コンピュータと教育　61−6
（２００１．１０．１９）

プログラムの正しさの理解を目的とした教材作成システム

A System for Learning and Understanding Program Correctness

森　正　　角川 裕次　　阿江 忠
Tadashi Mori, Hirotsugu Kakugawa, Tadashi Ae
広島大学大学院工学研究科情報工学専攻
Information Engineering, Graduate School of Engineering, Hiroshima University

概 要

あらゆる所でコンピュータが使われている昨今において，一つ一つのプログラムを正しく書ける事はプログラマにとってますます重要な能力となっている。一方，このような能力を身に付けるには，「プログラミング」，「コンパイル・実行」，「デバッグ」といった行動が不可欠であり，学習者のこういった行動も同時に支援する必要がある。

そこで我々は，プログラムの正しさの理解を目的とした教材を，WWW 上に作成するシステムを構築している。システムは，「トップダウン的なプログラムの正しさの理解」，「自由記述形式の虫食い問題」，「教材プログラムの実行」，「表明による解答のチェック」，「学習者の実行履歴自動取得」といった特徴を持つ。また，教師がこれまでに手作業で行っていた作業の自動化を試みる。

Today, we can see computers everywhere, on which many softwares are running. Essencially, it becomes very important writing correct programs more and more. At the same time, in order to learn the skills to write them, programmers should have the oppotunities of several actions like “coding,” “executing it,” “debugging it,” and so on.

We are developping a system to learn the correctness of computer programs. It has the features which make students understand it, and give the oppotunities to learn the skills for writing correct programs. The system also provide some automation which helps teachers to concentrate the contents of teaching materials.

1　はじめに

今日，コンピュータはあらゆるところで利用されている．加えて，その上で動くソフトウェアは年々大規模化し，ソフトウェア障害が重大な結果を引き起こす場合も数多く存在する．このような状況においては，一つ一つのプログラムを正しく書ける事が，以前にも増してプログラマに要求される．当然，そのような良いプログラマのための教育，プログラムを正しく書くための教育は重要な位置を占める事になる．

そこで我々は，プログラムの正しさを学習者に理解させることを目的とした教材を作成するシステムを構築している．システムは，プログラミング言語 Java によって記述されたソースコードとその解説が混在した XML 形式の入力ファイルから，適宜必要な部分を抜き出しつつ，World Wide Web 上に教材を提示する．入力ファイルは教師が書くことを前提としており，学習者に必要なのは汎用的な Web ブラウザと，Java の実行環境，それにいくつかの用意されたクラスライブラリだけである．

本稿では，本研究で構築している「プログラムの正しさの理解を目的とした教材作成システム」について，その概観，特徴，利点，今後の見通し等の報告を行う．

1.1　関連研究

プログラムの理解や記述，プログラミング教育に関する研究はこれまでにも数多くなされて来た．

Donald E. Knuth は自身の提案した文芸的プログラミングのために，WEB システムを構築した [1]．文芸的プログラミングにおいて，プログラマはプログラムとそのソースコードに対する解説を混在させて記述し，

システムは，その解説とソースコードを利用してドキュメントを出力する．ドキュメントは，ソースコードにおけるアウトラインから詳細部への一種のハイパーリンクを含み，人が読んで理解するのに適切な構造を持たせることが可能となっている．

K. Halewood らは，プログラムの構造を NS ダイヤグラムを用いて記述していくことでプログラミング言語 Pascal のプログラミングをトップダウン的に行うシステムを提案している [2]．また ZStep95[3] やプログラム紙芝居 [4] といった研究においては，ソースコード上の実行点を動的に可視化し，追跡する，もしくは実行点の履歴を追跡することで，プログラムの動作の理解を助けるシステムが提案されている．

これらの研究においては，ソースコードを読んで理解する，理解しやすい単位や順序でプログラムを表示するといった事は注目されているが，本当にそのプログラムが思った様な動作を行うのかという，「プログラムの正しさ」を理解させるための工夫は特になされていない．

