様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　６ 年　 ５ 月 １０ 日現在

機関番号：１２６０８
研究種目：基盤研究(C)（一般）
研究期間：2019～2023
課題番号：１９Ｋ１１８９７

研究課題名（和文）ソフトウェア追跡性とソフトウェア解析技術の融合

研究課題名（英文）Fusion of software traceability and software analysis techniques

研究代表者
権藤　克彦（Gondow, Katsuhiko）

東京工業大学・情報理工学院・教授

研究者番号：５０２６２２８３

交付決定額（研究期間全体）：（直接経費）　　3,300,000円

研究成果の概要（和文）：主な研究成果は次の通り：深層学習を用いたLearn&Fuzzのカバレッジ計測による定量化，JavaScriptのasync/await機能の可視化ツールAwaitVizの提案・実装，カラー画像コードRICを用いた動画配信の新しいビジネスモデルの提案，イベント駆動型のJavaScriptに対する効率的なコンコリック解析，並行プログラムに対する中間コードレベルでの逆実行による状態復元，プログラマの意図抽出と変数名の一貫性チェック，プログラマによる原始データ型を使った高レベルな概念の識別手法，Swift言語のARC機能での強い循環参照やメモリリークの自動検知手法．

研究成果の学術的意義や社会的意義
コード理解，バグ修正の正しさの確認，影響範囲の把握などに役立つため，ソフトウェア工学上，追跡性リンクの確保は非常に重要である．しかし，ソフトウェア開発において追跡性の担保は有用であるが実現が難しい．またもう一つの背景として，ソフトウェア解析は自動で大量のコードを処理可能だが，精度が悪い．また，プログラマの意図は解析できないという問題がある．本研究が目指す，この２つの技術「追跡性リンクの担保」と「ソフトウェア解析」の融合では両者の欠点を補い，解析精度の向上や「プログラマの意図」の保存を可能にする．これはソフトウェア開発の費用の大幅な削減と品質の向上につながるという意味で学術的社会的意義がある．

研究成果の概要（英文）：The main results of our research are as follows: quantification of learn& fuzz coverage measurement, a visualization tool AwaitViz for JavaScript async/await features, a new business model for video distribution using color image code RIC, efficient concolic analysis for event-driven JavaScript, state recovery for concurrent programs by reverse execution at the intermediate code level, programmer's intention extraction and consistency checking of variable names, programmers' high-level concept identification methods using primitive data types, and automatic detection of strong retain cycles and memory leaks in the Swift language's ARC feature.

研究分野：ソフトウェア工学

キーワード：ソフトウェア解析　追跡性リンク

様　式　Ｃ－１９、Ｆ－１９－１（共通）

１．研究開始当初の背景

ソフトウェア開発において追跡性の担保は有用であるが実現が難しい．ソフトウェアの追跡性とは「ソフトウェアが満たすべき要求等が，実装コード等のどこに反映されたか」を双方向に確認できる性質である．追跡性をたどるリンクを追跡性リンクという．コード理解，バグ修正の正しさの確認，影響範囲の把握などに役立つため，ソフトウェア工学上，追跡性リンクの確保は非常に重要である．
手動によりソフトウェア開発中に追跡性を確保するアプローチを時中型(prospective), 開発後に復元を試みるアプローチを事後型(retrospective)と呼ぶ．時中型は人手の手動コストが高く，事後型は単語の類似度計算に基づくため精度が悪い（特に誤検出が多い）という欠点がある．
この問題に対し，我々は先行研究でハッシュを用いた時中型追跡手法を提案・実装した．本研究の前身である我々の研究（科研費基盤 C，課題番号 26330077，H26～30）で，言語独立・ハッシュ値・行粒度の追跡手法を研究開発した．この追跡手法はプログラマのコピー＆ペーストに連動させることで，プログラマの手間を減らしつつ，高精度に追跡性リンクを確保できる．また，ソフトウェア解析が苦手とする（特に直感に反する）「人間の意図」を容易に確実に保存できる．
その一方で，コピー＆ペーストに連動するため，保存できる情報量が少ないという欠点がある．
またもう一つの背景として，ソフトウェア解析は自動で大量のコードを処理可能だが，精度が悪い．また，プログラマの意図は解析できないという問題がある．ソフトウェアのコード解析技術には，大きく，静的解析，動的解析，コンコリック実行解析（動的記号実行解析）の 3 種類があるが, コード網羅率, 解析精度, スケーラビリティなど, 様々な点でトレードオフや欠点がある．
静的解析はすべての実行パスの可能性の網羅を試みるが，変数値などの実行時情報が無いため，例えば，正確な実行フローを得ることが出来ない．例えば，C 言語の関数ポインタを用いたコードがあった場合，「すべての関数が呼び出される」と仮定せざるを得ず，false-positive が多くなる．
動的解析は具体的なある入力値に対して実行を行い解析を行う．実際に実行するため得られた情報は正確だが，通った実行パスに関する情報しか得られない．このため，false-negative が多くなる．
コンコリック実行解析は記号実行手法とソルバを使い, if 文などの条件分岐の条件を反転することで，通っていない実行パスを動的解析し，コード網羅率の上昇を試みる．しかし，ソルバが対応できるのは整数値などに限定されていることが多く，複雑なデータ構造は扱えない．また非決定的な動作をするコードではコード網羅率が改善できない．
また，一般的にソフトウェア解析では「プログラマの意図」を抽出できない．

