理解度モデルを用いたアルゴリズム学習支援方法の提案

2010SE068 伊藤 光　2010SE091 加藤 雄規　2010SE106 近藤 啓太
指導教員：蜂巣吉成

1　はじめに

プログラムを作成するにはプログラミング言語の文法を知っているだけでは不十分であり，解決しようとしている問題のアルゴリズムを考えなければならない．アルゴリズムを理解することはプログラミング学習において重要な役割を占めている [1].

現状のアルゴリズム学習では，学習用の教科書を用いて，その中の説明や，図表，ソースプログラムなどを学習者が読んで理解することが多い．授業などではプレゼンテーションソフトウェアなどを利用して，アルゴリズムをアニメーションで示し，学習者の理解を促すような工夫も行われている．しかし，図表やアニメーションで学習者がアルゴリズムを理解したつもりでも，実際にプログラムとして記述しようとした場合に，記述できない場合も少なくなく，アルゴリズムを十分に理解しているとは言えない．また，学習においては 1 つの教科書を用いることが多く，一般に教科書では 1 つの実装例 (ソースプログラム) が載っている．アルゴリズムを実現するソースプログラムは複数ありえるので，学習した教科書以外の実装例を読んでもそのアルゴリズムがわからなければ，理解したとは言えない．

本研究では，教科書の解説を読むだけの学習や，1 つのソースプログラム例のみの学習で理解したつもりになっている状態を改善・補完するために，自習用として，空欄補充問題を用いて学習者の理解度を把握しながら，効率的にアルゴリズムの学習を支援する方法を提案する．

アプローチとして，本研究におけるアルゴリズム理解を定義し，その定義を満たす学習方法を調査した結果，空欄補充問題を用いた支援方法が適当であると考えた．本研究ではアルゴリズムの中でも，最も基本的な問題といわれているソートアルゴリズムを対象にする．空欄の箇所を決めるための基準として，複数のソースプログラムを調査し，アルゴリズムやソースプログラムの構造を表した構造モデルを作成し，これを用いて理解度を計算するための理解度モデルを作成した．作成した構造モデルに沿って出題された空欄補充問題を解答し，理解度モデルに従って各空欄の解答の正誤から理解度を算出して，学習者に自分の理解度を把握させる学習支援方法について考察を行う．

2　関連研究

アルゴリズムの視覚表現による理解支援として，構造化チャート [2] などによる学習支援方法が挙げられるが，これらの研究ではデータの流れや変数の値の変化を理解することを目的としており，実際にソースプログラムとして記述できるための支援は行われていない．

アルゴリズム理解を目的としたソースプログラムの空欄補充問題を出題する研究として，PDG を用いてソースプログラムに空欄を設定する研究が挙げられる [3]．この研究では PDG で制御構造上，より深い場所にあり，データ依存関係の辺の数が多く，データ依存の変数が多い頂点に空欄を設ける．しかし，特定の箇所のみが空欄となり，その他の箇所に対しての理解は考慮されない．

3　本研究におけるアルゴリズム理解

本節ではアルゴリズムを実現する複数のソースプログラムに対する，アルゴリズム理解の定義と重要性を記述する．

3.1　アルゴリズムとソースプログラム

次のバブルソートでの実装例 (ソースプログラム 1，ソースプログラム 2) のように処理の詳細な手続きの異なるプログラムが存在する．[4] ではソースプログラム 1 が，[5] ではソースプログラム 2 が掲載されている．

ソースプログラム内の下線は重要箇所を表す．

ソースプログラム 1 [4]

```
1void bubbleSort()                          条件なし逐次処理
2{                                          不等号
3   int i,j,sorted;                         比較基準値
4   j = n;                                  交換
5   do {
6     sorted=1; j=j-1;
7     for (i=1;i<=j;i++)
8       if (a[i]>a[i+1]){
9           swap(&a[i],&a[i+1]);
10          sorted=0;
11      }
12    }while (!sorted);
13}
```

