，英語によるエラーメッセージの意味を日本語によって説明することが重要であると考える。その結果として中上級者においては，英語によるエラーメッセージに基づいて，エラーを修正するようになると考える。また，初学者が生成するコンパイルエラーを収集，分析すれば，初学者が「どのような箇所において理解できていないのか」ということを教師は理解できる。したがってフィノケーションでは，エラーメッセージの収集とその翻訳により初学者を支援するというアプローチにより，初学者の学習支援を行う。また英語によるエラーメッセージを日本語に変化するという処理は，コンパイル作業が明示的に行われるC言語やJava言語などの他のプログラミング言語に用いることができる。

フィノケーション[1,5,6,7]は，プログラミングの初学者のロボットによる学習支援と教師への初学者の理解度情報の提供を実現するために以下のような仕組みを持つ。

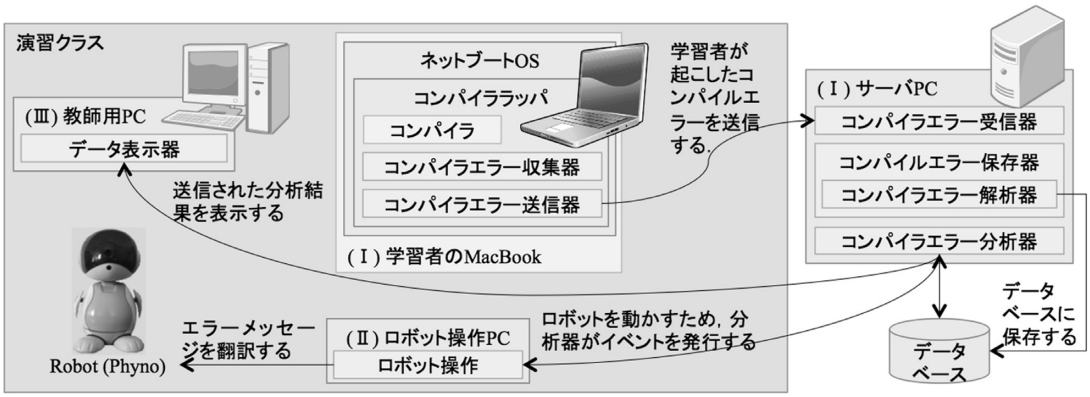

図 2　フィノケーションのシステム概要図

(a) 初学者にシステムの利用を意識させない仕組み(図 2 中のⅠ)

プログラミングの初学者は，コンピュータに関しても初学者であることが多い。そのため初学者は，プログラミングに必要な作業と，システムを動かすための作業を切り分けることが難しい。提案システムを使用するためにいくつかの動作やコマンドを必要とすると，初学者は「プログラミングに必要な知識なのか？」,「提案システムを使うために必要な知識なのか？」ということを意識しながら作業する必要がある。そのため，初学者のプログラミングの学習効率を落としてしまう。初学者に「提案システムを使っている」という感覚を意識させずに，普段通りのプログラミング学習で，提案システムによる学習支援を受ける事のできる仕組みが必要である。

フィノケーションでは，初学者に提案システムを意識させずに使える仕組みとして，Mac OS X のネットブートを用いた。本学部では，全学生が MacBook を保持しているためである。ネットブートシステムは，サーバ上になる OS イメージを各初学者の保持する MacBook にロードさせて起動する。ネットブートの利点は，学生の MacBook は管理者側で完全調整された OS とソフトウェアで動作するため，あたかも本学が設置した PC のように利用できる点にある。この機能を利用して，初学者の個人の MacBook にロードさせる OS に各個人のエラーを収集し，サーバに送信するスクリプトを組み込んでおけば，初学者はいつも通りにプログラミングすることで，本提案システムによるサポートを受ける事ができる。

