コードクローン検出が用いる局所性鋭敏型ハッシュに与えるパラメータ決定手法

徳井 翔梧　吉田 則裕　崔 恩瀞　井上 克郎

コードクローンとは，ソースコード中に存在する互いに一致または類似した部分を持つコード片のことである．我々が提案したブロッククローン検出法では局所性鋭敏型ハッシュ(LSH) を用いることにより，従来の研究では困難であった意味的に類似するコードクローン対を検出できた．LSH とは，高次元なデータを確率的にハッシュ化し，近傍点を見つけるアルゴリズムである．しかし，LSH は検出漏れの可能性がある．本研究では，LSH の衝突確率とクローン検出の利用者が与えた類似度の閾値に対して一定の再現率を得るための，LSH に与えるパラメータ決定手法を提案する．更に，決定されたパラメータをブロッククローン検出法に適用し，異なる規模の 2 つのプロジェクトに対して実験を行うことで，本手法の有効性を示す．

A code clone is a code fragment that has identical or similar code fragments to it in the source code. Our previous research developed a block code clone detector using Locality Sensitive Hashing (LSH) that could detect not only syntactic clones but also semantic clones which had been difficult to be detected. LSH is a near neighbor search algorithm that performs probabilistic hashing of high-dimensional data. However, LSH-based detection is possible to miss several code clones. In this study, we propose an approach to determine LSH parameters in order to obtain the constant recall when a collision probability of LSH and a similarity threshold are given by a user. Finally, we apply this approach with the block code clone detector to two projects. The result shows the effectiveness of this approach.

1　まえがき

ソフトウェアの保守における問題のひとつとして，コードクローンが指摘されている [6]．コードクローンとは，ソースコード中に含まれる互いに一致または類似した部分を持つコード片のことであり，コードクローン対をクローンペアと呼ぶ．一般的に，コードクローンの存在はソフトウェアの保守を困難にすると言われている．コードクローンに対する様々な保守や管理の方法が提案されているが，ソースコードの規模が大きくなるとソースコード中に含まれるコードクローンも膨大な量となり，手作業でそれらを管理することは困難となる．そこで，コードクローンを自動的に検出するための手法が提案されている [10] [6]．

横井らは情報検索技術 [4] と局所性鋭敏型ハッシュ (LSH) [2] を利用することによって，意味的に処理が類似したコードブロック単位のコードクローンを検出するブロッククローン検出法を提案した [7]．LSH とは，高次元なベクトル集合を確率的にハッシュ化し，類似度が高いベクトル対を高速に見つけるアルゴリズムである．この検出法は，情報検索技術の一種である TF-IDF (Term Frequency-Inverse Document Frequency) 法 [4] を用いてコード片ごとに特徴ベクトルを生成し，局所性鋭敏型ハッシュ(LSH) の一種である FALCONN[†1] [2] を用いてコサイン類似度が閾値以上のベクトルペアの探索 (類似探索) を行い，コードクローンを検出する．ブロッククローン検出法では従来法において困難であった意味的に類似するコー

A Determination Method of Locality Sensitive Hashing Parameters for Code Clone Detection

Shogo Tokui, Katsuro Inoue, 大阪大学, Osaka University.
Norihiro Yoshida, 名古屋大学, Nagoya University.
Eunjong Choi, 奈良先端科学技術大学院大学, Nara Institute of Science and Technology.

†1　https://falconn-lib.org/

ドクローン対を検出することが可能である．ブロッククローン検出法は，他の検出手法と比べて検出結果の適合率や再現率が高い．また，大規模なプロジェクトに対して現実的な計算時間でコードクローン検出可能である．実際，ブロッククローン検出法を用いて Linux Kernel 4.14 のコードクローン検出を行うと，20 分程度で検出が完了し，100MLOC においても 4 時間程度で検出可能であった．

しかし，現状のブロッククローン検出法は，LSH における検出漏れが発生する問題がある [9]．つまり，類似度が閾値以上である検出されるべきクローンペアが，LSH を用いた類似探索において検出されない可能性がある．本研究では，検出漏れをユーザーが指定する割合に抑え，かつ可能な限り高速に実行するための，FALCONN に与えるパラメータを決定する手法を提案する．