アルゴリズムを何らかの形で可視化しようという研究も数多く見受けられる．

POLKA は C++と X Window System を用いて実装された並列プログラムを可視化するシステムである [6]．POLKA はアルゴリズムを 2 次元や 3 次元アニメーションとして視覚化する枠組を提供する．また，Zeus も並列アルゴリズムの可視化システムである [5]．利用者がアルゴリズムの基本的な動きをシステムに与えることで，いくつもの異なる視点からのアルゴリズムの視覚化を行う．

これらの研究では，アニメーション等による，アルゴリズムの直観的な理解は意図しているが，その動きが本当に正しいのかを理解するには不十分であると考えられる．

一方，プログラムのソースコードやその他の情報を入力として与え，正しいプログラムであることを判定するという試みも数多くなされている．

Adam,A. らが提案しているシステム LAURA においては，問題に対して理想的な解法を一つ入力として与え，学習者のプログラムと比較することで学習者の解法が正しいかどうかが判定される [7]．

W.L.Johnson によって提案されているシステム PROUST は，その入力としてプログラマが何を意図して書いたプログラムなのかについての情報を与えるところを特徴としている [8]．システムは与えられた意図 (ゴール) から，あり得る中間的なゴールを割り出し，入力として与えられたソースコードに存在するバグを検出しようとする．

これらの研究では，プログラムの正しさを検証するためのアイデアを内部に利用しているが，学習者にそれを意識させ，考えさせるといった事は意図していない．

2　システムの構成

本研究ではプログラムの正しさを学習するのに必要な，もしくはあるのが望ましい要素として，次の 4 つを挙げる．

1. 人間の理解しやすい順序，単位で教材を記述，表示を行う．
2. 個々の学習者に正しいプログラムについて考える機会を与える．
3. プログラムの正しさを理解するための情報を，教材を通して明示する．
4. 教材として提示されたソースコードが確かに正しく動作する事を学習者が確認出来る．

我々は，上記の 4 つの要素の両立を目指し，システムを構築してきた．この節では，そのシステムについての説明を，順次行っていく．

2.1　システムの全体像

システムの全体像を図 1 に示す．

システムは大きく分けて二つの部分，学習ページ及びソースコード生成部と実行履歴取得サーバからなっている．それではこれらについて解説を行っていく．

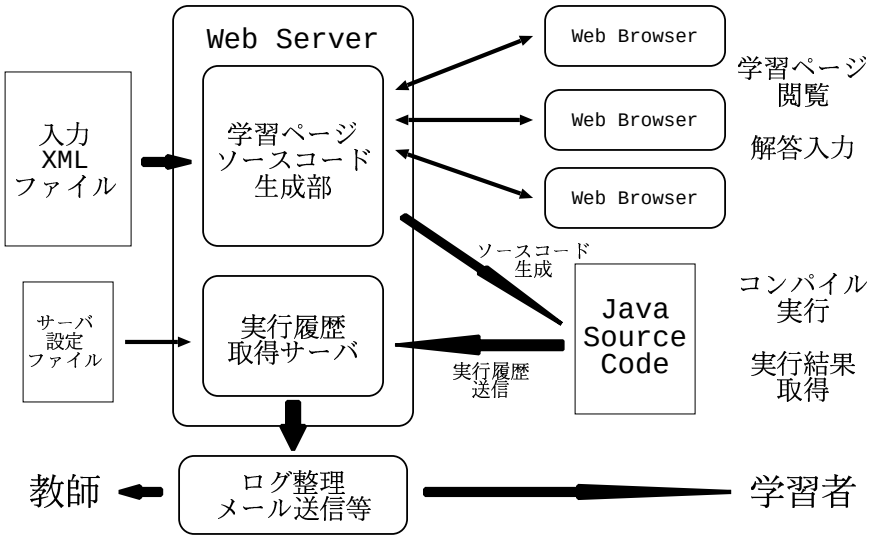

図 1: システムの全体像

2.2 学習ページ及びプログラムコード生成プログラム

2.2.1 システムの特徴及び，利用の流れ

システムの特徴をまとめると次の様になる．

1. アウトラインから詳細部への，トップダウン的な教材を作成出来る．
2. プログラムの一部を埋めるタイプの問題を利用出来る．
3. プログラムに対する付加された表明のサポートと，それらを利用した解答判定が可能．
4. 教材に利用されるプログラムを，学習者が実際に実行出来る．

