

ユーザ定義を取り込んだプログラムパターンの翻訳

5 A E - 2

福田 薫 松本 憲幸 岡山 敬 金子 博*

株式会社 東芝**

1. はじめに

ソースプログラムの意味情報の解釈を支援するツール開発のための要素技術として、ソースプログラムに対してステートメント単位で自然言語コメントを付加するプログラムパターン翻訳ツールの研究を行っている。第1段階として開発したプログラムパターン翻訳ツール(以降 PPT-V0)[1]は、C言語の文法の意味やシステムライブラリに含まれる関数や変数に関する知識ベースを利用して翻訳を行う。

(図1)

実際のユーザプログラムには、ユーザが定

義した関数や変数が出現するが、これらのユーザ定義の意味定義を PPT-V0 の処理の枠内で処理しようとした場合、システムライブラリ知識を利用する場合とは異なり、アプリケーション、ソースファイル、関数、処理ブロックなどユーザ定義の有効範囲の切り替えに対応できない。

本稿では、ユーザがC言語で定義した関数や変数の意味定義を利用した翻訳を可能としたプログラムパターン翻訳ツール PPT-V1 の処理方式と効果について説明する。

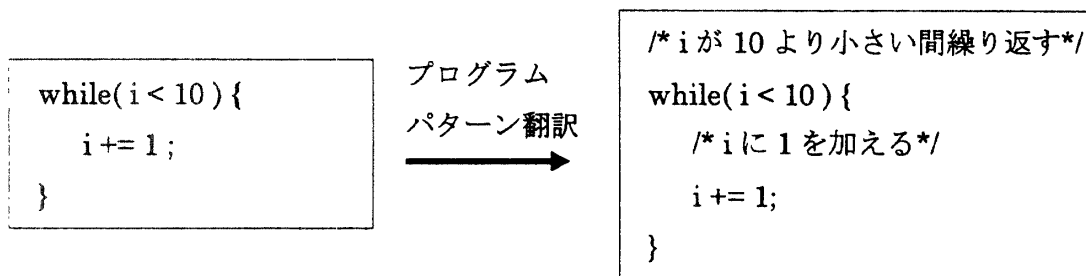


図1 プログラムパターン翻訳の実行例

2. 翻訳ツールの機構

PPT-V0 の枠組みでは、ソースプログラムファイル単位で翻訳処理を実行するため、翻訳対象となるソースプログラムファイルの処理中に意味の変化を生じないグローバルおよび static 関数・変数に関しては、その情報を定義した知識ベースを翻訳時に指定し、取り込むことで対処可能となる。

Translation of C Language Program Patterns considering the user's definition

*Kaori Fukuda, Noriyoshi Matsumoto

Takashi Okayama, Hiroshi Kaneko

**TOSHIBA Corporation

一方、関数やその内部の処理ブロックの中で定義を生じる意味情報に関しては、翻訳処理中に動的に意味情報を変化させながら解釈する必要がある。C言語では、関数に関してはこのようなスコープは存在しないため、実際に翻訳に影響を与えるのは局所的に定義された変数や構造体の意味情報に限定される。

文脈に依存しない意味情報定義は、用語辞書として記憶するのが自然であるが、文脈に依存して変化する意味情報定義は、文脈上に配置するのが自然である。これは、文脈依存の意味情報定義は、本来ソースブ

プログラムの文脈上に配置するのが適切なことを意味する。一方、static 定義またはそれよりも広いスコープを持つ意味定義のものは、参照効率の面から用語辞書を利用した方が有利である。

以上を考慮し、PPT-V1 は、すべてのユーザ定義による意味情報はプログラムの文脈上に知識ベース制御コメントとして記述し、翻訳時にユーザ用語辞書に登録、グローバルおよび static 関数・変数の意味情報は用語辞書に抽出して記憶、局所変数の意味情報は翻訳時にスタック上に記憶する方