２．研究の目的

本研究の目的は, 時中型(prospective)な追跡子とソフトウェア解析の技術を組み合わせることで, ソフトウェア追跡の情報量とソフトウェア解析の精度を向上させ，ソフトウェアの保守コストを大幅に減少させることである．
時中型追跡子の少ない情報量をソフトウェア解析で補完することで情報量を増大しつつ,
時中型追跡子で表現した人間の意図で，ソフトウェア解析結果を補完して精度の向上を図る．これにより時中型追跡子とソフトウェア解析の欠点を解消し，ソフトウェア保守コストの大幅減少を図る．

３．研究の方法

２つの異なる技術である「時中型追跡子」と「ソフトウェア開発」の技術を組み合わせることは自明ではない．そこで以下の王道の研究方法を取った．
・「時中型追跡子」の研究に「ソフトウェア開発」の技術を追加することを試みる．
・また逆に「ソフトウェア開発」の研究に「時中型追跡子」の技術を追加することを試みる．
・以上の研究に合わせ, この分野の研究動向のサーベイを随時行い, 最新の研究成果も取り込む．

４．研究成果

研究の結果，以下の研究成果を得た：

・文法に基づくファジングの最新技術の一つである Learn&Fuzz は深層学習を用いて入力文を自動生成するが，制限が大きいという問題がある．この研究では再現実装を用いて，コードカバレッジを計測することにより，この制限を定量化した．（ソフトウェア解析技術関連）

