動的解析に適したインタプリタに関する研究

蔡 遠航

北陸先端科学技術大学院大学 情報科学研究科

2001 年 2 月 15 日

キーワード: 動的解析, インタプリタ, 実行情報, Sapid, CASE ツールプラットフォーム.

ソースプログラムの解析には、静的解析と動的解析がある。ソフトウェアの開発保守において、プログラムのデバッグやテストなどが重要な役割を持ち、これらには動的解析が有効とされている。動的解析とは、ある入力を与えてプログラムを実行し、任意の実行時点におけるプログラム状態の情報 (実行情報) を用いてソースプログラムを解析することをいう。

しかし一般的に、動的解析は静的解析よりも解析器の作成が困難である。静的解析が扱う抽象構文木とシンボルテーブルに加えて、動的解析器は、スタック、ヒープ、レジスタなども扱う。既存の動的解析器ではこれらの動的情報の抽象化やモデル化が不十分であり、動的解析器が特有のフォーマットを持つ情報として存在し、また外部に提供する仕組みがないため、再利用性が低い。現状では、動的解析ツールごとに解析器が作られている。解析器を作る作業は、動的解析器作成の本質ではないにもかかわらず、コストのかかる作業である。

抽象化やモデル化されたプログラムの実行情報を提供するには、プログラムをコンパイラ、インタプリタなどを用いて実行し、それと同時に実行情報を格納して、その後実行情報の抽象化やモデル化を行うのが一般的なやり方である。

コンパイル済みのプログラムやインタプリタを実行すると，独自の動的な情報をもとにして、実行をする。この場合、システムごとに異なるフォーマットを使ってその独自の動的な情報を解釈する、という余計な作業が必要になる。この作業を略し、モデル化や抽象化して、動的解析により使いやすい動的な情報が求められている。しかし、現状ではこの動的な情報を提供する解釈器が少ない。

一方、CASE ツールのプラットホームになることを目的に、細粒度リポジトリとして Sapid が提案されている。これは、I-model というモデルに基づいて作られたソフトウエアリポジトリである。I-model に基づいたリポジトリは、これに基づいてソースコードを

静的解析した結果を直接解釈実行することが可能な仮想機械を作成できるなど、これまでのモジュールや関数単位のリポジトリに比べて、十分に細かい粒度を持っている。このことは、Sapid が動的解析に有効であることを示している。

Sapid には C 言語インタプリタ Sint が提供されている。しかし、Sint によるプログラムの実行時の動的な情報を提供するのは不十分である。抽象化やモデル化された実行情報を提供していないからである。

したがって、Sapid は静的な情報を提供するのが優れているに対し、動的な情報を提供するにはまだ不十分である。Sint を用いてプログラム実行情報を格納して、抽象化やモデル化して提供できれば、CASE ツールを開発するときにしばしば必要になる静的解析と動的解析機能に優れる解析器を提供することができ、開発の省力化や CASE ツール間のデータ統合なども期待できる。

以上のことから、動的解析に適したプログラムの解析器をを提案、実装する際、新しい解釈器を初めから作るよりも Sint を拡張した方が実装の省力化ができ、CASE ツールプラットフォームとしての Sapid の本質に相応しい。

本研究では、動的解析のための、実行情報に関するモデルと API を提案した。それに基づいて、Sint を拡張し、動的解析に適したインタプリタを提案、実装した。

これの実現には、まず、インタプリタが動的解析に適すために、スライシングとデバッグを例に取り、動的解析への要求事項や実行情報を論理的に扱うことの利点について考察した。さらに、実行情報をモデル化する際、「抽象度が高いかつ汎用性も高いモデルの提供」を念頭におき、抽象度と汎用性の関係を論じた。

これらをもとに実行情報を選別し、実行時点、実行された文、変数、変数の値、領域、関数呼び出し関係と返り値を表すスタックの状態といった 6 つの情報に着目して、これらに関するモデルを提案した。また、Sint 及び Sint が用いる動的解析を支援するライブラリ SIP2 の両方を拡張して、上述の 6 つの実行情報を抽出、格納するように実装した。実行情報モデルは 6 つのクラスと 6 つの関連からなり、また、モデルと一緒に 9 つの API を提供した。そして、このインタプリタを用いて、関数呼び出し関係を調べるツールと変数履歴ツールの理論的な実現手法を論じ、このインタプリタを用いて、動的解析ツールを簡単に実現するために一定の有効性を持つことを示した。

本研究で提案、実装したインタプリタの実現により、変数やスタックといった論理的な概念を直接扱うことが部分的にできるようになった。また、動的解析ツールを開発する際にしばしば必要となる解析器を提供することができ、開発の省力化や CASE ツール間のデータ統合なども期待できる。