以降，2 章では，ブロッククローン検出その関連技術について述べる．3 章では，FALCONN のパラメータの解析を行う．4 章では，FALCONN に与えるパラメータ決定手法の提案を行う．5 章では，有効性の評価を行う．最後に 6 章では，まとめと今後の課題について述べる．

2 ブロッククローン検出とその関連技術

本章では，ブロッククローン検出法と，ブロッククローン検出法が用いる LSH である FALCONN について説明を行い，最後にブロッククローン検出法の問題点について述べる．

2.1 ブロッククローン検出法

山中らは情報検索技術を利用して，意味的に処理が類似した関数単位でのコードクローンを検出する手法 [8] を提案した．しかし粒度が大きいため，小さい粒度のコードクローンが検出ができない問題があった．そこで，横井らはコードブロック単位での検出法を提案し，精度を向上させることに成功した [7]．ここで，コードブロックとは，if 文や for 文などの波括弧で囲まれたコード片を指す．

ブロッククローン検出法は以下のステップで実行される．

STEP 1 構文解析を行い，抽象構文木を生成

STEP 2 抽象構文木からコードブロックと単語を抽出

STEP 3 TF-IDF 法により，ブロック単位の特徴ベクトルを計算

STEP 4 FALCONN を利用した類似探索を行い，コサイン類似度が閾値以上のクローンペアを検出

STEP 4 で用いる LSH ライブラリである FALCONN について，2.2 節にて説明する．

2.2 近似最近傍探索ライブラリ FALCONN

本節では，ブロッククローン検出法が用いる近似最近傍探索ライブラリ FALCONN について述べる．

Alexandr らは，大規模なベクトル集合の近似最近傍探索問題を解くための手法として，理論的かつ実用的な手法 FALCONN を開発した [2]．以降，LSH の定義と cross-polytope LSH について述べ，FALCONN に実装されている LSH を利用した類似探索アルゴリズムについて説明する．

2.2.1 局所性鋭敏型ハッシュ(LSH)

LSH (Locality-Sensitive Hashing）とは，ハッシュ関数を用いて類似度が高いベクトル対を高速に見つけるアルゴリズムである [1]．LSH はは，あるベクトルと近似因数 $c < 1$ に対して，類似度が δ 以上のベクトルが存在するとき，類似度が $c\delta$ 以上のベクトルをすべて返す，LSH のハッシュ関数は以下の 2 つの定義を満たす．

定義 1. $S(x, y) > \delta \Rightarrow Pr(x, y) > P_1$

定義 2. $S(x, y) < c\delta \Rightarrow Pr(x, y) < P_2$

類似度 $S(x, y)$ が定義された d 次元空間上において，$Pr(x, y)$ は，2 つのベクトル x, y が同じハッシュ値を取る確率を表す．LSH では，$Pr(x, y)$ を衝突確率と呼ぶ．P_1, P_2 は $P_1 > P_2$ を満たす．LSH の 1 つのハッシュ関数の時間計算量のオーダーは $O(dn^\rho)$ と表される．d はベクトルの次元数，n はベクトル数を表す．ρ は P_1, P_2 から算出される時間計算量の評価基準であり，式 1 で表される値である [3]．

$$\rho = \frac{\log(1/P_1)}{\log(1/P_2)} \tag{1}$$

2.2.2 cross-polytope LSH

FALCONN は，コサイン類似度のための LSH であり，理論的に優れている cross-polytope LSH に基づく実用的なアルゴリズムである [2]．2 つのベクトル x, y に対して，コサイン類似度 $C(x, y)$ は式 2 で求められる．

$$C(x, y) = \frac{x \cdot y}{\|x\|\|y\|} \tag{2}$$

cross-polytope LSH における入力ベクトルのハッシュ値は，単位球上にランダム回転した後のベクトルが，含まれる区画のラベルである．FALCONN は，ランダム回転の処理に高速アダマール変換を用い，また，前処理として次元圧縮をするなどしてメモリ削減や高速化を行っている．

cross-polytope LSH では，ランダム回転後にその点が存在する単位球上の区画がそのベクトルのハッシュ値となる．cross-polytope LSH において，ある類似度 δ に対して，$C(x, y) > \delta$ をみたす 2 つのベクトル x, y の衝突確率は，単位球を T 個の区画に分割する場合，式 3 で表される [2].