・JavaScript のコールバック問題を解決するため，async/await という言語機構が導入されたが，async/await を用いて書き換えられたコードの実行順序の理解が難しいという問題がある．我々は追跡技術の一種であるコード計装を用いて，async/await の実行順序の理解を支援する可視化ツール AwaitViz を提案実装した．（ソフトウェア追跡技術関連）
・QR コードは画像による情報メディアであるが SNS などへのアップロード時に画像圧縮を受け読み取りが困難にある．この問題を解決した我々の新しいカラー画像コード RIC を用いて，動画配信の新しいビジネスモデルを提案し，実装方法を示した．（ソフトウェア追跡技術応用関連）
・JavaScript などのイベント駆動型のプログラミング言語に対して，2 つの類似コード間での実行結果の違いを解析する差分解析の研究を行った．本研究では(1)共有変数への read/write の動的情報を用いて，コードカバレッジを増加させるイベント列の生成する手法，(2) コード差分と条件分岐との距離を用いて，コード実行をコード差分に誘導する手法，を確率的に切り替えることで，従来の手法よりも効率よく差分を検出することに成功した．探索空間を絞り込むためにテイント解析を用いており，その点で追跡子とソフトウェア解析を組み合わせた研究成果となっている．
・並行プログラムがクラッシュした原因を自動解析することは重要だが，通常はコアダンプなどの限定的な情報しかないため困難な作業である．本研究では中間言語レベル（LLVM IR）で逆実行を行うことで，クラッシュ原因を自動解析する支援技術を提案した．マシン語レベルの情報を中間言語レベルに変換（リフト）する技術，メモリ位置を抽象的に扱うことで，既存手法（REPT）よりも解析精度が平均 40%向上した．マシン語と中間言語との対応関係を追跡する技術を含んでおり，これも追跡子とソフトウェア解析を組み合わせた研究成果である．
・ソースコードからプログラマの意図を抽出し，その変数名の意味的な一貫性をチェックする新しい手法を提案し実装した．このシステムはソースコードだけからその役割に基づいて変数に対する，そのプロジェクト独自の命名規則を学習する．この手法を 12 のオープンソースに適用し，その結果を複数の人間が評価したところ，1080 個の変数名のうち，39%の 416 個で別のより良い変数名を提案できた．この変数名を修正するパッチをもとの開発者に送ったところ 3 つのプロジェクトで採用された．
・プログラマが原始データ型を使ってパス名や座標などの高レベルな概念をどのように表現するかの調査を行うために，API の呼び出しを調べることにより，ある既定義の概念に対する表現を正確に識別する新しい方法を提案した．具体的には Java 標準 API で使われる 12 の概念的な型を定義し，26 のオープンソースプロジェクトからそれぞれの概念型に対する表現を抽出した．得られた表現に基づき，決定木分類器の訓練を行ったところ，83%の F スコアで，正しく概念型を予測できた．この結果は十分な例が与えられれば，ソースコードから
・Swift 言語の ARC 機能により発生する強い循環参照やメモリリークを自動的に検知する新しいツール UCDetector を提案した．Swift 言語の「静的型付けで安全な言語でありながら低レベルなプログラミングが可能」という特徴，Swift リフレクション API，デバッガ lldb Pyton API を用いることで，簡易かつコンパクトな実装が可能だったこと，その際に自明ではない様々な障壁があったことという知見を得た．また，実装した循環参照検知器の精度と効率に対する予備評価の結果も報告した．

５．主な発表論文等

〔雑誌論文〕　計1件（うち査読付論文　1件／うち国際共著　0件／うちオープンアクセス　0件）

１．著者名 権藤 克彦、新山 祐介、荒堀 喜貴	４．巻 39
２．論文標題 UCDetector：ユーザ空間で実装したSwift言語用の循環参照検知器	５．発行年 2022年
３．雑誌名 コンピュータ ソフトウェア	６．最初と最後の頁 4_97～4_128
掲載論文のDOI（デジタルオブジェクト識別子） 10.11309/jssst.39.4_97	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

〔学会発表〕　計15件（うち招待講演　0件／うち国際学会　7件）

１．発表者名 Yusuke Shinyama, Yoshitaka Arahori, Katsuhiko Gondow
２．発表標題 Improving Semantic Consistency of Variable Names with Use-Flow Graph Analysis
３．学会等名 28th Asia-Pacific Software Engineering Conference (APSEC), Dec. 2021, pp. 223-232, https://doi.org/10.1109/APSEC53868.2021.00030（国際学会）
４．発表年 2021年

１．発表者名 Yusuke Shinyama, Yoshitaka Arahori, Katsuhiko Gondow
２．発表標題 How Do Programmers Express High-Level Concepts using Primitive Data Types?
３．学会等名 28th Asia-Pacific Software Engineering Conference (APSEC), Dec. 2021, pp. 360-368, https://doi.org/10.1109/APSEC53868.2021.00043（国際学会）
４．発表年 2021年

１．発表者名 E. Tominaga, Y. Arahori, and K. Gondow
２．発表標題 DiverJS: Path Exploration Heuristic for Difference Analysis of Event-Driven Code
３．学会等名 SAC '21: Proc. 36th Annual ACM Sympo. on Applied Computing, March 2021, pp. 1768-1777（国際学会）
４．発表年 2020年～2021年

