プログラムスライシングを用いた初級プログラマのための知識獲得支援

2V-03

石澤　崇裕　天野　直樹　野中　誠　東　基衛
早稲田大学大学院　理工学研究科　経営システム工学専門分野

1　はじめに

適応型情報システムの研究プロジェクト DAISY の一環として，初級プログラマを対象とした適応型学習プロセス支援・改善システムの構築を目指した共同研究を実施している．これまでに学習プロセスにおける初級プログラマの傾向を抽出した[1]．本報告では，初級プログラマの作業プロセスにおける停滞発生に関する傾向把握実験を行った結果を述べる．そして，実験結果に基づき初級プログラマの知識獲得を支援する手法を提案する．また，提案手法の有効性を実証するために行った実験の結果について述べる．

2　傾向把握実験の結果

初級プログラマの停滞の内容を分析するために，プログラミング作業中の停滞や学習を定性的に観測する実験を行った（被験者：プログラミングの初等教育を受けた学生 4 名）．この結果，表 1 に示すように初級プログラマの停滞の多くは「モジュールライブラリ知識」の欠如が主な要因であることが明らかになった．

表 1 ライブラリ知識の不足による停滞の割合

被験者	総開発時間（分） X	停滞解決時間（分） Y	ライブラリ知識の不足が要因（分） Z	割合 Z/Y
A	408	152	79	52.0%
B	199	91	72	79.1%
C	409	323	297	92.0%
D	385	115	115	34.8%

これは，初等教育を終えた学生が実践的なプログラムを開発するためにはライブラリの活用が必須であるが，必要な知識の検索が困難となるためだと考えられる．

A Method for Supporting Knowledge Acquisition for Novice Programmers by Program Slicing
ISHIZAWA Takahiro, AMANO Naoki, NONAKA Makoto, and AZUMA Motoei
Graduate School of Science and Eng., Waseda University

3　提案する知識獲得支援手法

上述の問題を，既存のソースコードから抽出した知識をリポジトリに蓄積し，停滞時に適切な知識を初級プログラマに提供する手法によって解決する．

3.1　既存のコードからのライブラリ利用知識抽出

既に開発されたソースコードからスライシング解析[2]によってライブラリ利用知識を抽出する．図 1に *“ファイルから順に数値を読み込み平均値を求める”* というプログラムに適用した結果を示す．

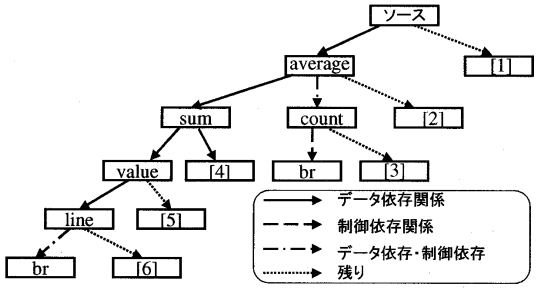

図 1　スライシング解析

この結果，図 2に示すように特にネストが深い”br”, ”line”, ”残り[6]”のスライスはファイル入力を行う際に必要となるライブラリ利用知識を表している．

```
br（ファイルからデータを読み込む）
        file = new File(args[0]);
        FileReader fr = new FileReader(file);
        BufferedReader br = new BufferedReader(fr);

        while(br.ready()){
        }

line（ファイルから文字列を一行毎に読み込む）
        file = new File(args[0]);
        FileReader fr = new FileReader(file);
        BufferedReader br = new BufferedReader(fr);

        while(br.ready()){
                line = br.readLine();
        }

残り[6]（文字列を一行読み込んで代入）
        line = br.readLine();
```

図 2　スライシングの結果得られたライブラリ利用知識

3.2 ライブラリ利用知識の共有

上述の手法で獲得したライブラリ利用知識（スライス）を再利用可能な状態で共有するために，各スライスに図 3に示す属性を付与しリポジトリに蓄積する．

Slice-N	
含むメソッド名	:String[]
含むAPIクラス名	:String[]
ネストの深さ	:int
スライスの中身(コードの断片)	:String

図 3 各スライスの属性

3.3 蓄積された知識の提示

作業プロセス中に停滞が発生している場合，

- ソースコード開発履歴
- ライブラリ利用履歴
- ライブラリドキュメント参照履歴
- コンパイルエラー履歴

などの作業履歴に含まれるモジュール名から，その要因であるライブラリ利用知識の欠如を特定できる．この方法で得られたモジュール名をトリガーとし，リポジトリに含まれるスライスの属性とをマッチングすることにより，停滞時に適切なライブラリ利用知識の提示を行うことが可能となる．

4 実験環境の開発

本提案手法を実現するためにプログラミング作業履歴を獲得可能なエディタを開発した．機能仕様は以下の通りである．

① テンプレートによる構文入力と作業履歴獲得

- クラス入力
- 属性入力（基本データ型・ライブラリ利用）
- メソッド入力
- ブロック文入力

② 学習履歴獲得

- API ビューワーの実現・ドキュメント参照履歴獲得

5 検証実験

提案手法の有効性を検証するために，プログラミングの初等教育を終えた学生9名を対象に実験を行った．

実験計画： 被験者を3名毎3つのグループに分ける（以下グループA・B・Cとする）．プログラムエディタから得られる作業履歴を解析し，モジュールライブラリ利用の知識に関する停滞が生じた場合，グループAには解答のソースコードを提示する．グループBにはスライス木のネストが深いものから順に停滞が解決するまでスライスを提示する（提案する支援手法の適用）．また，グループCには何れの提示も行わない．

手順： 被験者にファイル入力を含む課題プログラムの仕様を与え，制限時間120分でコーディングを行わせる．そして，課題遂行時間・学習効果を測定する．なお，学習効果の測定はコーディング終了後に簡単なテスト（100点満点）を行うことで実施する．

結果： グループA・Bでは全員課題を達成できた．一方，グループCの被験者は何れも制限時間内に課題を完成できなかった．グループA・Bの課題遂行時間，および学習効果の平均は表 2の通りである．

表 2 検証実験の結果

	課題遂行時間（分）	学習効果（点）
グループA	40.0	11.7
グループB	62.3	85.0

考察： グループA・Bを比較すると，課題遂行時間はグループAの方が短いが，一方で学習効果はグループBの方が高く両グループの差が顕著である．これは，グループAは停滞時に解答を全て見ているため，与えられたコードの意味を十分理解せずに解答をほぼ丸写ししていることが原因と考えられる．一方，グループBはスライスによって局所的な解は与えられるが，その他の部分は自力で解決策を模索しなければならない．従って与えられたスライスそのものの意味を理解する必要が生じ，当該するAPI ドキュメントを参照するなどの行動が見られた．このためグループBはグループAと比較して学習効果が高くなったものと考えられる．

6 まとめ

検証実験によって本報告で提案する手法が初級プログラマの知識獲得支援に有効であることが実証された．

参考文献

[1] 阿部正典他：ソフトウェア開発システムにおける人的資源の質の改善について—初級プログラマの作業中における学習と知識のモデル化—，初級プログラマの傾向の一般化と問題点の明確化，情報処理学会第58回全国大会，1999

[2] 平川一郎他：*プログラムスライシングを用いたプログラミング知識の抽出*，信学技法 KBSE95-8，1995