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

平成　２９　年　　６　月　　１　日現在

機関番号：　１３９０１
研究種目：　若手研究(B)
研究期間：　2014～2016
課題番号：　２６７３００３６

研究課題名（和文）トピック抽出を用いたコードクローン検出手法

研究課題名（英文）A code clone detection approach based on topic extraction

研究代表者
吉田　則裕（Yoshida, Norihiro）

名古屋大学・情報学研究科・准教授

研究者番号：００５８２５４５

交付決定額（研究期間全体）：（直接経費）　　1,800,000円

研究成果の概要（和文）：識別子名の類似性に基づくコードクローン検出ツールを開発し，有効性の評価を行った．
本ツールは，単純に同一の識別子名を含むコード片をコードクローンとして検出するだけでなく，類似した用いられ方をしている識別子名を含むコード片もコードクローンとして検出する．コード片が類似した用いられ方をしている識別子名を含むかどうかを判定するためには，自然言語処理の分野で提案されているトピック抽出技術を用いた．

研究成果の概要（英文）：This research aims at developing a code clone detection tool based on the similarity of identifier names and evaluating its usefulness. This tool is able to detect code clones that not only includes the same identifier names but also identifiers including similar usage contexts. This tool is developed using a topic extraction technique for natural language processing in order to determine whether a pair of code fragments includes identifiers that share similar usage context.

研究分野：　ソフトウェア工学

キーワード：　コードクローン

１．研究開始当初の背景

(1) ソフトウェア開発にかかるコストを増大させている要因の１つとして,ソースコード中のコードクローンが挙げられる．コードクローンとは，ソースコードの一部分（コード片）のうち，他のコード片と類似したものを指し，コピーアンドペースト等により作成される．あるコード片を修正するとその全てのコードクローンを見つけ出し,修正を行う必要が生じることがある．特に，ソースコード中に欠陥が見つかった場合には，その欠陥を含むコード片のコードクローンを探し，検査する必要がある[1]．しかし．ソースコード中のコードクローンを人手で探すためには大きな労力が必要となる．特に，数千万行からなる大規模ソースコードが対象の場合，全てのコードクローンを人手で探すことはより困難となる．

(2) そこで,数多くの研究者がソースコード中のコードクローンを自動的に検出するツールの開発　を行ってきた[2][3][4]．これらコードクローン検出ツールは，ソースコードをツール毎に決められているプログラム表現(例えば，構文木)に変換し，そのプログラム表現上において，等価な部分をコードクローンとして検出する[3]．

(3) これらコードクローン検出ツールに共通していることは，コード片に含まれる識別子名（変数名など）が類似していても，ツール毎のプログラム表現において,差異が大きい場合はコードクローンとして検出しないことである．しかし,類似した識別子名を含むコード片は，類似した機能を持つことが多いため，同時にバグ修正や機能追加を行うことが多い[5].そのため，類似した識別子名を含むコード片をコードクローンとして検出すべきであると考えられる．

＜引用文献＞

[1] M. Datar, N. Immorlica, P. Indyk, V. S. Mirrokni: Locality-Sensitive Hashing Scheme Based on P-stable Distributions, Proc. of SoCG 2004, pp. 253-262, 2004.
[2] CCFinderX: http://www.ccfinder.net.
[3] L. Jiang, G. Misherghi, Z. Su.: DECKARD: Scalable and Accurate Tree-Based Detection of Code Clones, Proc. of ICSE 2007, pp.96-105, 2007.
[4] CloneDR: http://www.semdesigns.com/Products/Clone/index.html?Home=DMSToolkit

２．研究の目的

(1) 本研究の目的は，識別子名の類似性に基づくコードクローン検出ツールを開発し，有効性の評価を行うことである．

(2) 本ツールは，単純に同一の識別子名を含むコード片をコードクローンとして検出するだけでは なく，類似した用いられ方をしている識別子名を含むコード片もコードクローンとして検出する．その理由は，類似した用いられ方をする識別子であっても，開発者によって異なる名前が付けられることがあるため，類似した用いられ方をしている識別子名を等価であると見なした方が，類 似した機能を持つコード片をコードクローンとして検出しやすいと考えられるからである.コード片が類似した用いられ方をしている識別子名を含むかどうかを判定するためには，自然言語処理の分野で提案されているトピック抽出技術を用いる．具体的には，2 つのコード片間で共通するトピックが占める割合が大きいなら，それらコード片をコードクローンと判定する．

３．研究の方法

(1) ソースコードに含まれるトピックを抽出する．各ソースファイルに含まれる語(基本的には識別子名.ただし,複数の語からなる識別子名は各語に分割)を抽出し,トピックの特定を行う．