１．発表者名
S. Hoshino, Y. Arahori, and K. Gondow

２．発表標題
STRAB: state recovery using reverse execution at IR level for concurrent programs

３．学会等名
SAC '21: Proc. 36th Annual ACM Sympo. on Applied Computing, March 2021, pp. 1532-1541（国際学会）

４．発表年
2020年～2021年

１．発表者名
Y. Jitsunari, Y. Arahori, K. Gondow

２．発表標題
Quantifying the Limitations of Learning-Assisted Grammar-Based Fuzzing

３．学会等名
Int. Conf. on Advanced Information Networking and Applications (AINA 2019) pp 470-484（国際学会）

４．発表年
2019年

１．発表者名
E. Tominaga, Y. Arahori, K. Gondow

２．発表標題
AwaitViz: a Visualizer of JavaScript's Async/Await Execution Order

３．学会等名
SAC'19: Proc. 34th ACM/SIGAPP Sympo. on Applied Computing, pp. 2515-2524（国際学会）

４．発表年
2019年

１．発表者名
M. Kim, K. Lee, K. Gondow

２．発表標題
Implementation of Image SuperDistribution System

３．学会等名
3rd Int. Conf. on E-Business and Internet (ICEBI 2019), 6 pages（国際学会）

４．発表年
2019年

１．発表者名
石山泰地、荒堀喜貴、権藤克彦

２．発表標題
分散並行ファジング

３．学会等名
第26回ソフトウェア工学の基礎ワークショップ(FOSE2019)，ポスター発表

４．発表年
2019年

１．発表者名
和田智優、荒堀喜貴、権藤克彦

２．発表標題
ωCatch: クラウドシステムにおけるパフォーマンスバグの正確な自動検知に向けて

３．学会等名
第26回ソフトウェア工学の基礎ワークショップ(FOSE2019)，ポスター発表

４．発表年
2019年

１．発表者名
星野シンジ、荒堀喜貴、権藤克彦

２．発表標題
並行バグの効率的な自動原因解析を可能にする静的解析

３．学会等名
第26回ソフトウェア工学の基礎ワークショップ(FOSE2019)，ポスター発表（ライブ論文ポスター賞）

４．発表年
2019年

１．発表者名
冨永江奈、荒堀喜貴、権藤克彦

２．発表標題
強化学習によるイベント駆動コードの等価性検査戦略の生成

３．学会等名
第26回ソフトウェア工学の基礎ワークショップ(FOSE2019)，ポスター発表

４．発表年
2019年

１．発表者名
李兆亮、荒堀喜貴、権藤克彦

２．発表標題
強化学習に基づく並行バグ検知

３．学会等名
第26回ソフトウェア工学の基礎ワークショップ(FOSE2019)，ポスター発表

４．発表年
2019年

１．発表者名
和田智優，荒堀喜貴，権藤克彦

２．発表標題
クラウドシステムの非決定的性能バグ検査器

３．学会等名
第12回データ工学と情報マネジメントに関するフォーラム (DEIM 2020)

４．発表年
2019年

１．発表者名
冨永江奈、荒堀喜貴、権藤克彦

２．発表標題
イベント駆動コードの差分解析を可能にするパス探査経験則

３．学会等名
日本ソフトウェア科学会 第21回プログラミングおよびプログラミング言語ワークショップ（PPL2019）

４．発表年
2019年

１．発表者名
春日涼太郎 、荒堀喜貴、権藤克彦

２．発表標題
Typestate 解析を応用した静的解析に よる 分散並行システムのバグの検出

３．学会等名
日本ソフトウェア科学会 第21回プログラミングおよびプログラミング言語ワークショップ，ポスター（PPL2019）

４．発表年
2019年

〔図書〕　計0件

〔産業財産権〕

〔その他〕

―

６．研究組織

	氏名 （ローマ字氏名） （研究者番号）	所属研究機関・部局・職 （機関番号）	備考

７．科研費を使用して開催した国際研究集会

〔国際研究集会〕　計0件

８．本研究に関連して実施した国際共同研究の実施状況

共同研究相手国	相手方研究機関