N_005

プログラミング理解の契機の抽出に向けた学習者コンテキストのモデル化

Modeling of Learner's Contexts to Find Out Clues of Understanding in C Programming

田口 浩[†] 奥井 善也[†] 島田 幸廣[‡] 島川 博光[§]
Hiroshi Taguchi Yoshiya Okui Yukihiro Shimada Hiromitsu Shimakawa

1. はじめに

社会の急速な情報化を支えるため，実用的なプログラミング能力をもつ情報処理技術者を効率的に育成できるプログラミング教育の実施が求められている．

個人の理解状況に合わせたC言語プログラミング教育を実現するために我々は，学習者ごとに最適な演習課題を選出して出題する手法を提案した [1]．当手法を大学でのC言語プログラミング演習科目に適用した結果，個々の学習者の理解を向上させることができた．しかし，当手法では，学習意欲が低下しつつある学習者には学習の継続を，学習意欲が高い学習者にはプログラミング能力の向上を優先して出題を行うため，学習者間の理解の差を埋めることができないという問題が残った．この問題点を解決するためには，先に理解できた学習者のノウハウを理解に苦しんでいる学習者に展開するための手法が必要である．そこで本論文では，プログラミング学習中に発生する種々のデータを取得して学習者ひとりひとりの状態を捉えたうえで，学習者の理解を促進させるきっかけとなった事象を抽出して共用する手法を提案する．

個々のプログラミング学習者の学習履歴を把握し，効果的な教育の実現を目指す既存の研究 [2, 3] では，学習者が作成したソースコードとそのコンパイルおよび実行結果から学習履歴の把握を試みている．しかし，学習した内容を理解できたかどうかを判断するには，そのような情報だけでは不十分である．また，ソースコードの正確性を診断する手法 [4] や，理解に行き詰まっている学習者を自動的に検知する手法 [5] も提案されているが，ソースコード中の誤りや理解の行き詰まりを検知した後の処置を支援する形式的手法やツールは提供されていない．本手法では，講義中の理解状況や教員によるソースコード評価結果といった情報も取得し，それらを統合して扱うので，学習者の理解過程を詳細に把握できる．さらに，ある学習者の理解のきっかけを他の学習者が共用できるようにすることによって理解を支援する．

以下，2. で学習者コンテキストと理解の契機の概念について述べる．3. では学習者コンテキスト・モデルの定義と，それを利用した教育支援手法を説明し，4. で提案手法に基づく学習者コンテキスト獲得システムの実現法について述べる．そして，5. で実際の教育現場で本手法を適用した結果を示したうえで，最後にまとめを行う．

2. 学習者コンテキストと理解の契機

2.1 学習者の状態を表すコンテキスト

世の中の状況を捉えるために利用し得るあらゆる情報を取得・解釈したうえで，ユーザにとって有意義な情報や必要な情報を判断し，ユーザの現在のコンテキストに適合したかたちでそれらの情報を提供することを目指す技術として，コンテキスト・アウェアネス [6] がある．ここでのコンテキストとは，ユーザが置かれた状態を表す情報を指し，ユーザやユーザの周囲に存在する物体の位置，ユーザがとった行動など，さまざまな情報が含まれる．

コンテキスト・アウェアネスの考え方に基づけば，学習に関連する事象を記録し，適切に解釈することにより，学習者の状態を表せられることになる．本論文では学習者の状態を捉えるために必要な情報の集合を，一般的なコンテキストとは区別して学習者コンテキストとよぶ．

2.2 プログラミング学習における理解の契機

あるプログラミング技法を新たに学び，それを理解して実現できるようになるまでの期間は学習者ごとに異なる．そして，自身で一度実現できた技法については，その後は容易に実現できるようになることが多い．これは，学習者の理解を促進させるきっかけとなる事象が学習中に存在し，その事象に遭遇する時点や事象そのものが学習者ごとに異なるためであると考えられる．本論文ではこのような事象を理解の契機とよぶ．