(2) コード片間の等価性判定を行う．コード片間の等価性を判定するため，構文木中の部分木に対して特徴ベクトルを付加する．特徴ベクトルは，部分木に含まれるトピックの分布を表す．次に，特徴ベクトル間の距離に基づいて部分木のクラスタリングを行う．最後に，同一クラスタに属した部分木に対応するコード片を等価なコード片として提示する．各部分木をベクトルで表現する利点は，部分木間の距離を求める問題をベクトル間の距離を求める問題に変換することで，計算量を低下させることができることである．特徴ベクトルの数が膨大になると,クラスタリングにかかる計算時間が大きくなるため，LSH (Locality Sensitive Hashing)[5]というハッシュ関数を用いたクラスタリングを行う．LSH は,「類似したベクトルは高い可能性で同じハッシュ値になり，異なるベクトルは高い可能性で異なるハッシュ値になる」という性質を持つハッシュ関数である．

＜引用文献＞
[5] M. Datar, N. Immorlica, P. Indyk, V. S. Mirrokni: Locality-Sensitive Hashing Scheme Based on P-stable Distributions, Proc. of SoCG 2004, pp. 253-262, 2004.

４．研究成果

(1) 提案手法の実装を行い，「本ツールがコードクローンとして検出したコード片が,既存のコードクローン検出ツールで検出されるか」を評価した.入手可能であり,かつ代表的なコードクロー ン検出ツールである CCFinderX[6] や DECKARD[7], CloneDR[8]との比較を行った.また,「既存のコードクローン検出ツールと比較して,本ツールのスケーラビリティ(検出時間やメモリ消費量)は高いか」についても評価を行った.

(2) (1)で示した観点から評価を行ったところ，他のツールと比較して有効に機能するパラメータを特定することができた.また，スケーラビリティについても他のツールと比べて十分に高いことがわかった.

(3) 企業に所属する実務者の協力を得て，「検出したクローンセットがその後の保守作業において同時に修正される，もしくは同一の欠陥を含んでいるか」を評価してもらった．その結果，ソフトウェアの保守作業を十分に支援できる性能を有しているという意見が得られた.

(4) 既存のコードクローン検出手法のほとんどはコンパイラで使用されている字句解析や構文解析に基づくものである．よって，トピック抽出に基づく本手法は，新しい分類のコードクローン検出手法と言える．ここ数年，コードクローン検出に関する研究は既存手法の改良が主になってきており，新しい分類の検出手法と言える本手法の改良を続けることで，国内外に大きいインパクトを与える研究になりうると考える.

＜引用文献＞
[6] CCFinderX: http://www.ccfinder.net.
[7] L. Jiang, G. Misherghi, Z. Su.: DECKARD: Scalable and Accurate Tree-Based Detection of Code Clones, Proc. of ICSE 2007, pp.96-105, 2007.
[8] CloneDR: http://www.semdesigns.com/Products/Clone/index.html?Home=DMSToolkit

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計 6 件）

① 中山 直輝，吉田 則裕，藤原 賢二，飯田 元，高田 光隆，高田 広章: "コードクローンとの位置関係に基づく欠陥混入傾向の調査"，情報処理学会論文誌，Vol.57，No.2，pp.681-693，2016 年（査読有）．http://id.nii.ac.jp/1001/00148176/

② 後藤 祥，吉田 則裕，藤原 賢二，崔 恩瀞，井上 克郎: "機械学習を用いたメソッド抽出リファクタリングの推薦手法"，情報処理学会論文誌，Vol.56，No.2，pp.627-636，2015 年（査読有）．http://id.nii.ac.jp/1001/00113143/

③ 戸田 航史，亀井 靖高，濱崎 一樹，吉田 則裕: "Chromium プロジェクトにおけるレビュー・パッチ開発経験がレビューに要する時間に与える影響の分析"，コンピュータソフトウェア，Vol.32，No.1，pp.227-234，2015 年（査読有）．http://doi.org/10.11309/jssst.32.1_227

④ 崔 恩瀞，藤原 賢二，吉田 則裕，林 晋平: "変更履歴解析に基づくリファクタリング検出技術の調査"，コンピュータソフトウェア，Vol.32，No.1，pp.47-59，2015 年（査読有）．http://doi.org/10.11309/jssst.32.1_47

⑤ 後藤 祥，吉田 則裕，藤原 賢二，崔 恩瀞，井上 克郎: "メソッド抽出リファクタリングが行われるメソッドの特徴調査"，コンピュータソフトウェア，Vol.31，No.3，pp.318-324，2014 年（査読有）．http://doi.org/10.11309/jssst.31.3_318

⑥ 山中 裕樹，崔 恩瀞，吉田 則裕，井上 克郎: "情報検索技術に基づく高速な関数クローン検出"，情報処理学会論文誌，Vol.55，No.10，pp.2245-2255，2014 年（査読有）．http://id.nii.ac.jp/1001/00106364/

〔学会発表〕(計 16 件)

① Seiya Numata, Norihiro Yoshida, Eunjong Choi, Katsuro Inoue: "On the Effectiveness of Vector-based Approach for Supporting Simultaneous Editing of Software Clones", 17th International Conference on Product-Focused Software Process Improvement (PROFES 2016), Trondheim, Norway, 2016.

