公開

（事後評価・研究者用）

研　究　報　告　書

「プログラムコードの整合性ある自動抽象化による表明強化の支援」

研究期間：2019 年 4 月～2021 年 3 月
研究者番号：50223
研 究 者：小林　努

1．研究のねらい

現代のソフトウェアの品質保証における最大の難しさの 1 つはその複雑さにあり、その本質は、対象ソフトウェアの動作する環境や利用の際のルールといったドメインの複雑さにある。
ここで、特にドメインがどのような性質を持つか（例えば、常に成り立つべき「不変条件」など）が品質保証に重要である。
例えば、Java や C#などの言語でサポートされている Design by Contract (DbC)というアプローチでは、開発者がプログラムコードの中に不変条件などの性質の形式的記述を書き、それがテストコードの生成やコードの解析に利用される。
このような明示的に性質を扱うアプローチは高品質なソフトウェアの開発に有効とされる一方、現実には十分に活用されているとは言えない。これは、性質を記述するのに労力がかかることに加え、開発者が複雑な対象の性質を把握することが困難なためである。特に、開発初期段階の分析で対象の性質を把握する作業には限界があり、開発時には開発者に認識されない暗黙的な性質も多く存在することになる。このような暗黙的な性質は品質保証の対象とならず、開発後期やリリース後の大きな問題につながる。

そこで本研究では、与えられたプログラムコードをもとに、ドメインに関する性質（特に不変条件）を獲得する手法を提案した。
ここで特に、獲得する不変条件は (1) 多くのバグ・難しいバグを排除できるような強い不変条件 (2) プログラムの式の一部を大量に連ねたような条件でなく、ドメイン上の意味を簡潔に表すような不変条件 となることを目指した。
本提案の中心となったのは、テストコードの性能（どれだけ多くの欠陥・どれだけ難しい欠陥を発見できるか）を解析するための変異解析という技法の応用である。変異解析では、テスト対象のコードに人工的な欠陥を埋め込み、テストコードがその埋め込んだ欠陥を検出できるか否かを分析する。
本提案の手法では、プログラムコードと対応付いた形式仕様に対し、変異解析と同様に人工的な欠陥を埋め込み、不変条件がその欠陥を検出できるか否か分析することを目指した。さらに、その結果をもとに不変条件を強化することを通じ、不変条件を獲得することを目指した。

2．研究成果

（1）概要

本研究では、プログラムコードにおけるドメインのロジックと対応付いた形式仕様に対し、多様な人工的欠陥を埋め込み、不変条件が埋め込んだ欠陥を検出できるかを分析し、検出で

きなかった欠陥を検出できるように不変条件を強化することを目指した。
この目標を達成するにあたり、主に以下の 3 つの研究テーマに取り組んだ。
テーマA：形式仕様に対する人工的欠陥の埋め込みおよびその検出チェック
テーマB：検出結果に基づく不変条件の強化
テーマC：プログラムコードにおけるドメインのロジックと形式仕様の対応付け
本研究の提案手法は特定のプログラミング言語・仕様記述言語に限ったものではないが、具体的に研究を進めるにあたってはプログラミング言語として Java を、仕様記述言語として Event-B を採用した。
研究の初期段階で行った feasibility study の結果、特にテーマA・Bについて学術上・実用上意義深いものであると考えられたため、特にこれらに注力を行った。
結果として、テーマA・Bについては手法を確立し、Event-B の開発環境 Rodin Platform のプラグインとして手法の実装を行った。またテーマCについても方法論と典型的な対応付けについて整備を進めることができた。

（2）詳細

研究テーマA：形式仕様に対する人工的欠陥の埋め込みおよびその検出チェック
研究テーマB：検出結果に基づく不変条件の強化

テーマAでは、ソフトウェアテストの技法である変異解析を応用し、与えられた振舞いの形式仕様 b に対して人工的に欠陥を埋め込んだ版(mutant)b'を構築し、与えられた不変条件 i が b'を検出できるかどうかチェックする手法の構築を目指した。
テーマBでは、b'を検出できるように不変条件 i を強化する手法の構築を目指した。
テーマAの手法はテーマBの手法と密接に関連しているため、ここでは両テーマを同時に説明する。