実際のシステム利用の流れは次の通りである．

1. 教材の元となる入力ファイルを教師が記述．
2. 学習者はシステムが作成する教材ページに Web ブラウザを使ってアクセスし学習，解答すべき所があれば解答を入力．
3. システムは解答判定用のコードを付加した Java のソースコードを生成．
4. 学習者は生成されたコードをダウンロードし，コンパイル．
5. 学習者はコンパイルしたプログラムを実行し，正誤判定を含めた実行結果を得る．

本システムの教材ウェブページ生成部及び学習用ソースコード生成部は，Java 言語のサーバ実行環境である Servlet を用いて実装を行い，システムの入力ファイルの記述フォーマットとしては XML を採用した．また，学習用プログラムの実行はクライアント側で行われる事を想定しているため，クライアント側にも Java 言語の実行環境が必要となる．

2.2.2 プログラムの正しさの理解について

ここで，2節の最初に挙げた，我々がプログラムの正しさを学習するのに必要な，もしくはあるのが望ましいと考える4つの要素に対する本研究ののアプローチについて述べていく．

- 人間の理解しやすい順序，単位での教材の記述，表示

  本システムではプログラミング言語 Java で記述されたプログラムのソースコードを基にした教材を用いて学習を進めていく．ここで我々は，人がプログラムを理解していく過程に注目した．

  我々がプログラムが何をしているのかを理解していく際に，その詳細部をいきなり理解出来ることはまれである．普通，我々はまずそのプログラムが大まかに何を行っているのかを理解してから，その詳細へと理解を進めていくと考えられる．

  そこで，本システムにおいても，学習者がアウトラインから詳細部へと理解を進められる教材を作成できる様にすることにした．具体的には，システムはプログラムの説明ページをアウトラインから詳細部へとハイパーリンクを使って辿ることの出来る複数のウェブページから構成する．各ページには表示している内容に対する説明を記述する．

  同時に，本システムは教材の表示，ページ生成に関して，以下の様な付加的な機能を持つ．

  - 教材における，任意のある部分に対するリンクを作成する機能．
  - 全体のページの関係を示すメニューを表示する機能．
  - 小段落単位でページを切り出す機能．
  - アブストラクトを与える機能．

- 正しいプログラムについて考える機会

  現在，World Wide Web 上において実現されている教育システムの多くは選択式の解答しか扱うことが出来ない．しかしこれでは，学習者にプログラムの正しさについて考えさせるのに十分であるとは言えない．そこで，本システムではこのような選択式の解答だけでなく，空欄にコードの一部を入力させるような教材を作成できる様にした．

  このような解答を自由に記入出来る形式の問いの場合，その正誤判定をどうやって行うかが問題となる．本システムでは対象となる Java のソースコードに，後述するプリコンディション，ポストコンディション，ループインバリアントをチェックするコードを付加する．これらのコードは，いくつかの入力値に対するプログラムの実行の際に，これらの条件の真偽をチェックする．

- プログラムの正しさを理解するための情報の明示

  プログラムの正しさを学習者が確認するのに必要な情報として，プログラムに付加された表明，特にプリコンディション，ポストコンディション，ループインバリアントがある．本システムではこれらを，プログラムの正しさを理解するのに必要な基本的な情報と位置付け，これらを教材の中に明示することが出来る様にした．

  また，本研究ではこれらの表明を利用して，学習者の入力内容の正誤判定を行うことを考える．本システムでは後で説明する様に，教材として使われているプログラムを学習者に提供し，実際に動かすことが出来る．そこで，これらの条件をチェックするコードを対象となる Java のコードに追加し，学習者が入力したコードも含めてプログラムを実行した際，プログラムの変数値を基に，3つの条件が確かに真になっているかどうかをチェックする．

- プログラムが正しく動作すること確認

  プログラムが正しく動作することを確認するには学習者がプログラムを実際に走らせ，正しい実行結果を得ることが出来るのが最良であると考えた．そこでシステムに，教材の中で使われているプログラムのソースコードを抜き出す機能を持たせ，必要なコードを付加した形で学習者に提供する事にした．学習者はこれをダウンロードし，コンパイル，実行することで実行結果を得ることが出来る．またその際に，問題を設定した教材においては，学習者が記入した解答に対する，上記の表明を利用した判定も行う．