プログラミング学習における理解の契機についての考察を行うため，実際に大学で開講されているC言語プログラミング演習科目での受講者の理解状況とその履歴を調査した．調査は，2004年9月から2005年1月にかけて立命館大学 情報理工学部 情報システム学科で開講された1回生配当科目「プログラミング演習2」の受講者146名を対象に実施した．この科目では，C言語の基本的な学習内容を一通り学習した学生を対象に，データ構造やアルゴリズムといった問題解決手法の習得を目標として全14週の演習を行う．当科目において受講者が提出したソースコードと，それらに対する教員の評価結果より，多くの演習課題に含まれる学習内容である「メモリ領域の動的な確保」と「確保したメモリ領域の解放」について，各受講者の理解状況の推移を調べた．

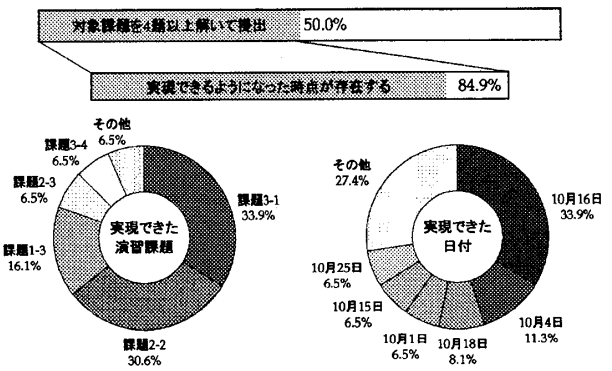

図 1: 学習履歴調査結果

[†]立命館大学大学院 理工学研究科
[‡]株式会社 ゴビ
[§]立命館大学 情報理工学部

「メモリ領域の動的な確保」についての調査結果を図1に示す．この結果より，多くの受講者がある時点を境に当該内容を実現できるようになっていることがわかる．これは，学習中に当該内容に対応する理解の契機が存在したことを示しているといえる．次に，理解できるようになった演習課題とその提出日の分布割合に注目すると，いずれも上位3項目が特に高い値を示していることがわかる．これより，多数の受講者に共通する理解の契機が複数個存在するといえる．「確保したメモリ領域の解放」についても同様の調査結果が得られた．したがって，各学習者の学習過程を把握して共通する理解の契機を見つけられれば，理解に苦しむ学習者に対する効果的な学習支援が実現可能であると推測される．

3. 学習者コンテキストを用いた教育支援

3.1 学習者コンテキスト・モデル

プログラミング学習における理解の契機を見つけるためには，個々の学習者の学習および理解の過程を詳細に把握しなければならない．そこで，学習中に発生するあらゆるデータを取得し，適切に解釈することを考える．

大学などでのプログラミング教育は，プログラミング技法の知識を教える講義形式の科目と，そこで学んだ内容を実習する演習形式の科目で構成される場合が多い．これらの科目では，授業形式が異なるうえにクラスの規模も異なるため，それぞれの科目で取得できるデータの種類および粒度は大きく異なる．本研究では，学習者の状態を把握するためにまず，図2に示すように，講義科目と演習科目においてそれぞれに適した方法で，学習中に生じる以下の4種類のデータを収集する．

学習内容　講義形式の科目では，Microsoft Office PowerPoint（以後，PowerPointと略す）のようなプレゼンテーション・ツールを用いて講義が進められることが多い．そこで，講義科目では，教員がプレゼンテーション・ツール上でスライドを切り替えた時刻を取得することによって，各時点での講義内容を把握する．一方，演習科目では，ソースコードを収集するさいに，どの演習課題に取り組んでいるかという情報も共に取得する．

講義科目での理解状況　講義科目において，教員の説明内容に対する学習者の主観的な理解状況を取得する．学習者が何らかの感情を持った時点でその情報を即座に収集するために，選択式で回答してもらう．選択肢として，理解できたので次の説明に進んでも良いことを示す「ガッテン」，興味を抱いたのでもっと詳しく説明してほしいことを示す「へぇー」，今の説明では理解できないので別の方法での説明を聞きたいことを示す「わかんねー」，そして，聞き逃したなど，理解のうえでもう一度同じ説明をしてほしいことを示す「ちょっと待った」の4つを用意しておく．学習者は教員の説明内容に対して抱いた感情に最も近いものを，これらの選択肢の中から選択することによって理解状況を示す．