(b) ロボットによる支援の仕組み

プログラミングの初学者は，教師から与えられた課題に対して試行錯誤して解くことを繰り返し，プログラミング技術を習得する。しかし初学者は，コンパイラの出すエラーメッセージを理解できなかったり，エラーメッセージの意味を取り違えたりする事が多い。その結果初学者は，必要以上にプログラムに対して苦手意識を持ってしまい，自分で学習を進めることを諦めてしまう。したがって，初学者がプログラミングに対して苦手意識を持つことなく，学習意欲を保つための支援が必要である。

本研究では，初学者には難しいと思われるエラーメッセージを，より初学者に分かりやすく，修正方法のヒントやアドバイスを示す。これにより，エラーメッセージの意味がわからなかったため，学習を諦める初学者をフォローアップできる。

本研究では，この初学者の課題解決を支援するため，ロボットを導入する。ロボットは，初学者のプログラミングに関する行動とその結果(例えば，コンパイルとその失敗など)に対して，コミカルな仕草とともに解決のヒントを初学者に提供する。プログラミング支援にロボットを用いる理由は，初学者に対して単に日本語によりヒントを与えるだけでなく，ロボットによる様々な仕草も加えることにより，初学者のプログラミングへの興味とモチベーションの維持を目的としている。ロボットに予め様々な仕草を登録しておき，初学者が特定の条件を満たしたときに，特定の仕草をロボットが行う。このように，条件によってロボットに異なる仕草をさせることにより，エンターテイメントの要素も導入する。本研究では，学習支援に用いるロボットとして Phyno[3]を採用する。Phyno は 3 歳児をモデルとした子供ロボットであり，可愛らしい仕草を行うようデザインされている。この可愛らしさが注目を集め，たとえ画一的な応答しかできないロボットであっても，親しみを覚えることがわかっている[4]。また，初学者が課題に取り組んでいる最中に，ロボットが割り込んでアドバイスをすることになる。そのため，初学者に気づいてもらうため，大きな仕草をさせる設計とした。

(c) 初学者の理解度を分析する仕組み

教師には，初学者の理解度により教える内容の細かさを変えたいときが往々にしてある。従来であれば，初学者の理解度は定期試験やレポートなどの大きな節目でなければ，測れなかった。しかしこのタイミングでの理解度チェックでは，プログラミングに関して学習意欲を無くした学生のフォローアップには手遅れである。そのため，初学者のプログラミング行動に基づいたデータをリアルタイムに収集し，教師が初学者の理解度に適した授業内容を構成する支援が必要である。したがって本研究では，プログラミング教育における教師への支援として，初学者のプログラミングの行動(コンパイル)に関するデータを収集し，その統計値を提示するものとした。

(3) フィノケーションの改良

改良前のフィノケーションでは，エラーの種類やコンパイル回数などの簡単なエラーによる支援のみを行っていた。本研究では，参考文献[2]で用いられている「ソースコードの編集に関わるメトリクス計測に基づく定量的分析」をフィノケーションシステムに導入した。

学習者のソースコード編集履歴をより細粒度に計測する方法として，「トークン編集距離」と呼ぶメトリクスをフィノケーションシステムの機能Ⅲに導入した。トークン編集距離は 2 つのソースコードを入力とし，字句解析を行った後，2 ソースコード間の 2 つの文字列間の不一致度を表す値(レーベンシュタイン距離)を算出することで求める。具体的には一方の文字列から他方の文字列に変形するのに要した文字の挿入，削除回数の最小値で求められる。トークン編集距離を求めるにあたっては，

ソースコードを字句解析しソースコードのトークン化を行う。本研究で用いたトークン化は下記のルールに従う。

- 空白文字，コメントは全て無視する。
- 識別子，予約語，リテラルは 1 文字に符号化し，その他はそのままとする。
- トークンの種類(型名やリテラル)やスコープの違いは考慮しない。