2.3 使用例

ここで，実際のシステムの使用例を示す．現時点で，以下の様な教材をサンプルとして作成した．

- ビット演算を用いて掛け算を求める例
- GCD を求める例
- 挿入ソートの例
- シェルソートの例
- クイックソートの例

ここでは，その中の挿入ソートの教材を使って，実際の使用例を示す．

1. 正しさの理解

   挿入ソートの例においては，実際にソートを行うメソッド insertionSort() の正しさが全てと言っても過言ではない．そのためこの例では，学習者が insertionSort() の項から学習を始めるように導かれる．学習者はメソッド insertionSort() の項を，2.2.2 節に示したように，そのアウトラインから詳細部へと理解を進めていく，具体的には，挿入ソートの場合，二重のループの動作によってソーティングを行っていく．ここで，学習者が教材に示されている道筋に従ったとすると，次のように学習を進めていくと考えられる．

   (a) 学習者はまず，メソッド insertionSort() のアウトラインのページで，このメソッドのプリコンディション，ポストコンディションを確認し，これらが満たされれば確かにソーティングが正しく行われていることを確認する (図 2)．

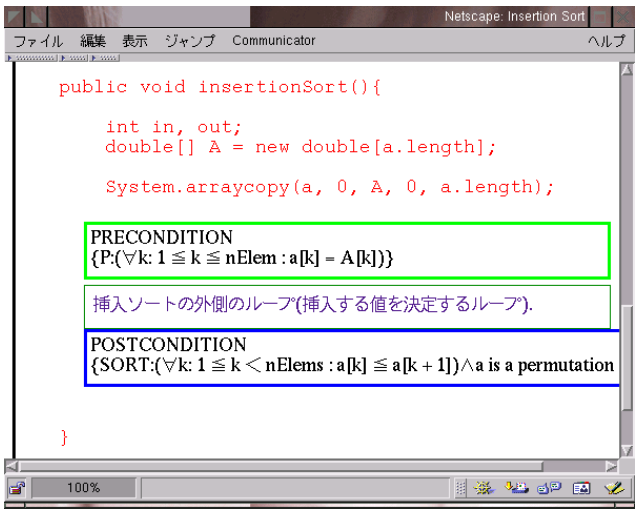

図 2: 教材の表示画面 (アウトラインレベル)

   (b) ハイパーリンクを辿って，外側のループに関する項に進む．ここでは外側のループに関するインバリアントが提示されている (図 3)．学習者はこのインバリアントがループの実行中，常に真であることを確認すると同時に，ループを抜ける際，一つ前で確認したポストコンディションを真にすることを確認する．

   このように，学習を行うべき所を教材が導きつつ．学習者は「何が成り立てばプログラムは正しいのか」を意識しながら学習を進めていく．

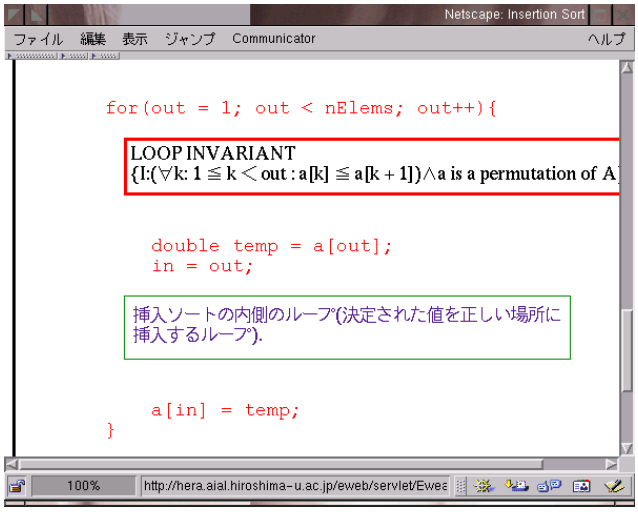

図 3: 教材の表示画面 (より詳細なレベル)