ソースプログラム 2 [5]

```
1void bubble sort(int data[], int n)
2{
3  int i, j;
4  for(i = 0; i<n - 1; i++){
5     for(j=n - 1; j>i; j--){
6       if(data[j] < data[j - 1]){
7       swap(&data[j],&data[j - 1]);
8       }
9     }
10  }
11}
```

3.2　アルゴリズム理解

ソフトウェア開発における保守工程では，既存のプログラムを再利用して新しい機能を追加することや欠陥を修正することがある．上述したバブルソートのプログラムをカスタマイズしたり，バグが含まれていれば，それを修正す

ることも想定される．そのためにはこれらのプログラムを読んで理解する必要があり，デバッグでは誤り箇所を発見して，正しい記述に修正する必要がある．

以上のことから，本研究におけるアルゴリズム理解とは，アルゴリズムを実現する複数のソースプログラムに対して，それらを読んで理解できること，および，その計算手続きの重要箇所を記述できることとする．

4　アルゴリズム学習手順

本研究は，教科書の解説を読むだけの学習や，1 つのソースプログラム例のみの学習で理解したつもりになっている状態を改善・補完するための，自習用としての学習方法を提案するが，本節では授業との関係も考える．自習や授業でアルゴリズムを学習していく手順を学習手順と名付ける．自習では独自で教科書を読んで学習することが多いが，授業も教科書に基づいて行われることが多く，学習手順も同じようになる．授業でのアルゴリズムの学習手順を調査し，学習手順と本研究の提案方法の関係を明確にすることによって，学習者が満たしている前提条件を割り出し，学習支援に必要な要件を考察する．

4.1　学習手順の調査

複数の教科書を対象に，どのような順番で学習内容が構成されているかの調査を行い，その結果から，大学の講義等での具体的なアルゴリズムの学習手順の全体像を考察したところ，次のようになった．

1. 教科書等の自然言語による解説を読んで理解する．
2. 図表や数列の具体例による視覚的表現を用いた解説を見て理解する．
3. 実装例を読んで理解する．
4. 自分でプログラムを記述して理解する．

これらの手順を必要に応じて繰り返し用いることで最終的な理解を目指していると考えられる．また，実際の学習手順は，例えば 1 と 2 を逆の順で学習したり，3 や 4 を行わないなど，学習者や教員よって手順に個人差が生じることが想定される．

4.2　学習手順と提案方法の関係

授業等で上記の学習手順に基づき学習を行っても，実際には次の問題が生じる．

- 1 と 2 の学習では処理の詳細な手続きを学べない．
- プログラムにおける実現方法を学ぶための 3 の実装例が教科書には 1 つしか載っていないことが多い．
- 4 は必ずしも行われるとは限らない，行ったとしても教科書に基づいて 1 つの実装例のみの学習が多い．

これらの問題を解決して効率的に学習を進めるために，本研究では考察した手順の 3，4 の部分を支援する．その際，処理の詳細な手続きを学ぶために複数のソースプログラムによる問題を用いる．本研究での学習支援方法は『授業や教科書でのアルゴリズム学習を一通り終えた後で，理解できないところを補うためのもの』とする．本研究での学習者が満たしている前提条件を次のとおりとする．

- 文法学習についてはアルゴリズム学習の前段階で済ませている．
- 教科書の自然言語による解説と図説で，アルゴリズムの処理の概要は理解している．
- 少なくとも教科書の実装例に関しては処理の詳細な手続きを理解している．

5　アルゴリズム学習支援方法

本研究では 3.2 節で定義したようなアルゴリズム理解を支援するための方法を提案する．学習者は，4.2 節で定義した前提条件をすべて満たしているものとする．

本節では，自習用として問題を出題し，アルゴリズムの学習を支援する方法を考察する．

5.1　出題方法

3.2 節で定義した通り，アルゴリズムのソースプログラムに対して，計算手続きの重要箇所を記述できることが重要であり，それを実現するための出題方法を考察する．

ソースプログラムを用いる学習方法として，全文記述問題，誤り訂正問題，空欄補充問題，選択問題が挙げられる．

全文記述による学習は，本研究におけるアルゴリズム理解の「ソースプログラムを読んで理解すること」を実現できず，計算手続きの重要箇所に問題を含ませることもできない．計算手続きの重要箇所に問題を含ませることが可能な誤り訂正問題，空欄補充問題，選択問題は本研究において有効であると言える．しかし，誤り訂正問題は重要箇所だけを的確に学習することが困難な上，難易度も空欄補充問題と選択問題に比べて高い．選択問題は重要箇所が明確に見えるが，難易度が低いので，本研究では空欄補充を用いた学習支援を想定し考察をする．