ここではまず、埋め込む欠陥とは何か、そして検出するとはどういうことかが重要となる。
不変条件は、仕様 b の振舞いが到達可能な全ての状態が満たす条件であるため、到達可能な状態と不変条件を満たす状態との関係は図 1-a のようになる。
本研究の目的は、図 1-a における到達可能な状態がどのようなものかの把握が難しいので、把握が容易な不変条件を強化することで到達可能な状態をより良く「説明」することである。それはすなわち到達可能な状態が不変条件を満たす範囲内で、不変条件を狭めていくことに対応する(図 1-b)。

図 1-a

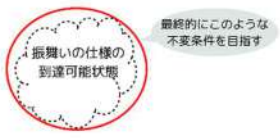

図 1-b

従って、埋め込む欠陥としては、対象システムの振舞いの遷移を広げるような欠陥、すなわ

ちイベントの発火条件の弱化および状態遷移の非決定性の増大とした。
また、不変条件 i が振舞いの仕様 b'に埋め込まれた欠陥を検出するとは、b'の到達可能な状態のうち i を満たさないものが存在することを検出することとした(図 1-c)。

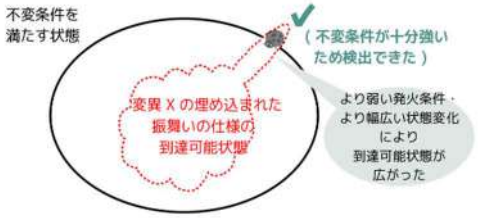

図 1-c

この検出チェックは、i を満たす状態から b'で定義された振舞いにより i を満たさない状態に遷移するかどうか、すなわち
∀ s . i(s) ∧ b の発火条件(s) ∧ b の状態遷移(s,s') ⇒ i(s')
を検証することで行うことができる。本提案では、この式をもとにした式の充足可能性・不能性の検査問題として検出チェックを定式化した。
このような定式化により、欠陥を埋め込んで生成された mutant が相変わらず不変条件を満たす（不変条件が mutant を検出できなかった）場合(図 1-d)に、埋め込んだ欠陥を検出することができるように不変条件を強化する(図 1-e)ということが体系的に行えるようになった。

図 1-d

図 1-e

ソフトウェアテスト分野で従来行われてきた変異解析の研究でプログラムコードに埋め込まれる欠陥は、文法エラーを起こさない範囲のランダムなものが基本であった。
一方で、形式仕様の式には論理式の形を取るものが多く、本研究が対象とした手法 Event-B では一階述語論理と集合論をベースにした言語が採用されている。
このような式に対しては、上記に説明した、「発火条件の弱化」「状態遷移の満たす条件の弱化」「不変条件の強化」を（当てずっぽうに試すのではなく）確実に行うことが可能である。
例えば、「P∨Q」を強化するには、「false」「P」「Q」「P∧Q」「(P を強化した式)∨Q」などに式を変更すれば良い。
また、「S⊆T」を弱化するには、「true」「(S の部分集合)⊆T」「S⊆(T の上位集合)」などに式を変更すれば良い。
本研究では、Event-B の言語で定義されている全ての式と述語の形式に対して、上記のような強化ルール・弱化ルール・部分集合構成ルール・上位集合構成ルールを定義した。

また、本アプローチは、多様な欠陥を埋め込み、それを検出する不変条件を構成するもので

あるため、その本質は探索である。
探索を効率化するため、式の強化であればまず大きく強化した上で徐々に弱化する、また埋め込んだ変異に関係のある強化を優先する、簡潔な式になる強化を優先するなどの各種ヒューリスティクスを提案している。
また、弱化・強化の典型的な場合のパターン化も行った。
さらに、過学習のような状態を避けるためのヒューリスティクスや対話的手法も提案した。
その上、本研究の前段階である ACT-I 期間の研究（整合性を保持する形式仕様の自動抽象化システム「ソフトウェア顕微鏡」の開発）において提案した仕様の抽象化手法を用いることにより、元の仕様の一部の側面のみを反映した抽象版の仕様を構築し、それに対して変異解析と条件強化を行うことでより効率化を行える見込みがあることを確認した。

