6H−6

静的解析と動的解析を併用した JAVA プログラムスライス手法の提案と実現

藤井将人　廣瀬航也　大畑文明　井上克郎

大阪大学 大学院基礎工学研究科

1 まえがき

デバッグを効率良く行なう手法の一つにプログラムスライス（*Program Slice*，以降，単にスライスと呼ぶ）がある [4]．スライスとは，直観的には，ある文 s のある変数 v の値（スライス基準 (s, v)（*Slicing Criterion*）と呼ぶ）に影響を与える可能性のある文の集合であり，プログラム文間の依存関係（*Dependency Relation*）を解析することで求められる．

スライスには，大きく静的スライス（*Static Slice*）[4] と動的スライス（*Dynamic Slice*）[2] があるが，これらには正確性，コストに関してそれぞれ問題があった．そこで，静的，動的スライスの中間に位置する **DC** スライス（*Dependence-Cache Slice*）[1] が提案されている．

しかし，[1] は手続き型言語のみを対象としており，オブジェクト指向（OO）言語を考慮したものとなっていない．本稿では，DC スライスの OO 言語への適用における問題点を考察し，具体的な言語として JAVA を対象としたスライス手法の提案を行う．また，提案手法をスライスシステムとして実装し，有効性の確認を行った．

2 Dependence-Cache(DC) スライス

プログラム文間の依存関係には，データ依存関係（*Data Dependency Relation*）と制御依存関係（*Control Dependency Relation*）がある．DC スライスでは，制御依存関係を静的に，データ依存関係を動的に解析する．特定の入力データによる解析のため，静的スライスに比べスライスサイズは小さく，実行系列を保存しないため，動的スライスに比べコストは十分に小さい．表 1 において，静的スライス，動的スライス，DC スライスの特徴を示す．これらは，[1] での実験により確認されている．

2.1 計算手順

以下に DC スライスの計算手順を示す．なお，グラフ到達問題によるスライス計算は [3] に基づく．

Step 1: 静的制御依存関係解析

次の条件をすべて満たすとき，文 s から文 t への制御依存関係があるという．

- s が条件文である
- t の実行は，s の結果に依存する

Step 2: 動的データ依存関係解析

以下の条件をすべて満たすとき，文 s から文 t への変数 v に関するデータ依存関係があるという．

- s で v を定義する
- t で v を参照する
- s における v の定義が t に到達する（v を再定義しない経路が s〜t 間に少なくとも 1 つ存在する）

詳細は 2.2 で述べる．

Step 3: プログラム依存グラフ構築

プログラム依存グラフ（*Program Dependence Graph, PDG*）とは，プログラムに含まれる代入文，入出力文，条件文，手続き呼出し文などの各文を節点で，2 つの文間の依存関係を辺で表わした有向グラフである．

Step 4: スライスの抽出

スライス基準に対応する節点から PDG 辺を逆向きにたどる．到達可能な節点集合に対応する文がスライスとして抽出される．

表 1: スライス手法の比較

	静的スライス	DC スライス	動的スライス
制御依存関係	静的	静的	動的
データ依存関係	静的	動的	動的
解析対象	文	文	実行系列
解析コスト	小	中	大
スライスサイズ	大	中	小

2.2 動的データ依存関係解析

実行中，文 s で変数 v が使用されたとき，その値を定義した文 t が分かれば，t から s への v に関する動的なデータ依存関係が抽出できる．

そこで，プログラムに現れる各変数 v に対してキャッシュ $C(v)$ を用意する．プログラムの各実行時点において，$C(v)$ には v を最後に定義した文番号が格納されている．s の実行において，v が参照されたとき，$C(v)$ から s への v に関するデータ依存関係として抽出する．また，v が定義されたとき，$C(v)$ の値を s の文番号に更新する．

3 Java プログラムスライス手法

本節では，前節で紹介した DC スライスの OO 言語への適用における問題点を考察し，JAVA を具体的な対象言語とした，オブジェクト指向 **DC** スライス（*Object-Oriented Dependence-Cache Slice*）の提案およびその実現について述べる．

The Proposal and Implementation of Slicing Method of JAVAPrograms with Static and Dynamic Analysis.

