不具合発生時に類似する報告に対する修正コード片を提示する手法

2020SE020 金本知泰
指導教員：名倉正剛

1　はじめに

ソフトウェアに不具合があった場合，原因特定，デバッグ，テスト，再デプロイの順で不具合に対する修正が行われる．修正の実施にあたっては，特定した原因事象を発生させたコード上の箇所で利用する API の利用手順に関する知識や，あるいはその箇所で利用するプログラム構文に対する知識など，多くの知識を要する．したがって原因箇所を特定できたとしても修正方法の検討は容易ではない．修正方法が分からない時にはさまざまな情報源から情報収集を行いながら検討するが，Web を利用して情報収集を行うことも多い．Web 経由でソースコード管理を行う仕組みとして GitHub 等のソースコードリポジトリがある．Web から情報収集する際にそのようなリポジトリを対象に不具合報告と修正コード片を収集すれば，不具合修正の参考になる情報を得られる可能性がある．

2　GitHub での不具合解決の流れ

GitHub では不具合報告と変更履歴が管理されている．GitHub の Issue tracker とは，不具合報告を管理する機能のことである．開発者は Issue tracker によって，発生した不具合を登録する．そして，その不具合に対して，修正担当者を割り当て，割り当てられた担当者は修正を実施する．修正が実施されると修正内容がレビューされ，問題がなければ Issue tracker に解決済みとして登録する [1].

3　本研究で対象にする課題

GitHub 等のソースコードリポジトリを対象に，似たような不具合報告を収集することで，不具合修正の参考になる情報が得られることがある．しかし，2 つの課題がある．

課題 1 不具合報告で報告されている不具合に対して実施した変更を示す変更差分の修正コード片が対応づいていない

Issue tracker の不具合報告に似た報告が存在した場合，GitHub のリポジトリを参照することで，似たような不具合報告に対応する変更差分を取得し理解することができる．しかし，GitHub のようなソースコードリポジトリでは，不具合報告で報告されている不具合に対して実施した変更した変更を示す変更差分の修正コードが対応づけられていない．

課題 2 発生した同じような不具合に対して，情報収集をする開発者が同様の文章で検索できるとは限らない

発生した不具合に対する類似した事例を探すためには，情報収集をする開発者が，類似した事例の文章と同様の文章で検索する必要がある．しかし，情報収集をする開発者が同様の文章で検索できるとは限らない．

したがって，Issue tracker の不具合報告から，似たような報告を検出してそれを参考にすることは容易ではない．

4　提案手法

4.1　概要

本研究では，ソフトウェアに不具合が発生した際に，過去の類似した不具合に対する修正方法から修正方法を提示する手法を提案する．具体的には，GitHub の Issue tracker 機能を利用して不具合が報告される際に，類似した不具合報告を検索し，それに対応する修正コード片を提示する．

4.2　処理の流れ

提案手法の構成と処理の流れを図 1 に示す．開発者は，発生した不具合の不具合報告を入力する．

前処理　不具合報告と修正コード片の対応づけ

GitHub から，解決済み Issue を解析して，不具合報告とそれに対応する変更差分の修正コード片を抽出して，その関連を DB に保存する．

手順 1　類似不具合報告の検索

対象の不具合報告に，類似する不具合報告を検索し，類似不具合報告に対する修正コード片を取得する．

手順 2　類似修正コード片の検索

手順 1 で取得した複数の類似不具合報告に対する修正コード片から，修正前と修正後のブロックでペアを作成する．作成したペアに類似する修正コード片を検索し，各修正コード片に対応する不具合報告を取得する．

そして，手順 1 で検索した類似不具合報告とそれに対する修正コード片，手順 2 で検索した類似修正コード片とそれに対応する不具合報告を合わせて提示する．

4.3　前処理：不具合報告と修正コード片の対応づけ

GitHub から解決済み Issue を解析して，不具合報告とそれに対応する変更差分の修正コード片を抽出して，その関連を DB に保存する．この手順ではまず，Issue tracker に解決したコミット ID が記録されている不具合報告を抽出する．そしてそのコミット ID で示されるコミットと，不具合報告が作成される直前のコミットの差分を修正コード片として抽出する．なお，不具合に対する修正の際には，対象の不具合を修正するためのブランチを作成して，そのブランチに対して修正を実施し，修正が完了してその動作の確認やレビューによって修正が適切であることを確認したのちに，元のブランチにマージするような開発を実施することが多くある．そのような場合には Issue tracker にはブランチをマージしたコミット ID が記録される．Issue