5.2　対象とするアルゴリズム

本研究では，アルゴリズムの中でも，最も基本的な問題といわれているソートアルゴリズムを対象とする．以降はバブルソートを例に挙げる．

5.3　アルゴリズムの構造モデル

一般に学習をする際，階層構造の上から下へ順に理解を進めることが望ましいと考えられる [6]．本研究では，アルゴリズム毎にその計算手続きの重要箇所を選定し，それらを階層的に木構造で表現する．この木構造をアルゴリズムの構造モデルと名付ける．

本節では，空欄補充問題を作成するために，ソートアルゴリズムに関する自然言語の解説と実際のソースプログラムを解析し，処理の流れの要所となる部分を抽象化した構造モデルを作成する．

5.3.1 ソートアルゴリズムの解析

5.2 節で挙げたソートアルゴリズムについて，複数の参考書を用いて，処理の過程を自然言語やフローチャートなどから解析した結果，次の 3 種類の処理に分けられたので，それぞれ定義付けを行った．

移動 … 配列の要素の移動を行う処理．
範囲 … ソートを行う範囲に関する処理．
大小比較 … 要素同士の大小を比較する処理．

これら 3 つの要素は構造モデルの頂点となる．

5.3.2 ソースプログラムからみたアルゴリズム

次に，実際のバブルソートのソースプログラムから計算手続きの重要箇所に着目し，複数の書き方に対して重要箇所の抽象化を行った．抽象化した重要箇所にそれぞれ名前を付け，それぞれ定義付けを行った．

比較基準値…最大値など大小比較を行う際の基準値．
条件なし逐次処理…先頭から配列の末尾（ソート済み部分を省いてもよい）までをすべて通る処理．選択ソートで交換する先頭の値とバブルソートの隣接する要素もこの項目に含める．
不等号…配列の要素を大小比較する処理．
交換…配列の要素 2 つを交換する処理．

これらの項目を，複数の種類の下線を用いて 3 節のソースプログラム 1，ソースプログラム 2 の中で表現した．この下線部分の一部が問題の空欄となる．

5.3.3 構造モデルの作成

5.3.1 節と 5.3.2 節の解析結果を用いて，各アルゴリズムについて重要箇所をまとめると図のように 1 つの木として表現することができる．空欄補充問題では，この構造モデルの葉の概念に相当するソースプログラムの箇所が空欄となる．本研究では，問題の空欄の位置や個数は難易度や複数回答の可能性などから動的に変化すると考え，選出は人為的に行うものとする．

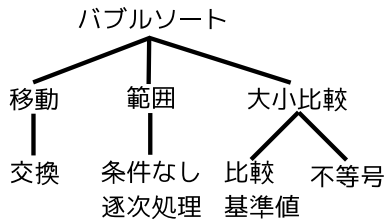

図 1 バブルソートの構造モデル

5.4 理解度モデル

5.3 節で述べたように，一般に学習で理解を進めるプロセスは階層構造の上から下へと順に進んでいく．階層構造の下の部分，即ち本研究の構造モデルでの，部分木を構成する全ての葉の理解を満たすことにより，部分木の根の部分が理解できる．学習者が理解していない箇所や適切な難易度に関する問題を出題するためには，構造モデルにおける頂点をどの程度理解しているかを把握する必要がある．この理解の度合いを数値化して表現したものを，本研究におけるアルゴリズムの理解度とする．また，学習者は自分の理解度を把握することによって，正しい自己評価をし，何をどう学習していけばよいかの指針を自分で作り上げることができる．

