第3章 C/C++プログラムのバグ検出！品質＆効率アップ！

あのFacebookも！世界の解析/検証ソフト

東 工太郎

● 検証用ソフトウェアの分類

C/C++プログラムの開発で役に立つデバッグ/検証用ソフトウェアは，動的解析に基づくものと静的解析に基づくものの2種類に分類できます．

▶動的解析

対象プログラムを実際に実行して得られる情報に基づきデバッグ/検証を支援する方式です．例えば，対象プログラム実行時のメモリ・アクセスのログを採取して不正なポインタ操作を検出するツールなどは動的解析に基づくデバッガに分類できます．

▶静的解析

プログラムの仕様や構造などプログラムを実行せずに得られる情報に基づき，デバッグ/検証を支援する方式です．例えば，対象プログラムのソースコードを解析し特定の制御フロー上でメモリ・リークを検出するツールなどは静的解析に基づくデバッガに分類できます．

一般的な傾向として，動的解析ツールはバグの誤検出が少ないものの検出漏れが多いという特徴があります．静的解析ツールは，検出漏れが少ないが誤検出が多いという特徴があります．実際の開発では，これらの違いを理解したうえで，用途に応じてツールを使い分ける必要があります．

本稿では次の内容を解説します．

(1) デバッグ/検証用ツールとして広く使われている動的/静的ソフトウェア
(2) 動的解析ソフトValgrindを使った並列処理の代表的バグであるデータ競合の検出実験

動的解析ソフト①バイナリ・レベル検査Valgrind

● 対象プログラムの性能や問題点を特定する

Valgrindは，対象プログラムの性能や機能の問題点を特定するフレームワークです．**図1**にValgrindによる実行時検査/計測の流れとしくみを示します．

Valgrindを利用すれば，次のような種々の問題点の特定ができます．

- 対象プログラムのキャッシュ・アクセスを計測し，ヒット率が十分でない場合を特定する
- メモリ・アクセスを検査し境界違反やメモリ・リークを特定する
- マルチスレッド処理を検査し，誤った共有データ・アクセス(データ競合)を特定する

問題点を特定するために，Valgrindは対象プログラムを実際に実行しながら実行中に生じる種々のイベ

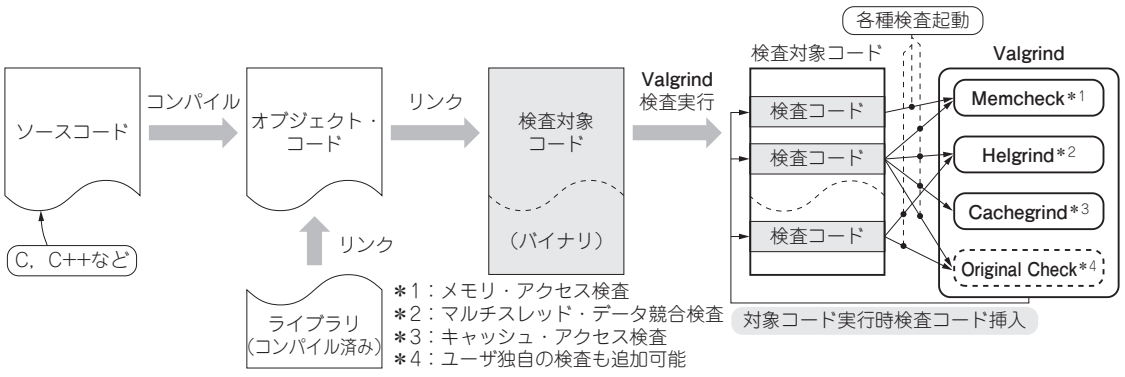

図1 バイナリ・レベルで解析できる！Valglindによる実行時検査/計測フロー