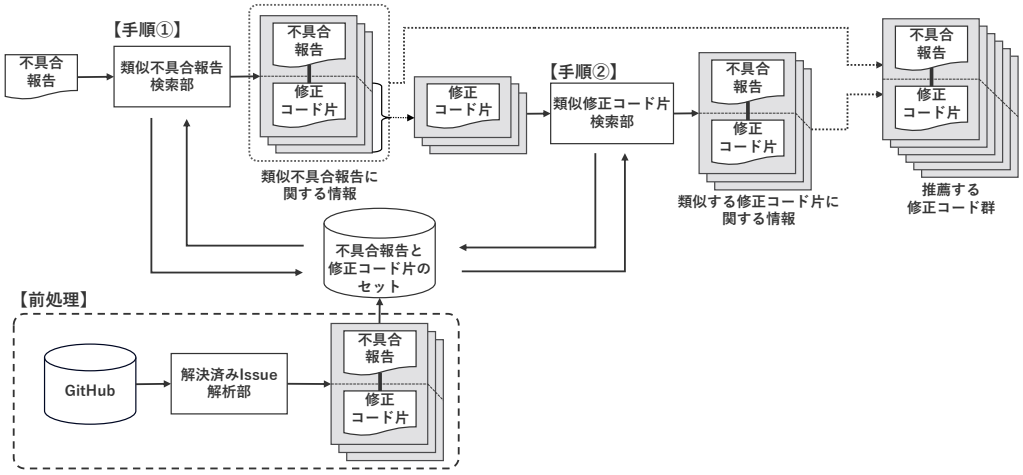

図 1　提案手法の構成と処理の流れ

tracker から取得したコミット ID がブランチをマージするコミット ID だった場合は，そのコミット ID で示されるコミットと，該当のブランチで作成したコミットの差分を，修正コード片として抽出する．

4.4　手順 1：類似不具合報告の検索

開発者が入力した不具合報告に対して，類似する不具合報告を全文検索により検索する．全文検索は，転置インデックス方式によって実施する．この方式では，事前に文書を解析して単語に切り分け，単語ごとにそれが含まれている文書のインデックスを記録した索引構造（転置インデックス）を利用して検索を行う．転置インデックスの作成の際には，形態素解析によって文章から単語を抽出する．この全文検索には，Apache Solr を利用する．前処理により作成した DB と，この検索により，課題 1 を解決する．

4.5　手順 2：類似修正コード片の検索

不具合報告の内容は一致していないが，修正内容が類似する場合がある．その場合は文章が異なる単語を利用しているため，手順 1 では類似したものとして判定できない．そこで，手順 2 では不具合報告文書によらず，手順 1 で検出したコード片に類似する修正コード片を検出する．

まず，手順 1 で検出した修正コード片を利用して，修正範囲のブロックを特定する．そして特定した修正範囲のブロックに対して，修正前と修正後のブロックでペアを作成する．前処理で作成した DB 内の修正コード片に対して同様に修正前後のブロックでペアを作成し，手順 1 で検出された修正コード片を利用して作成されたブロックのペアに類似するブロックのペアを特定する．特定したブロックのペアに対応する修正コード片を利用して DB から不具合報告を検索する．ブロックの類似性の判定方法は，ブロックから呼び出すメソッドの集合の類似性を用いる．類似修正コード片の検索により，課題 2 を解決する．

5　評価実験

GitHub の 141 個のプロジェクトから収集できた 2796 個の不具合報告と修正コード片のセットを DB に保存し，その DB のデータセットを使って評価した．DB の中にある 10 個の不具合報告を入力し，入力した不具合報告に関連する修正コード片が手順 1 の類似不具合報告の検索と手順 2 の類似修正コード片の検索によって提示されるか確認した．入力した 10 個の不具合報告のうち，1 個の不具合報告について観察した結果，手順 1 の類似不具合報告の検索で入力した不具合報告に関連する修正コード片を提示できることを確認できた．また，手順 2 の類似修正コード片の検索でも入力した不具合報告に関連する修正コード片を提示できることを確認できた．これらより，提案手法によって入力した不具合報告に関連する修正コード片を提示できる可能性があることを確認した．

6　おわりに

ソフトウェアに不具合が発生した際に，過去の類似した不具合に対する修正方法から修正方法を提示する手法を提案した．前処理と手順 1 の類似不具合報告の検索と手順 2 の類似修正コード片の検索の実装を行い，141 個のプロジェクトから収集できた 2796 個の不具合報告と修正コード片のセットを DB に保存し，その DB の中にある不具合報告を入力し，提案手法によって入力した不具合報告に関連する修正コード片を提示されるか確認した．評価の結果，提案手法が入力した不具合報告に関連する修正コード片を提示できる可能性があることを確認した．

参考文献

[1] Eirini Kalliamvakou, Daniela Damian, Kelly Blincoe, Leif Singer, and Daniel M. German. Open Source-Style Collaborative Development Practices in Commercial Projects Using GitHub. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Vol. 1, pp. 574–585, 2015.