機械翻訳を用いた擬似コード生成による学習者支援

Supporting Learners by Automatically Generating Pseudo Code

札場 寛之*, 小田 悠介*, Neubig, Graham*, 畑 秀明*, Sakti, Sakriani*, 戸田 智基*, 中村 哲*
Hiroyuki Fudaba, Yusuke Oda, Graham Neubig, Hideaki Hata, Sakriani Sakti, Tomoki Toda, Satoshi Nakamura
*奈良先端科学技術大学院大学情報科学研究科
*Graduate School of Information Science, Nara Institute of Science and Technology
Email: fudaba.hiroyuki.ev6@is.naist.jp

あらまし： ソースコードは特定のプログラミング言語で記述されており，初学者は理解に多くの労力を必要とする．一方，読解者の母語で記述された擬似コードはプログラミング言語に関する知識を必要とせず，ソースコードと共に提示することで読解の補助に有効であると考えられる．本研究では機械翻訳の手法により Python コードから日本語の擬似コードを自動生成し，読解者へ提示する手法を提案し，読解支援効果を実験で検証する．
キーワード： プログラミング教育，擬似コード，機械翻訳

1. はじめに

プログラマには他人が書いたプログラムのソースコードを読解する能力が求められる．この際，ソースコードは特定のプログラミング言語で書かれているため，その言語に慣れ親しんでいない読解者はプログラムの内容以前に記述言語の構文や仕様の不理解が読解の妨げとなる．特にプログラミング初学者はこの傾向が顕著であると考えられる．一方，アルゴリズムを教育する場面では，学習者の母語で記述された擬似コードを示すことでこの問題を回避している．一般のソースコードについても，各行の動作を説明する自然言語の文を読解者の必要に応じて示すことができれば，その言語に不慣れであってもより簡単にソースコードの読解を進めることができると考えられる．

上記のようにソースコードの読解を支援する自然言語の文を提示するには，ソースコードに対してあらかじめ対応する説明文を付与しておく必要があるが，この前処理は作業者に大きな負担を強いるため現実的ではない．与えられたソースコードに対応する説明文を自動的に生成することができれば，この問題を回避することが可能となると考えられる．

ソフトウェア工学の分野では，このようなソースコードから自然言語の説明文を自動生成する研究が提案されている[1,2]．本稿では小田らにより提案された，統計的機械翻訳（Statistical Machine Translation: SMT）を用いてソースコードから擬似コードを自動生成する手法[1]に着目する．この手法はソースコードと自然言語の間の対応関係をデータから自動的に学習するため，人手でソースコードから自然言語への変換規則を記述する必要がないという利点がある．このため，様々なプログラミング言語と自然言語の組に対してプログラムの読解支援システムを少ない労力で作成可能になると考えられる．

本稿では自動生成された擬似コードが実際に読解者の補助となるかを実験によって検証する．具体的には，人手で生成された擬似コードと自動生成された擬似コードを被験者に提示し，提示しない場合と比較し被験者のプログラムに対する理解度と読解に要する時間が改善されるかを調査する．

2. SMT による擬似コードの自動生成

小田らの研究[1]では，Python で記述されたソースコードから日本語の擬似コードを自動生成するために Tree-to-String 翻訳と呼ばれる SMT の手法を用いる．これは入力文の構文木を受け取り，木の部分構造を置き換えてゆくことで出力となる自然言語の文を生成する手法である．具体的には，Python ソースコードの抽象構文木を入力，日本語擬似コードを形態素解析した単語列を出力とする「対訳データ」を用意し，これらの間の対応関係を Tree-to-String 翻訳の枠組みで学習する．図 1 に示すのは Python ソースコードの例と，これを小田らの手法で処理して得られる日本語の例である．

def n_divide_a(n, a): n と a を引数とする関数 n_divide_a の定義
for i in range(n): n 未満の非負整数を小さいほうから順に i として
if a < 2: もし a が 2 より小さければ

図 1 小田らの手法による擬似コードの例

3. 実験

3.1 実験目的

本研究の目的は，2 節で述べた手法による擬似コードをソースコードとともにプログラミング初学者に提示した際，プログラムの読解にどの程度貢献するかを被験者実験により検証することである．

3.2 実験手法

Python で記述された算術に関するプログラム 75 個を用意し，これらを被験者ごとにランダムに 3 グループに分割し，それぞれ下記の処理を施す．

1. ソースコードのみ（code）
2. ソースコードの各行に自動生成した擬似コードを付与（code+auto）

3. ソースコードの各行に人手で擬似コードを付与（code+man）

これらのプログラムをランダムな順序で被験者に提示する．被験者は提示されたプログラムが理解しやすかったかどうかを「全く分からない」から「非常によく分かる」までの 6 段階で評価する．その後，被験者は提示されたプログラムがどのような処理を行うかの説明を記述する．ここで記述された説明を用いて第三者による理解度の評価を行い，被験者の主観的な理解度が客観的に妥当であるかどうかを確認する．これらの評価と同時に，被験者が実際にソースコードを眺めていた時間を記録する．これはプログラムの読解が完了する，または読解を諦めるまでの時間と考えることができる．

