アスペクト指向プログラミングの動的プログラムスライスへの応用

Application of Aspect-Oriented Programming to Calculation of Dynamic Program Slice

石尾 隆[*1] 楠本 真二[*1] 井上 克郎[*1]
Takashi Ishio Shinji Kusumoto Katsuro Inoue

[*1] 大阪大学大学院基礎工学研究科
Graduate School of Engineering Science, Osaka University

1. まえがき

プログラムデバッグを効率よく行うための手法として、プログラムスライスが提案されてきている。一般に、プログラムスライスは制御依存関係、データ依存関係を解析することで得られる。オブジェクト指向言語では実行時に決定される要素を含むため、動的解析が必要となるが、その収集には多大なコストを必要とする。

一方、アスペクト指向プログラミングでは、非機能的要求をモジュール化するアスペクトというモジュール単位を導入している。本研究では、Java における動的情報収集モジュールを AspectJ を用いて記述し、得られる再利用性とオーバーヘッドの軽減について考察する。

2. アスペクト指向プログラミング

アスペクト指向プログラミングは、オブジェクト指向プログラミングではうまくモデル化できない非機能的要求をオブジェクトから分離し、独立したモジュール単位「アスペクト(Aspect)」として記述することで再利用性、モジュール性を向上させる[1]。

一般に、複数のオブジェクトを横断するような要求を一つのオブジェクトにカプセル化することは困難である。そのような要求に関するコードをアスペクトとしてオブジェクトから分離することで、オブジェクトとアスペクトに独立した再利用の道を提供する。

3. プログラムスライス

プログラムスライスとは、プログラム中のある変数 v に対して、v の値に影響を与える全ての文をプログラムから抽出する技術で、その結果取り出された文の集合をプログラムスライスまたは単にスライス(Slice)と呼ぶ。スライスはデバッグ、保守、プログラム理解等に利用される。スライスのうち、全ての可能なデータに関するスライスを静的スライスと呼び、ある特定の入力のもとでの実行系列を解析したスライスを動的スライスと呼ぶ[2]。

オブジェクト指向言語で書かれたプログラムは、オブジェクトの参照・動的束縛など、動的に決定される要素を含む。そのため、静的スライスでは得られる精度に限界があり、動的スライスが必要となる。

4. 実行時情報の収集

4.1 従来の手法

プログラムの実行時情報を収集するには様々な手法があるが、Java では、次のような手段が考えられる。

(a) Java Virtual Machine(JVM)の改造
(b) Java Virtual Machine Profiler Interface(JVMPI)の使用
(c) 監視用命令を埋め込む専用プリプロセッサの作成

このうち(a)は、JVM の特定のバージョンに依存した手法である。(b)は、JVM に用意されている性能計測のためのインタフェースで、これは(a)(b)に共通することだが、システム全体を監視するために、注目する必要のないライブラリ内部の動作についても監視のオーバーヘッドが発生する。(c)は、プリプロセッサの保守性や再利用性、他のプリプロセッサとの競合などの問題がある。

4.2 アスペクトによる実現

プログラムの実行時情報の収集は、計測対象となるプログラムのオブジェクトを横断した非機能的要求である。これを一つのアスペクトとして記述し、対象のプログラムに結合することを考える。

Java プログラムの実行時情報を収集するアスペクトを、AspectJ を用いて作成した。AspectJ は、このアスペクトをコンパイル時に計測対象のプログラムに結合する。この情報収集モジュールは、次のような特徴を持つ。

(a) 処理そのものは Java で平易に記述できる。
(b) 標準 JVM で動作し、JIT コンパイラも利用できる。
(c) 複数の情報収集モジュールを同時に使用できる。
(d) 情報収集の範囲を指定できる。

アスペクトを用いた情報収集では、従来の手法と比べて得られる情報に制限があり、最終的に計算できるスライスの精度に影響するが、実行オーバーヘッドを軽減することで、より大規模なプログラムに対するスライス計算への対応が期待できる。また、作成したアスペクトは、主に情報を記録するための print 文から構成された 100 行程度のモジュールとなった。これはユーザが容易に理解し、利用できると考えられる。

5. まとめ

プログラムの実行時情報を取得するプログラムをアスペクトとして記述することで、情報取得プログラムの保守性と再利用性を向上させることができる。また、ユーザがアスペクトの結合箇所を容易にカスタマイズでき、対象プログラムのある特定の範囲の解析が可能である。

参考文献

[1] G. Kiczale et al. : “Aspect Oriented Programming”, In Proc. of ECOOP, vol.1241 of LNCS, pp.220-242(1997).
[2] F. Ohata et al.: “A Slicing Method for Object-Oriented Programs Using Lightweight Dynamic Information”, In Proc. of APSEC2001, pp.273-280(2001).