例えば初学者が課題の内容理解に手間取っている場合には，ソースコードに対する編集が一切行われていないことが考えられる。また，編集を繰り返しているのに正解となるソースコードとのトークン編集距離が変わらない場合, 初学者は誤った修正を行っている可能性が考えられる。このようにソースコードとその編集履歴を対象に計測できる定量的な値を時系列に観測することにより，学習者の迷いなどを検出できるようにした。

3. 初学者のプログラミング傾向の定性的な分析

本稿では，プログラミング言語の演習において,「学習者がどのように演習の課題に取り組んでいるか」をコーディング傾向と呼ぶ。本稿では初学者のコーディング傾向を明らかにするため，実際にプログラミング演習で収集されたデータを定性的な分析を行なった。

定性的な分析では，プログラミング初学者が具体的にどのようにして課題に取り組んでいるかを，十分な経験を持つプログラミング熟練者によって目視で分析する。分析にあたっては，コンパイルを実行した際のコンパイル対象のソースコードと，コンパイルの結果(エラーメッセージ)を用いた。この 2 つの入力をもとに,「修正の意味的内容」と「修正の妥当性」をプログラミング熟練者によって判断した。この判断結果を用いて，妥当な修正が行われなかったとき初学者はどのような誤りに陥っていたか，その修正の内容を確認する。

フィノケーションシステムをもちいて，京都産業大学の情報系学部 3 年生を対象にプログラミングのデータを取得した。分析者は C 言語を対象とした初学者向けプログラミング演習における指導経験を有している。分析に際しては，各ソースコードのスナップショットに対して下記の情報を目視で確認した。

- 着目しているスナップショット(ソースコード)
- GNU diff を利用した前回編集時との差分情報
- コンパイル結果(エラーメッセージ)
- プログラム実行結果(コンパイルに成功した時のみ)

上記 4 種類の情報より，下記に示す 3 つの観点から分析を行った。

- 不具合原因：ソースコードに不具合が含まれる場合，その内容を自然言語により記述した。
- 修正内容：diff の内容からどのような修正を行ったか，自然言語により記述した。
- 修正の妥当性：修正内容が課題の意図に沿ったものになっているか。もしくは，ソースコードの

表 1　定性的な分析の結果

ユーザ名	課題数	完成課題数	編集回数 妥当な編集	欠陥混入	合　計
A1	10	3	20	38	92
A2	10	10	19	7	44
A3	10	10	15	13	24

不具合を混入しているか。

表 1 に各学習者が取り組んだ課題数とその中で完成に至ったものの数，および編集回数を示す。また編集の中で，分析者が題意に沿って編集を行っている，もしくはプログラム中に欠陥を混入するような修正をしていると判断したものについて，その回数を示している。学習者は，C 言語の学習前に Java によるプログラミング演習の授業を履修していた。また，ユーザ A2 に関しては Ruby などのスクリプト言語を事前に自習していた。これらの点を踏まえ，以下に分析の結果見られた傾向について示す。

• 修正すべき箇所とは関係ない箇所を修正し続ける

特にユーザ A1 で多く見られた事象として，本来修正すべき箇所には手を加えず，課題内容とは関係の無い箇所を編集していることが確認された。プログラミング熟練者であれば，エラーメッセージなどをもとに修正箇所の絞り込みを行うことが考えられる。しかし，初学者の場合はエラーメッセージの情報を十分に理解できず，修正箇所を突き止めることが困難であるため試行錯誤的に修正を行っていると考えられる。このような初学者に対しては，コンパイラが出力するエラーメッセージの意味と，具体的な修正箇所を絞り込むための戦略をあらかじめ教育しておくことが有用であると考えられる。

• 自分の有しているプログラミング言語の知識に引きずられている

他の言語で利用できる機能や文法を C 言語でも利用しようとしてエラーを引き起こしている傾向が観測された。その中でも多く見られたのが文字列の結合である。Java であれば String 型の文字列は加算演算子により結合することが可能である。しかし，C 言語では加算演算子による文字列の結合はサポートされていない。初学者はこれまでに自分が獲得したプログラミング言語の仕様と異なることを理解するまでに，プログラミング熟練者よりも時間がかかると思われる。既に他の言語を学習した経験のある者に対しては，習得済み言語との相違点を事前に提示することで，効率的に教育できる可能性がある。