このような変異解析をベースにした強化手法により、発生しがちな欠陥をあらかじめカバーするような不変条件を獲得することができる。
上記の式の変異ルールと変異解析手法・強化手法は Event-B の開発環境 Rodin platform のプラグインとして実装している。
これらの研究成果は、研究期間終了時現在、論文としてまとめ上げつつある。

研究テーマC：プログラムコードにおけるドメインのロジックと形式仕様の対応付け

テーマCでは、プログラムコードにおけるドメインのロジックをもとに、その本質を抽出した形式仕様を構築することを試みた。
プログラムコードにおいてドメインのロジックを扱う研究はオブジェクト指向開発の分野で長年行われてきたが、その中でもドメイン駆動設計(DDD)と呼ばれる設計パラダイムが、特に開発の実務に関わるソフトウェアエンジニアの間で近年注目を集めている。
本研究では、この、実用的かつドメインのロジックが切り分けられるような DDD 手法で開発されたプログラムコードのドメインロジックに対応する部分を形式仕様として切り出すことを試みた。
具体的には、DDD では主にドメインの性質を陽に扱うためのソフトウェア設計パターンが提案されているが、それらの分析を通じ、特にドメインサービス・アプリケーションサービスと呼ばれるパターンを中心として形式仕様の形に落とし込む方法論を模索した。
研究期間終了時現在では一部の典型的な場合の対応付けを提案しており、今後より多くの場合の対応付けを提案していく予定である。
また、DDD で設計されたオープンソースのソフトウェアをもとに形式仕様を構築し、その上でテーマA・Bの手法を適用する実験を行っている。

3. 今後の展開

本研究では DbC の導入を促進しソフトウェアの信頼性を向上するという未来ビジョンを掲げてきた。
研究期間終了時における成果から、この目標に対し一定の基礎固めと一連の手法の提案ができたと考えられる。
学術への波及効果を考えると、特にテーマA・Bで提案した手法は、ソフトウェアテストの近年

の最大の関心の一つである変異解析について、論理式の性質を利用しより意味のある解析を行うというものであり、様々な発展の余地が非常に大きいと見込まれる。
例えば、変異解析では多種の変異が生成され状態爆発問題が起こるが、上記で言及したように、仕様の抽象化を用いることで重要な側面に注力しつつこれを軽減できることが見込まれた。これらの方向に向け技術的な発展を今後も進めていく予定である。
また、産業や社会への波及効果を考えると、特にテーマCで提案した手法の完成度を高め、近年エンジニアに注目されるドメインの性質を明示的に扱った開発を促進する、実用的かつソフトウェアの複雑さの本質に切り込む手法として発展させていく予定である。

4. 自己評価

研究終了段階において、挑戦的な問題に対応する手法について、その基礎を固め、具体的な手法を提案し、豊かな発展の方向性を示すことができたと考えられる。
一方、研究テーマCの進捗や研究内容の発表については当初の予定より遅れており、この点は計画時に問題の大きさを過小評価していたという点で改善の余地があったと考えている。この点は今後の糧としていきたい。
本研究の技術的・学術的な側面は、「ソフトウェアの検査基準自体の検査」という重要だが難しく注目されている問題に対し、ソフトウェアテストの基盤技術に式の意味の考慮を加えるというアプローチで切り込む独創的なものであったと考えている。
ソフトウェア工学では近年自動修正技術が産学界の両者から注目されているが、その一部を発展させることができたとも考えている。
また、社会的側面として、DDD などのソフトウェアのドメインに集中する設計手法は近年産業界からの注目が集まっており、それに沿った形で信頼性を向上するという社会的要請への対応に貢献する研究を進められたと考えている。

5. 主な研究成果リスト

(1)論文(原著論文)発表

(なし)

(2)特許出願

なし

(3)その他の成果(主要な学会発表、受賞、著作物、プレスリリース等)

なし