$$\ln \frac{1}{Pr(x, y)} = \frac{1-\delta}{1+\delta} \cdot \ln T + O_\delta(\ln \ln T) \tag{3}$$

2.2.3 類似探索

ベクトル集合から，類似度が閾値 θ 以上のベクトル対をすべて取得することを類似探索という．最も単純な類似探索アルゴリズムである，すべてのベクトル対の類似度を計算し，閾値以上のベクトルペアを取得する方法では，時間計算量は $O(dn^2)$ となる．しかし，LSH を用いることによって，時間計算量を $O(Ldn^\rho)$ とできる．LSH を用いた類似探索のアルゴリズム [1] は，以下の 3 つのステップで構成される．

STEP A ベクトルの集合から L 個のハッシュテーブルを作成

STEP B いずれかのハッシュテーブルで衝突するベクトル対を類似度が閾値以上のベクトル対の候補として抽出

STEP C STEP B で抽出したすべてのベクトル対の類似度を計算

LSH によって類似度が閾値以上のベクトル対の候補を絞ることにより，類似度を計算するベクトル対の数を減らす事ができる．

2.3 ブロッククローン検出法の問題点

我々は先行研究において，LSH を用いるコードクローン検出法に対して，以下 2 つの問題点を指摘した [9]．1 つ目は，ブロッククローン検出法の STEP4(類似探索) にかかる時間が検出全体の約 90%を占めている点である．2 つ目は，LSH を用いた類似探索において，約 10%の検出漏れが発生している点である．

3 FALCONN のパラメータの解析

2.3 節で述べた問題は，計算時間を短くするために，ブロッククローン検出法の開発者が検出漏れが起こるような LSH のパラメータを選択していたことが原因として生じた．

本章では，コードクローン検出のための適切なパラメータを選択するために，FALCONN のパラメータの解析を行う．FALCONN のパラメータには 11 個のパラメータがある．ここではパラメータの中でも，検出精度を表す衝突確率に関わる区画の分割数 T とハッシュテーブル数 L について説明を述べる．

3.1 区画の分割数 T

パラメータ T と衝突確率，及び時間計算量の評価基準 ρ の関係を説明する．T とは，単位球を分割する区画の数を表す．分割数 T に指定できる範囲は，ベクトルの次元数以下の自然数である．FALCONN は 1024 次元に次元圧縮を行うため，T には $1 \leq T \leq 1024$ の整数を指定できる．

2.2.2 節の式 3 から，類似度 δ に対して，$C(x, y) > \delta$ をみたす 2 つのベクトル x, y の cross-polytope LSH における衝突確率 $Pr(x, y)$ は T の対数に比例し，T を増加させると衝突確率 $Pr(x, y)$ は減少する．

また，2.2.1 節の式 1 と 2.2.2 節の式 3 から，時間計算量の評価基準 ρ は式 4 と表せる．

$$\rho = \frac{\frac{1-\delta}{1+\delta} \cdot \ln T + O_\delta(\ln \ln T)}{\frac{c^2(1-\delta)}{2-c^2(1-\delta)} \cdot \ln T + O_{c\delta}(\ln \ln T)} \tag{4}$$

時間計算量の評価基準 ρ の値が小さいほど，時間計算量のオーダー $O(Ldn^\rho)$ は小さくなる．時間計算量の評価基準 ρ は，定数である近似因数 $c < 1$ と類似度 δ を除くと，分割数 T にのみ依存する．式 4 から，T が大きいほど，ρ は小さくなる．参考文献 [2]

の 4 章を参照すると，分割数 T を増加させると時間計算量の評価基準 ρ は単調減少し，$T \to \inf$ のとき $\rho \to \frac{1}{7}$ となる．