式を取る。（図2）

図2において、オリジナルソース中の関数 foo の意味定義 (1) は翻訳時に関数 foo を含むプログラムに対応したユーザ用語辞書に登録される。変数 i の意味定義(2)は実行中の PPT-V1 プロセスのスタック中に記憶され、関数 foo の翻訳完了後は解放される。翻訳ソース中の局所変数 i を含む翻訳コメント(3)および(4)はスタック上に記憶された変数情報を利用し、ユーザ関数 foo を含む翻訳コメント(5)はユーザ用語辞書を利用して翻訳された結果である。

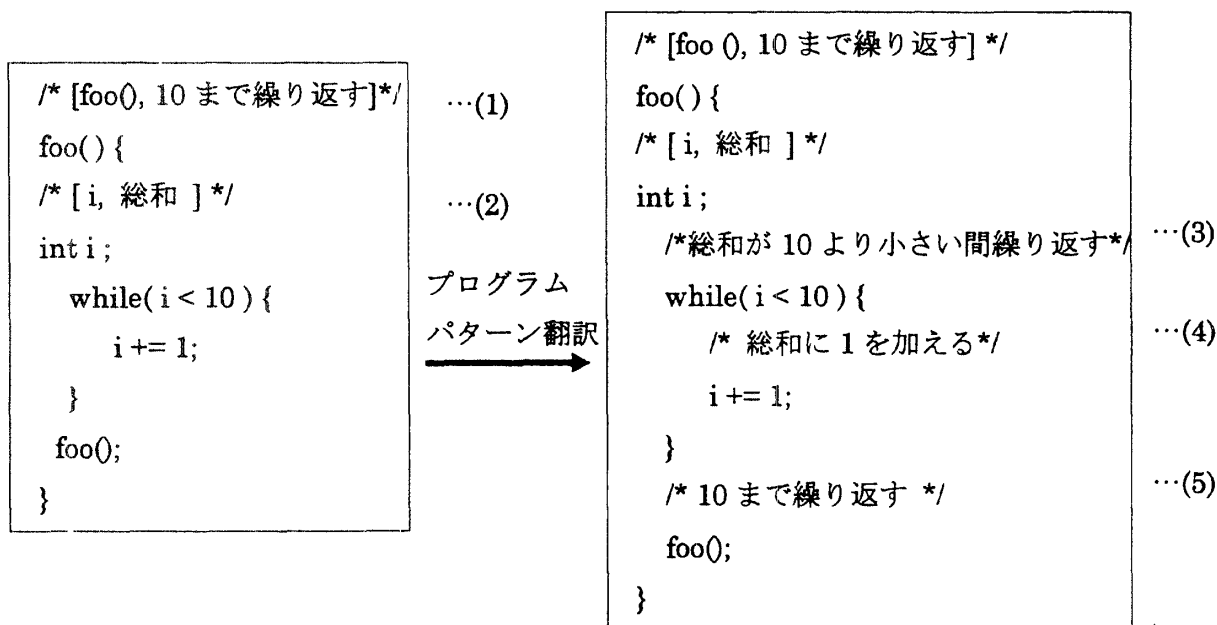


図2 プログラムパターン翻訳 PPT-V1 の処理方式

3. 評価

PPT-V1 により、ユーザ定義の意味情報を取り入れたソースプログラムのパターン翻訳が可能となる。PPT-V1 レベルの支援ツールの存在を前提とした場合、プログラム中に記述するコメントは、関数や変数の定義位置で意味情報を記述すれば、それらの参照位置では PPT-V1 が自動的に表示するため、実行ステートメントに対する過剰

なコメント付けは不要となる。

4. 参考文献

- [1] 岡山敬, 福田薫, 金子博, 松本憲幸.: Java 言語, C 言語プログラムパターンの翻訳. 情報処理学会第 54 回全国大会論文集 (1), pp.219-220, 1997.