本研究では，処理の詳細な手続きの違いによって複数存在するアルゴリズムのソースプログラムの，それぞれにつき空欄補充問題の作成を行い，正誤を計算式に従って点数化することで理解度を求める．ここでは，各空欄の点数を 2 点とする．ただし，同じ意図の空欄が複数存在し，空欄の関連性が強いと考えられる場合は，その個数で割った点数とする．正解すると，この点数が構造モデルの各葉の部分に加算され，各葉で，(正解の合計点)/(満点) によって理解度が算出される．また，葉以外の頂点では，(部分木の葉の正解の合計点)/(部分木の葉の満点の合計点) によって理解度が算出される．複数のプログラムを通しての理解度は，計算する箇所について，(すべてのプログラムでの該当箇所の正解の合計点)/(すべてのプログラムでの該当箇所の満点の合計点) によって算出される．構造モデルを用いて理解度を計算する仕組みを理解度モデルと名付ける．

学習者は，構造モデルに沿って出題された空欄補充問題を解答し，理解度モデルに従って解答の正誤から理解度を算出して，自分の理解度を把握することで，学習の指標を立てる．また，この理解度を反映させた問題を出題することで，効率的に学習を進めることができる．

6 実験・考察

5 節で提案した学習支援方法によって，実際に効率的にアルゴリズムの学習を支援することが可能であることを，具体的な問題を用いた実験によって検証する．

実験によって検証する点は以下の項目である．

1. 1 つの実装例を理解しただけで他の実装例に関する問題を解けるか
2. 理解度モデルを用いることで，学習者は自分の理解度を把握し，実際に学習に活かすことができるか
3. 本研究で提案する学習方法は，アルゴリズム学習における既存研究と比べて，アルゴリズム理解を支援することについて有効であるか

6.1 実験方法

今回の検証では，バブルソートに関する空欄補充問題を，学生 10 名を対象に出題する．実験は紙面上で行う．検証項目の 3 を調べるために，文献 [3] との比較を行う．被験者は理解度モデル側と PDG 側に 5 人ずつ無作為に分かれて実験を受けてもらう．

実験の手順は，まず 4.2 節で挙げた前提条件を満たすために，文献 [4] を用いて，スライドでバブルソートの説明を行い，説明後の確認テストと，文献 [4] のソースプログ

ラム (3.1 節のソースプログラム 1) に構造モデルを用いて作成した空欄補充問題 (問 1) を出題する．その後 3.1 節のソースプログラム 2 のように，異なるソースプログラムを使った問題 (問 2，3) を 2 問出題する．問 2，3 は理解度モデル側と PDG 側で，それぞれの問題作成方法によって別々の問題を用意し，問 1 を理解できている状態で問 2，3 を解けるかどうかで，検証項目の 1 を確認する．最後に，問 1，2，3 とも異なるソースプログラムを用いた問 4 を出題する．問 2，3 を解答後，それぞれの理解度を計算し，自分の理解度を把握した上で学習を行ったとき，問 4 での各処理の要所の正答率は上昇するかどうかで検証項目の 2 を調べる．問 4 は理解度モデルと PDG の両方の出題方法を踏まえて作成した問題を出題して正答率を比較することで項目の 3 を調べる．同時に，アンケートによって学習者が体感的に理解度モデルが役に立ったかどうかを調べる．

6.2 実験結果

実験結果を表 1，2 のように示す．問 4 については，処理の重要箇所に関する理解自体はできていると判断したケアレスミスは，正解として集計した．表 1 では理解度モデル側の被験者 F から J の各問の理解度を集計し，表 2 では全被験者の処理の重要箇所ごとの正答率を集計した．

表 1 各問の理解度

	F	G	H	I	J
問 2	1.00 (8/8)	0.50 (4/8)	1.00 (8/8)	1.00 (8/8)	1.00 (8/8)
問 3	0.70 (7/10)	0.40 (4/10)	0.80 (8/10)	1.00 (10/10)	0.80 (8/10)
問 4	1.00 (10/10)	0.90 (9/10)	1.00 (10/10)	1.00 (10/10)	0.90 (9/10)