3.2 ハッシュテーブル数 L

ハッシュテーブル数 L と衝突確率，及び計算時間の関係を説明する．FALCONN では，あるベクトル対が L 個のハッシュテーブルの少なくとも 1 個のハッシュテーブルで衝突するとき，そのベクトル対を類似が閾値 θ 以上のベクトル対の候補とみなす．式 3 の cross-polytope LSH の衝突確率 $Pr(x,y)$ を用いて，FALCONN の衝突確率 $Pr_L(x,y)$ は式 5 と表される．

$$Pr_L = 1 - (1 - Pr(x,y))^L \tag{5}$$

式 5 から，L 個のハッシュテーブルでの失敗確率 $1 - Pr_L(x,y)$ は式 6 と表せる．

$$1 - Pr_L(x,y) = (1 - Pr(x,y))^L \tag{6}$$

$1 - Pr_L(x,y)$ とは，どのハッシュテーブルでも衝突しない確率を表す．式 6 から，1 つのハッシュテーブルでの失敗確率 $1 - Pr(x,y)$ に対して，ハッシュテーブル数 L を増加させると，L 個のハッシュテーブルでの失敗確率 $1 - Pr_L(x,y)$ は減少する．すなわち，ハッシュテーブル数 L を増加させると，FALCONN の衝突確率 $Pr_L(x,y)$ が増加する．

また，L と計算時間の関係を調べるために，ハッシュテーブル数 L のみ変化させて類似探索の時間を計測した．$1 \leq L \leq 30$ の範囲を 1 ずつ変化させて計測した，実験結果のグラフを図 1 に示す．図 1 のグラフより，L を増加させると計算時間は線形に増加する．よって，FALCONN の時間計算量は $O(Ldn^\rho)$ と表せる．

3.3 解析結果に対する考察

本節では，3.1 節と 3.2 節での解析結果から考察を行う．分割数 T を増加させると，FALCONN の時間計算量 $O(Ldn^\rho)$ の評価指標 ρ を減少できるが，1 つのハッシュテーブルにおける衝突確率 $Pr(x,y)$ も減少する．ハッシュテーブル数 L を増加させると，FALCONN の衝突確率 $Pr_L(x,y)$ は増加できるが，FALCONN の計算時間が線形に増加する．

分割数 T とハッシュテーブル数 L はいずれも，衝突確率 $Pr_L(x,y)$ と FALCONN の時間計算量 $O(Ldn^\rho)$ の一方が良くなる値にするとともう一方が悪くなる．大規模なプロジェクトに対してコードクローン検出することを想定しているため，FALCONN の時間計算量 $O(Ldn^\rho)$ の評価基準である ρ を可能な限り小さくする事を優先し，T は最大値である 1024 に決定する．そのとき衝突確率は小さくなるので，ハッシュテーブル数 L で調整する．ハッシュテーブル数 L を決定する手法を 4 章で説明する．

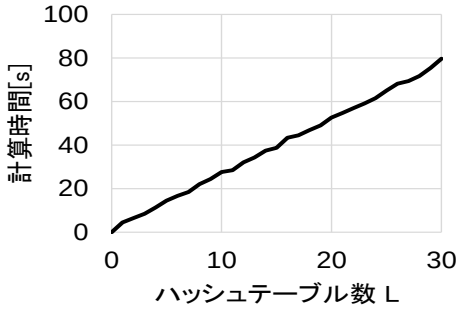

図 1　ハッシュテーブル数 L と計算時間の関係

4 FALCONN のパラメータ決定手法

本章では，類似度の閾値 θ と目標再現率 R を入力として，R 類似探索を行うための FALCONN に与えるパラメータを決定する手法を提案する．

4.1 再現率と R 類似探索の定義

本節では，再現率と R 類似探索の定義を行う．

まず，再現率 r の定義を行う．再現率 r は式 7 で与えられる．

$$r = \frac{|S_{LSH}|}{|S_{all}|} \tag{7}$$

類似度が閾値 θ 以上であるすべてのベクトル対の集合を S_{all}，LSH を用いた類似探索によって検出したベクトル対の集合を S_{LSH} とする．LSH を用いた類似探索では，2.2.3 節の STEP C で類似度の計算を行い，類似度が閾値 θ 以上のベクトル対を求めるため，包含関係 $S_{LSH} \subset S_{all}$ を常に満たす．