ソースコード　演習科目で出題された演習課題に対して学習者が作成したソースコードを収集する．学習

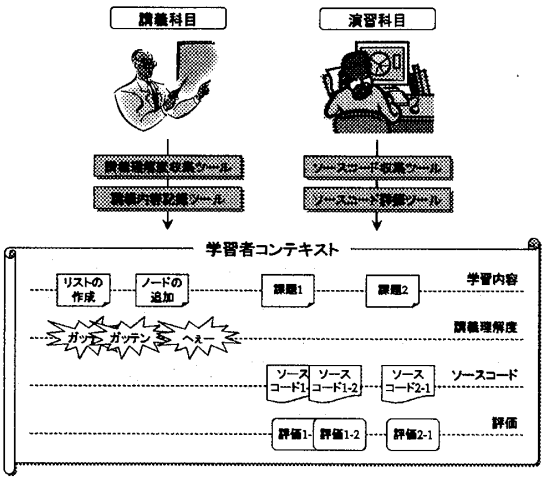

図2: 学習者コンテキスト・モデル

した内容を実現できるようになったかどうかを正確に判断するために，完成したソースコードだけでなく，作成途中のソースコードも一定時間おき，もしくは前回収集時からのコード量の差分が一定値を超えたときに収集する．

ソースコードに対する教員の評価　演習科目において収集されたソースコードに対する教員の評価も取得し，これを学習者の学習成果とする．演習課題ごとに「ファイルを読み込み用にオープンする」といった評価観点をあらかじめ10〜20個ずつ設定しておく．そして，教員はそれぞれの評価観点について，「理解できている」，「理解できていない」，「どちらともいえない」のいずれかを選択することにより，受講者が作成したソースコードの評価を行う．

そして，これらを同一の時間軸上にそれぞれ時系列として並べたものを学習者コンテキストと定義する．したがって，学習者コンテキストはそれら4つの要素の履歴で構成される．これにより，学習中のすべての時点における学習者の状態を十分に捉えることが可能となる．

3.2 理解の契機の抽出とその共用

2.2で述べた学習履歴調査の結果を見ると，ある学習内容を実現することができた複数の学習者に共通する理解の契機を見つけ出せば，その内容をまだ実現できていない他の学習者にとってもそれらが理解の契機となる可能性が高いことがわかる．複数の学習者に共通する理解の契機を見つけ出すためには，個々の学習者の学習過程から理解の契機と思われる事象の抽出を行う必要がある．

獲得された学習者コンテキストから理解の契機と思われる事象を抽出するには，まず，ソースコードに対する教員の評価の履歴に着目し，当該内容に対する評価が良くなった，すなわち，当該内容を実現できるようになった時点を特定する．具体的には，学習者が解いたすべての演習課題の評価観点から，当該内容に関連する評価観点のみを抜き出し，それらに対する評価を順にたどる．そして，評価が「理解できていない」から「理解できて

いる」に変わり，かつ，それ以降の評価がすべて「理解できている」となっている時点を見つける．理解の契機は当該内容を実現できるようになった時点の直前に存在すると考えられるので，次に，実現できるようになった時点の直前回の講義科目での理解状況に着目し，学習者が「ガッテン」や「へぇー」といった肯定的な反応を示している時刻をすべて求める．そして，講義内容の履歴から該当する時刻に説明されていた内容を抽出し，それらの事象を理解の契機の候補とする．

すべての学習者の学習者コンテキストから理解の契機の候補を抽出した後に，それらの事象がそれぞれ，どれだけ多くの学習者の理解の契機の候補となっているかを求める．そして，それらの値に対する閾値を設定しておき，閾値を超える数の学習者に共通する理解の契機の候補を，当該内容の理解の契機とする．

理解に苦しんでいる学習者の学習者コンテキストに，理解できた学習者の学習者コンテキストより導き出された理解の契機が存在するかどうかを調べれば，理解できない原因を突き止めることができる．導き出された理解の契機のいずれかが存在しなければ，その事象と学習者を意図的に遭遇させればよい．具体的には，講義科目での理解状況の履歴から，その説明が行われていた時間の学習者の反応を調べる．理解できていない学習者が「わからない」などの否定的な反応を示している場合や特に反応のない場合は，理解の契機となった講義内容をもう一度詳しく説明するといった指導を行う．このように，個々の学習者の学習者コンテキストを獲得して理解の契機を抽出し，共用することによって，学習者ひとりひとりの状態に即した学習支援が可能となる．