2. 問題の解答

次に学習者は，より内側のループへと進む．内側のループにはそのガードを答える問題が設定してあり (図 4)，学習者はインバリアントやアルゴリズムの動きをヒントにしながら問題に答える．問題は複数設定することが出来る．システムは前に入力した解答も保持し，学習者は全ての問題に対して解答を入力する．

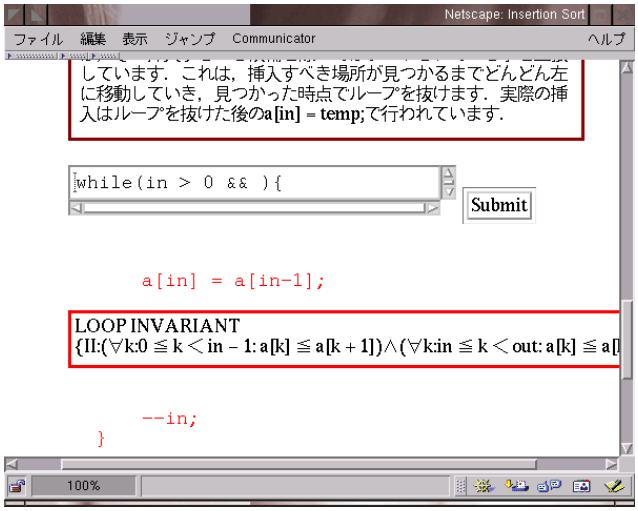

図 4: 解答の入力

3. プログラムの実行

入力が終ると学習者は教材で使われているプログラムをダウンロードすることが出来る．学習者が解答した部分については，システムが解答をソースコードに反映させた形で提供する．学習者はこのプログラムをコンパイル，実行して結果を得る．その際，教材内のプリコンディション，ポストコンディション，ループインバリアントの真偽も出力される．また，この問題では，内側のループの問いが内側のループのインバリアントに対応付けられており，その真偽によってその解答は正しかったのかが判断される．これらを繰り返して，学習者は正解を得るまで学習を続ける事になる．

実際にはこの後，学習者が確かに学習を行い，正解を得たことを教師に報告するため，3 節で述べるよ

うな実行履歴取得サーバを利用して，実行履歴の送信が行われる．学習者は，プログラムをあらかじめ教師が定めた時間に実行し，ログをサーバに送る．ロギングの〆切時間が来て，きちんとロギングを受け付けた事をメールで知らさせれれば，学習は終了である．

3 実行履歴取得サーバ

3.1 モチベーション

上述の通り，本システムでは生成したプログラムが学習者のコンピュータ上，つまりクライアント側でコンパイルされ，実行されることを前提としている．しかし，単にクライアント側で実行を行うだけでは，教師には学習者が行った学習に関して情報を得る手段が無い．そこで，本研究では，システムが吐き出す Java のコードにサーバプログラムと通信を行うコードを付加し，解答判定の結果を含む実行履歴を教師に自動的に提供する仕組みを考える．これにより教師は，学習者が本当に学習を行ったのかの確認や，課題の提出を受け付けた事を学習者に伝えるといった作業を自動化することが出来る．

3.2 開発の方針

開発の方針としては，次の 2 つを掲げている．

- イベント駆動型で動作する．
- サーバの基本となる部分の実装と，具体的な機能の実装を分ける．

3.3 サーバプログラム

実際に我々が作成したサーバは，システムの他の部分と同様 Java 言語で実装され，主に次の二つの部分から構成されている．

1. イベント処理の機能やサーバの基本的な枠組みを提供するクラス．
2. サーバ実装のサンプルとして，メールの送信や学習者からのログを整理する機能を提供するクラス．

それでは，各々について，説明を行っていく．

3.3.1 サーバの基本機能

我々はまず，実行履歴取得サーバの基本的な機能を提供するため,「ある時刻になる」,「学習者からログが送られて来る」といった際に，イベントを発生させて動作するサーバの枠組を提供するクラスを作成した．学習者からのロギングに関する実装は，プログラムのロギングを行うのに必要な機能を提供するクラスライブラリである Log4j[10] を用いて行った．これらのクラスを利用して，時刻や学習者からのロギングを利用して何らかの動作を行う実行履歴取得サーバを作成することが出来る．