被験者として Python の経験が 1 年未満の学生 8 名（非経験者），および 1 年以上の学生 6 名（経験者）を対象とした．提示された擬似コードが人手によるものか自動生成されたものかは被験者には伝えていない．

3.3 実験結果と考察

図 2 に，非経験者，経験者，これらを総合したグループについて実験で得られたプログラムの理解度の比較を示す．

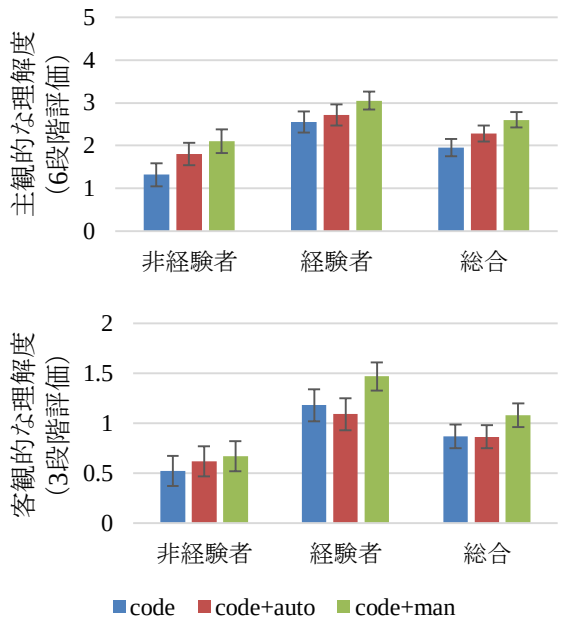

図 2 主観的・客観的な理解度の比較

この結果から，習熟度を問わず，ソースコードのみを提示した場合より擬似コードを同時に提示した場合のほうが主観的な理解度は向上することが分かる．自動生成した擬似コードを提示した場合，経験者の主観的な理解度は向上したが，客観的な理解度は提示しない場合と比べても低い．これは経験者が不自然な擬似コードを提示された際に，逆に混乱したためと考えられる．

図 3 に，各グループについてプログラムを理解するまでにかかった時間を示す．すべての場合において，人手による擬似コードを提示した場合は理解するまでの時間が最も短く，自動生成された擬似コードでは最も長くなった．これは，自動生成された擬似コードが不自然な文になっていた場合，それを理解しようとするために時間がかかるためと考えられる．特に非経験者について，コードのみを提示した場合でも比較的時間が短いのは，コードが理解できず，読解を諦めるのが早かったためと考えられる．

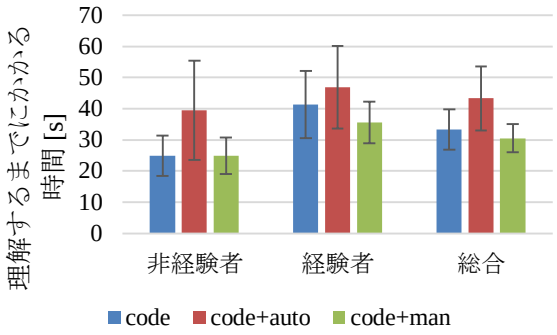

図 3 理解するまでにかかる時間の平均

実験後に集計したアンケートでは,「わかりやすい擬似コードが提示されている場合はソースコードを読まなくても理解できた」「短いプログラムに対しては（ソース）コードだけでも十分に理解可能だが，長いプログラムに関しては（擬似コードが）あるほうが理解しやすかった」という意見が得られた．また，Python 初学者による意見として「擬似コードが書かれている場合，関数の意味を類推することができるため，ドキュメントを調べる手間が省けた」「Python の経験がないため，文法を知らなかったが擬似コードが同時に提示されている場合はそれらの意味をなんとなく知ることができた」といったものが見られた．

4. 終わりに

本研究では，SMT を用いてソースコードから擬似コードを自動生成する手法について，実際に生成された擬似コードがソースコードの読解に役立つかどうかを検証した．Python 初学者による被験者実験の結果，自動生成した擬似コードをソースコードとともに提示したとき，特にそのプログラミング言語の経験が浅い場合プログラムの読解の補助となることが示された. 今後の課題としては, 行単位ではなく，より抽象的な説明文を生成することなどが挙げられる．

謝辞

本研究の一部は，頭脳循環を加速する戦略的国際研究ネットワーク推進プログラムの助成を受け実施したものである．

参考文献

(1) 小田悠介ら: “ソースコード構文木からの統計的自動コメント生成,” In Proc. IPSJ SIG-NL 219 (2014)
(2) Giriprasad Sridhara, et al.: “Towards Automatically Generating Summary Comments for Java Methods,” In Proc. ASE (2010)