SOBA: Java バイトコード解析ツールキット

Simple Objects for Bytecode Analysis

石尾 隆　秦野 智臣　井上 克郎

大阪大学大学院情報科学研究科

主な機能

・動的束縛を解決したコールグラフの構築

・メソッドごとの制御フローグラフの構築

・メソッド内制御依存/データ依存関係の計算

特徴は「シンプルな利用法」

(1) 解析対象の JavaProgram オブジェクトを取得し，

(2) その getter メソッドを基点に情報を収集する．

「ちょっとプログラムの情報がほしい」人向け．

プログラム例

引数で指定されたプログラム内のすべての呼び出し関係を列挙し出力するプログラム

```
public static void main(String[] args) {
  JavaProgram program = new JavaProgram(ClasspathUtil.getClassList(args));
  ClassHierarchy ch = program.getClassHierarchy();

  for (ClassInfo c: program.getClasses()) {
    for (MethodInfo m: c.getMethods()) {
      System.out.println(m.toLongString());
      for (CallSite cs: m.getCallSites()) {
        MethodInfo[] callees = ch.resolveCall(cs);
        if (callees.length > 0) {
          for (MethodInfo callee: callees) {
            System.out.println("  [inside] " + callee.toLongString());
          }
        } else {
          System.out.println("  [outside] " + cs.toString());
        }
} } } }
```

ディレクトリ，JAR/ZIP，クラスファイル単体を自動で判別して読み込む．入れ子になったJARにも対応．

プログラム内部のすべてのクラスの内容を階層的なコレクションとして扱える．解析に使用している ASM ライブラリの生情報にもアクセス可能．

呼び出し命令（CallSite）から動的束縛を解決．CHAとVTAの２種類の解析アルゴリズムを提供．

メソッド名や引数の型などの基本情報を利用可能．

実行例

プログラム例自体を解析した場合の出力

```
soba/example/dump/DumpMethodCall.main(java/lang/String[]:args): void
  [inside] soba/util/files/ClasspathUtil.getClassList(java/lang/String
      []:files): soba/util/files/IClassList[]
  [inside] soba/core/JavaProgram.<init>(soba/core/JavaProgram:this, soba/
      util/files/IClassList[]:lists): void
  [inside] soba/core/JavaProgram.getClassHierarchy(soba/core/JavaProgram:
      this): soba/core/ClassHierarchy
  [inside] soba/core/JavaProgram.getClasses(soba/core/JavaProgram:this):
      java/util/List
  [outside] java/util/List.iterator()Ljava/util/Iterator; called by soba/
      example/dump/DumpMethodCall.main(java/lang/String[]:args): void
  [outside] java/util/Iterator.next()Ljava/lang/Object; called by soba/
      example/dump/DumpMethodCall.main(java/lang/String[]:args): void
…（以下省略）
```

[大規模プログラムも解析可能]

Intel Xeon E5-2620 2.0GHz, RAM 64GB, 1スレッド環境において，Eclipse 4.4.1, Oracle JDK 1.8.0 の 99,770クラス，774,261メソッドから101,930,644個の呼出し関係を42分（3.2msec/method）で抽出

教えてください！

機能追加やドキュメントの整備に向けて，何があれば「十分」なのかを知りたいと思っています．

Q1. これまでどのような情報を使いましたか？

Q2. これからどのような情報がほしいですか？

Q3. どんな解説やコード例がほしいですか？

大学での卒論・修論，企業での開発など，皆様の状況と，ご要望，ご意見をお聞かせください．

https://osdn.jp/projects/soba/　MITライセンスでソースコード公開中．ドキュメントは準備中．

Software Engineering Laboratory