3.3.2 実行履歴取得サーバのサンプル

本研究では同時に，実行履歴取得サーバのサンプルとして，以下の様な機能を提供するクラス MyServer を作成した．

1. 教師がサーバを起動すると，学習者を登録すると共に，その旨を告げるメールを各学習者に送る．
2. サーバは設定された時間内にだけ，クライアントからのログを受け取り，XML の形でサーバ側に保存する．

3. 受付時間を終了した時点でまだ実行ログを受け取っていない学習者に対し，確認のメールを送付する．
4. 同時に教師に対し，終了時点での学習者の提出状況を知らせるメールを送付する．

教師がこのサーバを利用したい場合，2.2.1 節における利用の流れに次の様な内容が追加される．

1. 教師は教材の入力ファイルと同時に，ロギングを受け付ける時刻や教師，学習者の ID(e-mail アドレス)，その他の情報を設定 XML ファイルに記述．
2. システムが解答判定のためのコードを付加する際に，サーバへの実行履歴の送信用のコードも付加し，学習者用の Java のソースコードを生成．
3. 学習者がプログラムを実行すると，送信用のコードが実行結果をサーバ側に送る．
4. サーバは時刻やリモートのマシンからのロギングといったイベントによって，教師や生徒にメールを送るなどの動作を行う．

また，学習者から送られてきたログは XML の形でサーバ上に保存されているため，XSLT との組み合わせにより，Web ページとして閲覧することが出来る．

この実行履歴取得サーバは，Log4j の他に，メール送信のための Java Mail API[11]，XML と XSLT による Web ページの生成のための Cocoon[12] といった既存のクラスライブラリを利用している．

同時に，学習者側からのリモートのロギングを行うため，学習者側にも Log4j が必要である．

4 おわりに

本稿では，我々が構築してきたプログラムの正しさを学習するための教材作成システムについて報告を行った．また，リモートで学習者が行ったプログラムの実行をサーバ側でチェック出来る様にするための，実行履歴取得サーバについても説明を行った．現在システムに関しては，教材内でリンクを張る機能や，学習者に提供するプログラムにコードを付加する機能の実装を行っている．今後の課題としてはシステムの実装の完了や見直し，また，実際の運用や評価といったものが挙げられる．

参考文献

[1] Donald E. Knuth, 文芸的プログラミング, アスキー出版, 1992

[2] K. Halewood, M. R. Woodward, A Uniform Graphical View of the Program Construction Process: GRIPSE *Int. J. Man-Machine Studies*, **38**, pp.805–837, 1993

[3] ZStep95, *http://lieber.www.media.mit.edu/people/lieber/Lieberary/ZStep/ZStep.html*

[4] プログラム紙芝居，*http://www.sanpo.t.u-tokyo.ac.jp/~terada/kamishibai/index.html* ,2000.

[5] ZEUS, *http://www.research.compaq.com/SRC/zeus/home.html*

[6] POLKA, *http://www.cc.gatech.edu/gvu/softviz/parviz/polka.html*

[7] Anne Adam, Jean-Pierre Laurent. LAURA, a system to debug student programs. *Artificial Intelligence*, **15**, pp.75–122, 1980.

[8] W. Lewis Johnson, Understanding and Debugging Novice Programs, *Artificial Intelligence*, **42**, pp.51–97, 1990

[9] N. Wirth, Program Development by Stepwise Refinement, *Communications of the ACM*, **14**, No. 4 pp.221–227, 1971

[10] Log4j Project, *http://jakarta.apache.org/log4j/docs/index.html*

[11] JAVAMAIL API, *http://java.sun.com/j2ee/ja/javamail/index.html*

[12] Cocoon, *http://xml.apache.org/cocoon/index.html*

[13] Gregory R. Andrews, *Concurrent Programming Principles and Practice*, The Benjamin/Cummings Publishing Company, Inc. 1991

[14] Niklaus Wirth,“アルゴリズム+データ構造=プログラム,” 日本コンピュータ協会 (1979)