4. 学習者コンテキスト獲得システムの実現

4.1 システム構成

本論文で提案する手法に基づいて学習者コンテキストを獲得するシステムを構築した．

本システムの構成図を図3に示す．本システムは4つのデータ収集ツールで構成され，各ツールで収集されたデータはサーバ内の共通のデータベースで一元管理される．サーバサイドの動作環境として，OSはMicrosoft Windows XP Professional，Webサーバには Apache Tomcat 4.1を，DBMSには Oracle Database 9iを採用した．

4.2 データ収集ツール

本節では，各データ収集ツールの機能と実装例を示す．

講義理解度収集ツール　講義科目において，学習者が教員の説明内容に対する理解状況を登録することができる．本ツールは，Webブラウザをインタフェースとする Webアプリケーションであり，Javaサーブレットを用いて実装している．学習者の学習活動の邪魔にならないように PDA上から利用可能である．図4に PDA上に表示されたボタン画面を示す．学習者はこの画面に表示される4種類のボタンの中から，自身の理解状況に最も近い内容を表すボタンを押すことによって，理解状況を随時登録できる．

講義内容記録ツール　PowerPointを用いて行われる講義科目を対象として，PowerPoint上で教員がスライドを切り替えた時刻を記録する．次のスライドへ進めた場合だけでなく，前のスライドに戻った場合にも対応できる．本ツールは Microsoft Visual Basic 6.0を用いて実装している．

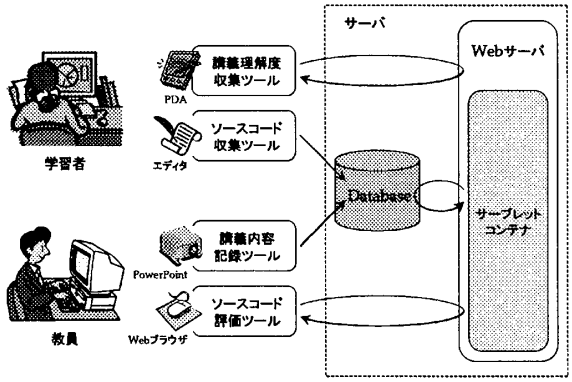

図 3: 学習者コンテキスト獲得システムの構成

図 4: 講義理解度収集ツール

ソースコード収集ツール　Linux上のエディタ Emacsや Windows上のエディタ Meadowを用いて演習課題に取り組んでいるさいに，作成しているソースコードを定期的に収集する．これらのエディタでは，編集中のファイルを定期的に自動保存する機能が実装されており，標準では300打鍵ごと，あるいは30秒間操作をしなかった場合に自動保存される．本ツールでは，自動保存されたファイルを取得し，取り組んでいる演習課題や現在時刻などの情報とともにサーバ内のデータベースに格納する．本ツールは，学習者が使用するコンピュータ上で実行される Javaアプリケーションとして実装している．

ソースコード評価ツール　教員が収集されたソースコードに対する評価を行うためのツールであり，Javaサーブレットを用いた Webアプリケーションとして実装している．評価のさいには，ソースコードとその演習課題に設定されている評価観点が表示されるので，教員はソースコードを参照しながら各評価観点に対する評価を登録する．

表 1: 理解の契機と思われる講義スライドの抽出結果

週	3	4	5	6	7	8	9	10	11	12
演習評価良好者	20名	14名	8名	7名	8名	5名	9名	8名	5名	5名
(ツール利用者)	(25名)	(18名)	(13名)	(17名)	(11名)	(11名)	(11名)	(12名)	(13名)	(15名)
契機と思われるスライド	3枚	3枚	2枚	3枚	0枚	1枚	1枚	2枚	5枚	1枚
(全スライド)	(49枚)	(21枚)	(46枚)	(25枚)	(28枚)	(37枚)	(34枚)	(26枚)	(26枚)	(34枚)

5. 評価

5.1 実講義への適用

2005年9月から2006年1月にかけて立命館大学 情報理工学部 情報システム学科で開講された1回生配当科目「データ構造とアルゴリズム」および「プログラミング演習2」において，学習者コンテキスト獲得システムの一部を実際に導入し，本手法の有効性を検証した．