次に，R 類似探索の定義を行う．目標再現率 R に対して，R 以上の再現率を満たし，可能な限り少ない計算時間で実行される類似探索を，R 類似探索と

定義する．R 類似探索となるためのパラメータ決定手法について 4.2 節で述べる．

4.2 R 類似探索となるための FALCONN に与えるパラメータ決定

本節では，3 章で解析した区画の分割数 T とハッシュテーブル数 L の決定手法を述べる．

分割数 T は，3.3 節より，時間計算量のオーダー $O(Ldn^{\rho})$ の評価指標である ρ に影響を与える．類似探索の計算時間を可能な限り小さくするために，時間計算量の評価指標 ρ を可能な限り小さくする $T = 1024$ に決定する．このとき衝突確率が減少するため，再現率 R 以上をみたすようにハッシュテーブル数 L を調整する．

以下のステップでハッシュテーブル数 L を決定する．

STEP I 式 3 と $T = 1024$ から，類似度が閾値 θ の 2 点の衝突確率 $Pr(x, y)$ を計算

STEP II 式 5 と目標再現率 R に対し，式 8 をみたす最小値に決定

$$Pr_L(x, y) \geq R \tag{8}$$

STEP I で計算する $Pr(x, y)$ の値は，$T = 1024$ より式 9 で求められる．

$$Pr(x, y) = 1024^{\frac{1-\theta}{1+\theta} - O_{\theta}(1)} \tag{9}$$

このとき，3.2 節の式 6 からハッシュテーブル数 L を増加させると計算時間が線形に増加するため，STEP II では，式 8 をみたす範囲でハッシュテーブル数 L を最小の値に決定する．

STEP II の式 8 は以下の定理 1 に従う．

定理 1. ある目標再現率 R に対して，*FALCONN* の衝突確率 $Pr_L(x, y)$ が式 *8* を成り立たせるとき，再現率の期待値は R 以上となる．

Proof. 2.2.3 節より，$Pr_L(x, y)$ は L 個のハッシュテーブルに対する衝突確率を表し，ある 1 つの類似ペアが検出できる確率と言える，$|S_{all}|$ 個のベクトル対は，それぞれ確率 $Pr_L(x, y)$ で衝突するから，衝突する衝突するベクトル対の数は二項分布に従う．よって，検出できるベクトル対の数の期待値 E は式 10 で求められる．

$$E = |S_{all}| \times Pr_L(x, y) \tag{10}$$

よって，再現率の期待値 E_{recall} は，式 11 で表される．

$$E_{recall} = \frac{E}{|S_{all}|} = Pr_L(x, y) \tag{11}$$

式 8 と式 11 から，式 12 が導ける．よって，再現率の期待値が目標再現率 R 以上となる．

$$E_{recall} \geq R \tag{12}$$

□

5 評価実験

本章では，本手法をブロッククローン検出法に適用し，4.2 節で決定したパラメータと，FALCONN のデフォルトのパラメータとでの比較を行い，本手法の有効性の評価を行う †2．

異なる規模の 2 つの C プロジェクトである PostgreSQL 10.1 と Linux Kernel 4.14 に対して，ブロッククローン検出法でコードクローン検出を行い，類似探索の処理時間と類似探索の再現率を調べる．対象のプロジェクトの情報を表 1 に示す．

ここでは，ブロッククローン検出法のコサイン類似度の閾値 θ は，デフォルトである 0.9 とし，目標再現率 R を 0.99 とした．このとき，本手法により決定された FALCONN に与えるパラメータを表 2 に示す．デフォルトは，FALCONN のデフォルトのパラメータを表す．FALCONN に与えるパラメータは他にも並列化のためのパラメータなどがあるが，それらのパラメータは，並列化をしないなどの統一を行った．

評価実験の結果を表 3 に示す．2 つのプロジェクトについて，目標再現率 0.99 のために決定したパラメータが再現率 0.99 を満たすことが確認できた．また，FALCONN のデフォルトのパラメータよりも高速に処理が完了した．そのため，本手法で決定したパラメータは R 類似探索の条件を満たす．

