1J－5

プログラム依存表現に基づく統合的ソフトウェア開発支援環境について

乃村能成　蒲池正幸　笠原義晃　程京徳　牛島和夫
九州大学 工学部 情報工学科

1. はじめに

プログラム中には本質的に必ず何らかの依存関係が存在する。プログラムの各文の間に存在する依存関係を把握することは、多くのソフトウェア開発活動に要求される。我々の研究室では、プログラム依存表現に基づいて、ソフトウェアの解析、テスト、デバッグ、保守、複雑さ計測を支援する統合的ソフトウェア開発支援環境の構築を目指して研究を進めている。本論文では、我々が構築している統合的開発環境の概要と特徴について述べる。

2. 諸概念の説明

まず、プログラム中に存在する、5つの依存関係について説明し、それらを用いた依存グラフ、依存ネットについて説明する。

2.1 制御依存性

制御依存性は1つのプロセスの中に存在する制御の流れの分岐によって発生する依存関係である。if文などにおける条件式vの評価によって文uが実行されるかどうかが決まる場合にuはvに制御依存しているという。

図1のAdaプログラム片について、例えば、図中のp_7,p_8の実行は、p_6の制御述語の真偽に依存している。つまり、p_7,p_8はp_6に制御依存している。

2.2 選択依存性

制御の流れが非決定的選択によって変化する場合には、非決定的選択文と選択肢の間には選択依存性が存在する。非決定的選択とは、プログラムに同じ入力が与えられても、各タスクの実行タイミングにより実行のたびに選択肢が変わるような選択のことをいう。

例えば、f_9, f_{12}の実行がf_8のselect文によって非決定的に選択されるので、f_9, f_{12}はf_8に選択依存している。

2.3 データ依存性

データ依存性は1つのプロセスの中に存在するデータの流れによって発生する依存関係である。文vで計算されている変数が文uで計算される変数の値に影響を与える場合にuはvにデータ依存しているという。

例えば、p_9における変数Pの値は、p_8におけるPの定義が伝搬しているので、p_9はp_8にデータ依存している。

2.4 同期依存性

同期依存性は2つのプロセス間で同期を取ることによって発生する依存関係である。あるプロセスに存在する文vの処理の開始や終了によって別のプロセスに存在する文uの処理の開始や終了が影響を受ける可能性がある場合にuはvに同期依存しているという。

例えば、f_9の実行は、別のタスクにあるp_9の実行によって引き起こされるので、f_9はp_9に同期依存している。

2.5 通信依存性

通信依存性は2つのプロセス間で通信を行うことによって発生する依存関係である。あるプロセスに存在する文vで計算される変数が別のプロセスに存在する文uで計算される変数の値に影響を与える可能性がある場合にuはvに通信依存しているという。

例えば、f_{10}で使用されている変数Iは、p_8で定義されている変数Pが引数として渡されたものである。つまり、f_{10}はp_8に通信依存している。

2.6 依存関係の表現

逐次プログラム中の制御の流れ、変数の定義と使用を表す有向グラフを定義使用グラフと呼び、並行プログラム中の制御の流れ、変数の定義と使用、各プロセス間の通信、同期を表す有向グラフを定義使用ネットと呼ぶ。

また、逐次プログラム中の制御依存、データ依存を表した有向グラフをプログラム依存グラフと呼び、並行プログラム中の5つの依存関係を表す有向グラフをプロセス依存ネットとよぶ。

本論文ではこれらの厳密な形式的定義は省略する。詳しくは参考文献[1][2]を参照されたい。

```
p1        task body PERCENT is
:
p5          PUT("MINUS_OPERATOR");
p6        elsif Y = 0 then
p7          PUT("ZERO_DIVIDE");
          else
p8          P := (X/Y)*100;
p9          FACT.E1(P);
          end if;
p10       end PERCENT;
f1        task body FACT is
:
f8          select
f9            accept E1(I: in INTEGER) do
f10             F := F*I;
f11           end E1;
          or
f12           accept E2(O: out INTEGER) do
f13             O := F;
f14           end E2;
          end select;
f15       end FACT;
```

図1: Adaプログラム片

3. システムの構成と特徴

従来の統合的ソフトウェア開発環境では、各ツールの機能、ツール間のインターフェイスが特化されている。しかし、プログラム依存表現はプログラム開発のあらゆる場面において必要とされるので、各ツール間のインターフェイスを統一し、用いることによって得られる

An integrated software development environment based on program dependences
Yoshonari Nomura, Masayuki Kamachi, Yoshiaki Kasahara, Jingde Cheng and Kazuo Ushijima
Kyushu University

利益は大きいと考える。この考え方に基づいて我々は、各種プログラムから、そのプログラム依存表現への変換ツール群を開発している。これらは、ソフトウェアの解析、テスト、デバッグ、保守、複雑さ計測などのプログラム開発活動における多くの場面で利用できる。