「データ構造とアルゴリズム」は，C言語の基礎的な学習内容を一通り学習した学生を対象に，基本的なデータ構造とアルゴリズムを講義形式にて説明する科目であり，185名が受講した．「プログラミング演習2」では，「データ構造とアルゴリズム」で学んだ手法を体得することを目標に，演習形式で実際の課題のプログラミングを行う．当科目は171名が4クラスに分かれて受講した．

学習者コンテキストを獲得するために，「データ構造とアルゴリズム」では講義理解度収集ツールと講義内容記録ツールを導入した．受講者の中から有志を募り，各週15名程度に講義理解度収集ツールを配布し，教員がPowerPointを用いて行う講義に対する理解状況を随時入力してもらった．「プログラミング演習2」においてはソースコード評価ツールのみを導入した．ソースコード収集ツールについては導入を見送り，受講者に各演習課題の完成したソースコードのみを提出してもらった．

5.2 適用結果

表1は，講義理解度収集ツールを用いた受講者のうち，その週の演習課題の出来具合が良好であった受講者の人数と，それらの半数以上が「ガッテン」または「へぇー」という反応を示した講義スライドの枚数を示している．演習科目で出題された各演習課題に設定されている評価観点の中で，その週の講義科目において新しく学んだ内容に関連する評価観点の評価がすべて「理解できている」である場合を出来具合が良好であるとみなした．なお，両科目はそれぞれ14回ずつ授業が行われたが，データ収集は第3週から第12週までの10回の授業で実施した．

表からわかるように，第7週を除くすべての週において，演習の出来具合が良好である受講者の半数以上が共通して肯定的な反応を示した講義スライドが1〜5枚ずつ存在している．第7週では，3枚のスライドにおいて8名中3名が共通して「ガッテン」や「へぇー」という反応を示した．また，評価が良好でなかった受講者の中で，それらのスライドのすべてに対して肯定的な反応を示した受講者は，すべての週において皆無かごく少数であった．これより，それらのスライドを用いて説明された内容を理解できれば，演習の出来具合も良くなるということがわかる．よって，それらの説明内容が理解の契機となっている可能性が高い．

このように，学習者コンテキスト獲得システムの一部を実際の講義科目および演習科目に導入した結果，本手法を用いて理解の契機と思われる事象を学習内容ごとに少数に絞って抽出することができた．したがって，本手法は先に理解できた学習者のノウハウの共有を実現するうえで有効であるといえる．

6. おわりに

本論文では，プログラミング学習における学習者の状態を表す学習者コンテキストのモデルと，それを用いた教育支援手法を提案した．本手法を大学で実際に開講されているC言語プログラミングの講義科目および演習科目に適用した結果，学習者コンテキストを獲得することによって，理解の契機と思われる事象を学習内容ごとに数個ずつ抽出することができた．

今後は，抽出された理解の契機の共用法についての研究を進める予定である．

参考文献

[1] Taguchi, H. and Shimakawa, H.: Exercise Sequence Adaptation in Programming Education, *Proc. the International Symposium on Communication and Information Technologies 2004*, pp. 774–778 (2004).

[2] 長 慎也, 筧 捷彦:proGrep - プログラミング学習履歴検索システム，情報処理学会研究報告 (コンピュータと教育)， No. 2005-CE-78, pp. 29–36 (2005).

[3] 知見邦彦，櫨山淳雄，宮寺庸造：失敗知識を利用したプログラミング学習環境の構築，電子情報通信学会論文誌 (D-I)， Vol. J88-D-I, No. 1, pp. 66–75 (2005).

[4] Anderson, P., Reps, T. and Teitelbaum, T.: Design and Implementation of a Fine-Grained Software Inspection Tool, *IEEE Trans. on Software Engineering*, Vol. 29, No. 8, pp. 721–733 (2003).

[5] 中村喜宏，赤松則男，桑原恒夫，玉城幹介：操作時間間隔の変動に着目したCAI学習の行き詰まり検知方法，電子情報通信学会論文誌 (D-I)， Vol. J85-D-I, No. 1, pp. 79–90 (2002).

[6] 上岡英史：コンテキストアウェアネスを用いたアプリケーションの研究動向，情報処理， Vol. 44, No. 3, pp. 265–269 (2003).