Masato FUJII, Koya HIROSE, Fumiaki OHATA and Katsuro INOUE

Graduate School of Engineering Science, Osaka University

3.1 方針

OO 言語では，手続き型言語での配列，ポインタに加え，動的束縛（*Dynamic Binding*）が動的決定要素として挙げられる．また，カプセル化（*Encapsulation*）により，インスタンス属性へのアクセスは一般にメソッドを介して行われるため，動的束縛に対する解析方針の違いによるスライス抽出精度への影響は大きい．静的解析により“呼び出される可能性のある”メソッドを抽出する手法も考えられるが，唯一に絞ることは難しく，十分な解析精度は得られない．そこで，DC スライスの OO 言語への適用においては，データ依存関係に加え，メソッド呼び出しに関する制御依存関係を動的に解析することが考えられる．

また，各インスタンスの属性はそれぞれ独立した存在であり，それらに関するデータ依存関係を独立に解析する必要がある．そこで，各変数に対応するキャッシュをその変数が生成されると同時に生成し，各インスタンス変数に関するキャッシュも同様に行うようにした．これにより，インスタンスごとに独立した解析が可能となる．

3.2 計算手順

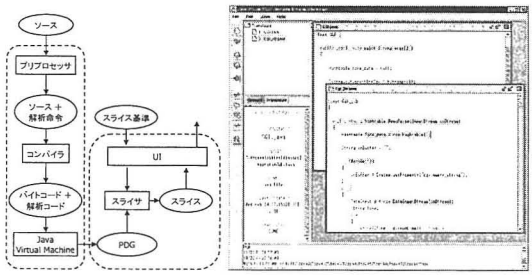
提案手法の計算手順について，2.1 で示した DC スライス計算手順からの変更点のみ示す．

Step 1’: 静的制御依存関係解析（メソッド呼び出しを除く）

Step 2’: 動的データ依存関係解析 ＋
（メソッド呼び出しに関する）静的制御依存関係解析

3.3 実現: Java スライスシステム

提案手法を解析コード埋め込み（プリプロセッサ）方式によるスライスツールとして実装し，GUI 部の作成も行った．いずれも，Java を開発言語として利用した．図 1 に，システムの概要および画面スナップを示す．



(a) 概要　　(b) 画面スナップ

図 1: Java スライスシステム

3.4 評価

実装した Java スライスシステム用い，静的スライス，動的スライスとの比較を行った．実験には，HTML ファイルを出力する CGI プログラム P_1，マウス操作によるペイントアプリケーション P_2 を用いた．なお，P_2 については対話型プログラムであり実行時間の正確な計測が困難なため，実行時間を計測していない．

表 2: スライスサイズ [行数 (全体に対する割合)]

	静的スライス	動的スライス	提案手法
P_1-スライス基準$_1$	26 (11.7%)	15 (6.7%)	**15** (6.7%)
P_1-スライス基準$_2$	83 (37.2%)	27 (12.1%)	**27** (12.1%)
P_2-スライス基準$_1$	48 (21.2%)	14 (6.2%)	**14** (6.2%)
P_2-スライス基準$_2$	45 (19.9%)	12 (5.3%)	**12** (5.3%)

表 3: 実行時間 [ms]

プログラム	通常実行	解析付き実行	解析付き実行/通常実行
P_1	138	582	4.22

表 4: 使用メモリ量 [KByte]

プログラム	通常実行	解析付き実行	解析付き実行/通常実行
P_1	478	645	1.35
P_2	836	920	1.10

表 2，表 3，表 4 に，スライスサイズ，実行時間，使用メモリ量をそれぞれ示す．実装システムでは提案手法のみを実現しており解析コストに関する比較はできなかったが，表 1 と同様の結果が予想される．なお，スライスサイズは手計算による．

3.5 考察

今回の実験においては，本手法で得られるスライスは静的スライスの約 30〜60%の大きさであり，また，動的スライスと等しいスライスが得られた．対象プログラムが比較的小規模であったため，各スライスとの差は小さなものであったが，表 1 の性質を満たしていることが確認された．動的決定要素を多く含む大規模プログラムにおいては，その差はより顕著なものになると考えられる．

4 まとめ

本研究では，オブジェクト言語プログラムに対する，動的，静的解析を組み合わせたスライス手法を提案し，Java を対象とした実装システムを用いてその有効性を確認した．提案手法では，適用効果の高い実行時決定要素に対してのみ動的解析を行い，解析コストを抑えながらスライス抽出精度を向上させることができる．

今後の課題としては，スレッド，例外処理への対応，大規模プログラムに対する評価実験がある．なお，スレッドや例外処理に関する制御依存関係解析も動的に行うことで，解析精度の向上を目指したいと考えている．

参考文献

[1] Ashida Y., Ohata F., and Inoue K. : “Slicing Methods Using Static and Dynamic Information”, *Proceedings of the 6th Asia Pacific Software Engineering Conference*, pp. 344–350, 1999.

[2] Korel, B., and Laski, J. : “Dynamic Program Slicing”, *Information Processing Letters*, Vol.29, No,10, pp. 155–163, 1988.

[3] Ueda, R., Inoue, K., and Iida, H. : “A Practical Slice Algorithm for Recursive Programs”, *Proceedings of the International Symposium on Software Engineering for the Next Generation*, pp. 96–106, 1996.

[4] Weiser, M. : “Program Slicing”, *Proceedings of the Fifth International Conference on Software Engineering*, pp. 439–449, 1981.