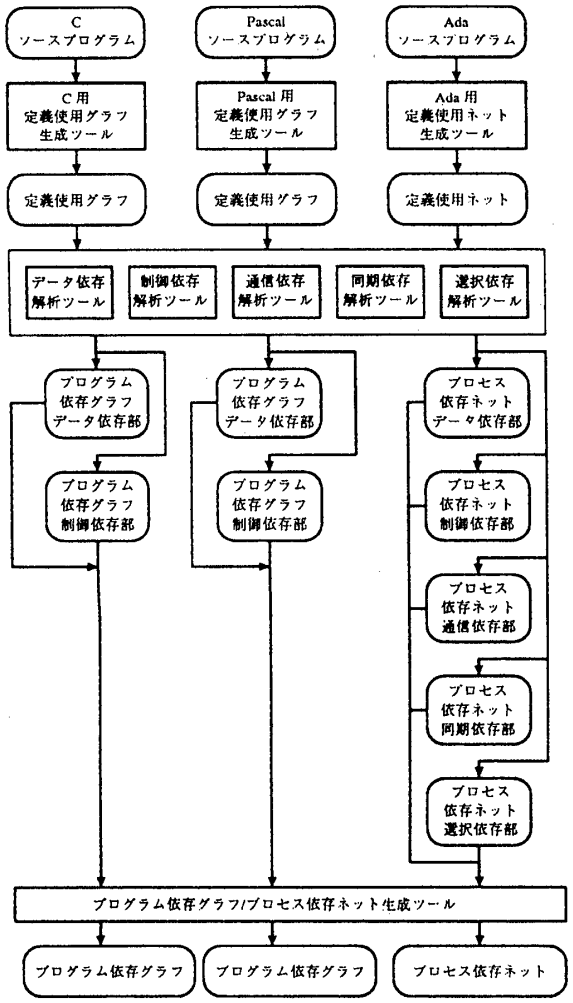

図 2: システムの構成図

図2に本システムの構成図を示す。本システムでは、ソースプログラムから定義使用グラフ(ネット)を生成するツール、定義使用グラフ(ネット)から各種依存関係を解析するツール、出力された各種依存関係を統合してプログラム依存グラフ(プロセス依存ネット)を生成する最終出力部の大別して3種を提供する。

3.1 定義使用グラフ(ネット)生成ツール

まず、ソースプログラムに対応する定義使用グラフ(ネット)生成ツールによって定義使用グラフ(ネット)を生成する。現在は、C, Pascal, Ada, Occam2についての開発を進めている。このツールは、ソースプログラムをその定義使用グラフ(ネット)に変換するだけではなく、ソースプログラムとの対応をとるために各種の名前表についての情報も出力する。これらの情報は、プログラム依存グラフ(プロセス依存ネット)を生成するために使用される。生成された定義使用グラフ(ネット)は、テキストの形で出力される。

3.2 依存関係解析ツール

次に、各種の依存関係解析ツールによって各種依存関係を抽出する。データ依存、制御依存解析ツールにより逐次プログラムの定義使用グラフから依存関係を抽出する。また、制御依存、データ依存、選択依存、通信依存、同期依存解析ツールにより並行プログラムの定義使用ネットからそれぞれの依存関係を抽出する。抽出された依存関係は、テキストの形で出力される。

定義使用グラフ(ネット)は、言語に依存しないので、一度ソースプログラムから生成された定義使用グラフ(ネット)は、言語の特徴を持たない。唯一、並行プログラムであるか逐次プログラムであるかの違いしか残らない。

従って、本システムの各種依存関係解析ツールは、対象プログラムがどの言語で書かれていたかを考慮する必要がないため、定義使用グラフ(ネット)をそれぞれの依存関係ごとに単一のツールで解析することができる。

3.3 最終出力部

最後に、プログラム依存グラフ(プロセス依存ネット)生成ツールが、得られた情報を統合して、テキストの形で出力する。

3.1から3.3で述べたツール群は、以下の特徴を持っている。

ツールは単一ではなくツール群として構成されている。それは、実際の開発の場面においては、全種類の依存関係を扱う必要がないため、選択的に依存関係を抽出できる方がより便利であると考えたからである。ユーザは、解析したい依存関係に応じてツール群の中からツールを選択することによって目的の出力を得ることができる。

また、解析結果を利用しやすいようにするため、全てのツールはテキストの形で入出力を行う。

各ツールによって出力される依存関係は、ソースプログラム言語に依存しない。そのため、ソフトウェアをグループによって開発している場合、個々の開発メンバが、どのような言語を使用しているかを気にすることなく、お互いのソースプログラムから抽出した依存関係の情報を共有することができる。

4. おわりに

本論文では、我々が構築している統合的開発環境の概要と特徴について述べた。現在そのプロトタイプの開発を進めている。今後は対応する言語を増やしていく。さらにその上でソフトウェアの解析、テスト、デバッグ、保守、複雑さ計測を行うための応用ツール群の開発を行う予定である。

参考文献

[1] J. Cheng, "Task Dependence Net as a Representation for Concurrent Ada Programs," in J. van Katwijk (ed.), "Ada: Moving towards 2000," LNCS, Vol. 603, pp. 150-164, 1992.

[2] J. Cheng, "The Tasking Dependence Net in Ada Software Development," ACM Ada Letters, Vol. 12, No. 4, pp. 24-35, 1992.

[3] J. Cheng, Y. Kasahara, M. Kamachi, Y. Nomura, and K. Ushijima, "Compiling Programs to Their Dependence-Based Representations", to appear in Proc. 1993 IEEE TENCON, Oct., 1993.