表 2 各問の正答率

	条件なし 逐次処理	不等号	交換	比較基準値
問 2(理解度モデル)	13/15	4/5	10/10	
問 3(理解度モデル)	7/10	5/5	10/10	3/10
問 4(理解度モデル)	9/10	5/5	10/10	10/10
問 4(PDG)	8/10	5/5	10/10	10/10

理解度モデル側の問 2 から問 4，問 3 から問 4 では全体的に正答率の上昇が見られた．特に問 3 から問 4 については，比較基準値に該当する正答率が大きく上昇している．

PDG 側と理解度モデル側の問 4 の正答率を比較した場合，条件なし逐次処理で理解度モデル側の方が若干上回ったが，全体として大きな差は出なかった．

アンケートについては，理解度モデル側への「作成した理解度モデルは理解度を客観的に把握するために役に立ちましたか?」という質問に対し，4 人が「役に立った」，1 人が「まぁまぁ役に立った」と回答した．しかし，理解度モデル側への「理解度モデルは自分の苦手な部分を把握するための学習の指標として役に立ちましたか?」という質問に対し，「役に立った」が 3 人，「まぁまぁ役に立った」と「まぁまぁ立たなかった」が 1 人ずつという結果になった．この内，「まぁまぁ立たなかった」と回答したのは，問 1 から 4 のすべての問で全問正解している被験者だった．

6.3 考察

検証項目の 1 に関しては，当初の仮説の通り，1 つの実装例で理解していても他の実装例は書けないことが起こりえることがわかった．検証項目の 2 に関しては，全ての被験者が自分の理解度を把握できたと言えるが，理解度を把握できても，学習に活かせない場合があったが，これは全ての理解度が満点になっていた被験者の場合にのみ起こり，その他の被験者は，被験者 G の問 3 から 4 での理解度の伸びのように，学習の指標を立てることで実際の学習に活かすことができたと言える．検証項目の 3 に関しては，明確な差を示すことができなかったので，問題の難易度の調整，出題方法など，今後の考察や検証が必要である．

本研究では，空欄補充問題による学習支援方法を前提に提案を行ったが，誤り訂正問題と選択問題については，理解度モデルを用いることが可能と考えられる．

また，今回はソートについての理解度モデルの研究を進めてきたが，探索やその他のアルゴリズムについても同様にこの方法が使えるかも調査が必要である．

7 おわりに

本研究では，現状のアルゴリズム学習で起こり得る，教科書の解説を読むだけの学習や，1 つのソースプログラム例のみの学習で理解したつもりになっている状態を改善・補完するために，自習用として，空欄補充問題を用いて学習者の理解度を把握しながら，効率的にアルゴリズムの学習を支援する方法を提案した．

実験の結果，学習効率について既存研究との明確な差を示すことはできなかったが，提案する学習方法によって理解度を把握し，学習の指標として役立たせることが可能であることが検証できた．今後は適切な難易度の調整や出題順を検証する必要がある．

参考文献

[1] クリスフォード・シェーファー（久野禎子，久野靖訳）:「Java によるデータ構造とアルゴリズム解析入門」. ピアソン・エデュケーション，2000.

[2] 斐品 正照，徳岡 健一，河村 一樹 :“構造化チャートを用いたアルゴリズム学習支援システム”．情処学論，Vol. 45，No. 10，pp. 2454-2467，2004.

[3] 柏原昭博，寺井淳裕，豊田順一 :“いかにプログラム空欄補充問題を作るか?”．信学技報，ET，教育工学，Vol. 99，No. 81，pp. 9-16，1999.

[4] 平田富夫 :「アルゴリズムとデータ構造改訂 C 言語版」. 森北出版株式会社，2010.

[5] 河西朝雄 :「C 言語によるはじめてのアルゴリズム入門改訂 3 版」. 技術評論社，2010.

[6] 浅野 考平，森戸 隆文 :“基本アルゴリズム理解の分析と教育への応用”．情処研報，コンピュータと教育，Vol. 2013-CE-119，No. 6，pp. 1-4，2013.