4. まとめ

本稿では，人とロボットとのインタラクションを行う一つのアプリケーションとして開発したプログラミングの教育を支援するロボットシステムを用いて，プログラミングの初学者の陥りやすい誤りを把握する仕組みと，ロボットシステムを用いて収集した結果の定性的な分析結果について述べた。

我々の開発したフィノケーションシステムを用いて収集したプログラミング学習者のプログラミング傾向を定性的に分析した結果，学習者が陥りやすい行動パターンを観測できた。これにより，フィノケーションシステムは，プログラミング学習者のプログラミング傾向を導出するためのデータを収集できることを示すことができた。また，学習者が誤った修正を行っている可能性を自動的に導出するための方法をフィノケーションに導入した。これにより，学習者のプログラミング中の行動から，その学習者の理解度を推定することができるようになった。

今後の課題としては，フィノケーションに導入したプログラミング傾向を定量的に計測する仕組みの評価とその改良を進めていく予定である。

参考文献

[1] Akihiro Ogino, Haruaki Tamada, and Hirotada Ueda, “A Prototyping of a Teaching Assistant Robot for C Language Class,” in *Universal Access in Human-Computer Interaction. Applications and Services - 6th International Conference (UAHCI 2011),* vol. 6768, 2011, pp. 597-604.

[2] 上田博唯，近間正樹，佐竹純二，佐藤淳，and 木戸出正継，“ユビキタスホームにおける対話インタフェースロボットの試作，” in *情報処理学会研究報告．UBI，［ユビキタスコンピューティングシステム］*，vol. 28, 2005, pp. 239-246.

[3] 伏田享平，玉田春昭，井垣宏，藤原賢二，and 吉田則裕，“プログラミング演習における初学者を対象としたコーディング傾向の分析，” in *信学技報　ソフトウェアサイエンス研究会*，vol. 2012-03-SS, 2012, pp. 67-72.

[4] 松本斉子，上田博唯，山崎達也，and 徃住彰文，“共生ロボットに対するコンパニオン・モデルの形成：ホームユビキタス環境における生活実証実験から，” in *ヒューマンインタフェース学会論文誌*，vol. 10(1)，2008, pp. 21-36.

[5] Haruaki Tamada, Akihiro Ogino, and Hirotada Ueda, “Robot Helps Teachers for Education of the C Language Beginners,” in *Human-Computer Interaction. Novel Interaction Methods and Techniques,* vol. 5611, 2009, pp. 377-384.

[6] Haruaki Tamada, Akihiro Ogino, and Hirotada Ueda, “A Framework for Programming Process Measurement and Compiling Error Interpretation for Novice Programmers,” in *Software Measurement, 2011 Joint Conference of the 21st Int'l Workshop on and 6th Int'l Conference on Software Process and Product Measurement (IWSM-MENSURA)Date of Conference : 3-4 Nov. 2011,* 2011, pp. 233-238.

[7] 玉田春昭, 荻野晃大, 上田博唯, “アシスタントロボットを用いたプログラミング教育支援システムの構築，” *信学技報マルチメディア・仮想環境基礎研究会*，vol. MVE2010-48, 2010, pp. 143-148.

A Coding Pattern Analysis for Novice Programmers using the Learning Support System with a Communication Robot

Akihiro OGINO
Haruaki TAMADA
Hirotada UEDA

Abstract

In this paper, we describe the Phynocation that is a learning support system with a communication robot for programing. The new version Phynocation has a function that analyzes a programing trend of students. This paper describes the new function that measures metrics related with editing source code. This paper also describes the result of qualitative analysis based on observation by experienced programmers.

Keywords : Commutation Robot, Education Support for Programing Skill, Human-Computer Interaction, Metrics, Repository Mining