FALCONN のデフォルトのパラメータのとき，再現率が 1 となったことから，時間をかけて処理するパラメータにしていると考えられる．また 2 章で，ブロッククローン検出法において検出漏れがあったことを述べたが，それは検出時間を削減するために検出漏れが起こるパラメータに変更したためであると考え

†2 CPU Intel Xeon 2.80GHz 4core，メモリ 32.0GB，OS Windows 10 64bit. Java 仮想マシンのスタック領域 1GB，ヒープ領域 15GB

表 1　対象プロジェクト

プロジェクト	行数	コードブロックの数	単語の数	クローンペアの数
PostgreSQL	1,138,235	24,788	11,937	8,291
Linux Kernel	17,474,091	424,236	78,675	117,490

表 2　FALCONN に与えるパラメータ

		デフォルト	本手法
PostgreSQL	T	2	1024
	L	10	4
Linux Kernel	T	32	1024
	L	10	4

表 3　実験結果

		デフォルト	本手法
PostgreSQL	計算時間	5m 2s	10s
	再現率	1.000	0.998
Linux Kernel	計算時間	6h 45m 10s	20m 6s
	再現率	1.000	0.998

られる．実験結果から，本手法によって決定したパラメータで R 類似探索を行えることが確認できたため，本手法の有効性が確認できた．

6　まとめと今後の課題

本研究では，ブロッククローン検出法が用いる LSH のパラメータの解析を行い，R 類似探索を行うためのパラメータ決定手法を提案した．そして，本手法をブロッククローン検出法に適用し，2 つの異なる規模のプロジェクトに対してコードクローン検出を行った．その結果，デフォルトのパラメータより短時間で，目標再現率 R 以上となることを確認した．

今後の課題として，以下が挙げられる．

- BigCloneBench [5] を用いて，検出漏れとなるクローンペアの特徴に関する調査
- 任意の目標再現率や閾値における速度と再現率の評価
- 他のコードクローン検出法に対する有効性の評価

謝辞 本研究は JSPS 科研費 25220003，16K16034，15H06344 の助成を受けた．

参 考 文 献

[1] Alexandr, A. and Piotr, I.: Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions, *Foundations of Computer Science*, Vol. 51, No. 1(2006), pp. 117–122.
[2] Alexandr, A., Piotr, I., Thijs, L., Ilya, R., and Ludwig, S.: Practical and Optimal LSH for Angular Distance, *Proc. of NIPS 2015*, 2015, pp. 1225–1233.
[3] Alexandr, A., Thijs, L., Ilya, R., and Erik, W.: Optimal hashing-based time-space trade-offs for approximate near neighbors, *Proc. of SODA 2017*, 2017, pp. 47–66.
[4] Baeza-Yates, R. and Ribeiro-Neto, B.: *Modern Information Retrieval: The Concepts and Technology behind Search*, Addison-Wesley, 2011.
[5] Jeffrey, S., F, I. J., Iman, K., K, R. C., and Mamun, M. M.: Towards a big data curated benchmark of inter-project code clones, *Proc. of ICSME*, 2014, pp. 476–480.
[6] Rattan, D., Bhatia, R., and Singh, M.: Software clone detection: A systematic review, *Information and Software Technology*, Vol. 55, No. 7(2013), pp. 1165–1199.
[7] 横井一輝, 崔恩瀞, 吉田則裕, 井上克郎: 情報検索技術に基づくブロッククローン検出, 電子情報通信学会技術研究報告, Vol. 117, No. 136(2017), pp. 109–116.
[8] 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(2014), pp. 2245–2255.
[9] 徳井翔梧, 吉田則裕, 崔恩瀞, 井上克郎: 局所性鋭敏型ハッシュを用いたコードクローン検出のためのパラメータ決定手法, Vol. 117, No. 477, 2018, pp. 57–62.
[10] 肥後芳樹, 楠本真二, 井上克郎.: コードクローン検出とその関連技術, 電子情報通信学会論文誌 *D*, Vol. 91, No. 6(2008), pp. 1465–1481.