② Norihiro Yoshida: "When, why and for whom do practitioners detect technical debts?: An experience report", 1st International Workshop on Technical Debt Analytics (TDA 2016), Hamilton, New Zealand, 2016.

③ Yuta Nakamura, Eunjong Choi, Norihiro Yoshida, Shusuke Haruna, Katsuro Inoue: "Towards Detection and Analysis of Interlanguage Clones for Multilingual Web Applications", 10th International Workshop on Software Clones (IWSC 2016), Suita, Osaka, Japan, 2016.

④ 上村 恭平, 吉田 則裕, 崔 恩瀞, 飯田 元, 曲 生国, 秋庭 真一: "ソースコードの削減可能量計測ツールの開発", 日本ソフトウェア科学会第 23 回ソフトウェア工学の基礎ワークショップ (FOSE 2016), 香川県仲多度郡琴平町, 2016 年.

⑤ 中村 勇太, 崔 恩瀞, 吉田 則裕, 春名 修介, 井上 克郎: "複数プログラミング言語で記述されたソフトウェアからのコードクローン検出", 情報処理学会第 190 回ソフトウェア工学研究会, 岐阜県岐阜市, 2016 年.

⑥ 沼田 聖也, 吉田 則裕, 崔 恩瀞, 井上 克郎: "欠陥の同時修正支援における関数クローン検出ツールの有効性調査", , 電子情報通信学会ソフトウェアサイエンス研究会, 北海道札幌市, 2016 年.

⑦ 石津 卓也, 吉田 則裕, 崔 恩瀞, 井上 克郎: "メタヒューリスティクスを用いた集約可能コードクローン量の推定", 情報処理学会第 193 回ソフトウェア工学研究会, 北海道札幌市, 2016 年.

⑧ 齋藤 雄輔, 藤原 賢二, 井垣 宏, 吉田 則裕, 飯田 元: "Pull Request 駆動型の開発を支援するツールの検討",電子情報通信学会ソフトウェアサイエンス研究会, 鳥取県東伯郡三朝町, 2015 年.

⑨ 佐野 真夢, 吉田 則裕, 春名 修介, 井上 克郎: "情報検索技術に基づく関数クローン検出を用いた変更管理システムの開発",情報処理学会第 190 回ソフトウェア工学研究会, 福岡県福岡市, 2015 年.

⑩ 中山 直輝, 吉田 則裕, 藤原 賢二, 飯田 元, 高田 光隆, 高田 広章: "コードクローンとの位置関係に基づく欠陥混入傾向の調査", ソフトウェアエンジニアリングシンポジウム 2015, 神奈川県横浜市, 2015 年.

⑪ 中山 直輝, 吉田 則裕, 藤原 賢二, 飯田 元: "開発履歴分析を用いたコードクローン内外における欠陥発生率の調査", 情報処理学会第 186 回ソフトウェア工学研究会, 大阪府吹田市, 2014 年.

⑫ 辻 健二, 崔 恩瀞, 吉田 則裕, 春名 修介, 井上 克郎: "コードクローン編集者数に着目した開発履歴の分析",情報処理学会第 186 回ソフトウェア工学研究会, 大阪府吹田市, 2014 年.

⑬ 槇原 絵里奈, 藤原 賢二, Putchong Uthayopas, Chantana Chantrapornchai, Jittat Fakcharoenphol, 井垣 宏, 吉田 則裕, 飯田 元: "日本とタイにおけるプログラミング初学者のプログラミング行動の比較", 電子情報通信学会教育工学研究会, 石川県金沢市, 2014 年.

⑭ 佐野 真夢, 崔 恩瀞, 山中 裕樹, 吉田 則裕, 井上 克郎: "識別子名のタグクラウドを用いたコードクローン理解支援ツールの開発", 情報処理学会第 184 回ソフトウェア工学研究会, 茨城県ひたちなか市, 2014 年.

⑮ Manamu Sano, Eunjong Choi, Norihiro Yoshida, Yuki Yamanaka, Katsuro Inoue: "Supporting Clone Analysis with Tag Cloud Visualization", International Workshop on Innovative Software Development Methodologies and Practices (InnoSWDev 2014), Hong Kong, China, 2014.

⑯ Tsubasa Saika, Eunjong Choi, Norihiro Yoshida, Akira Goto, Shusuke Haruna,

Katsuro Inoue: "What Kinds of Refactorings are Co-occurred? An Analysis of Eclipse Usage Datasets", 4th International Workshop on Empirical Software Engineering in Practice (IWESEP 2014), Suita, Osaka, Japan, 2014.

６．研究組織

(1)研究代表者

吉田　則裕（Yoshida, Norihiro）
名古屋大学・情報学研究科・准教授
研究者番号：00582545