一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
SS2017-74 (2018-03)

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

徳井　翔梧† 　吉田　則裕†† 　崔　　恩瀞††† 　井上　克郎†

† 大阪大学大学院情報科学研究科, 〒 560-0871 大阪府吹田市山田丘 1-5
†† 名古屋大学大学院情報学研究科, 〒 464-8601 愛知県名古屋市千種区不老町
††† 奈良先端科学技術大学院大学情報科学研究科, 〒 630-0192 奈良県生駒市高山町 8916-5
E-mail: †{s-tokui,inoue}@ist.osaka-u.ac.jp, ††yoshida@ertl.jp, †††choi@is.naist.jp

あらまし　局所性鋭敏型ハッシュ (LSH) とは，高次元なデータを確率的にハッシュし，近傍点を見つけるアルゴリズムである．クローン検出において LSH の処理に大きな時間を占めており，LSH の処理速度を向上させることが求められている．本研究では，検出精度を変えずに高速に LSH の処理をするために，LSH に与えるパラメータを，対象ソースコードから決定する手法を提案する．

キーワード　コードクローン，局所性鋭敏型ハッシュ，コードブロック

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

Shogo TOKUI†, Norihiro YOSHIDA††, Eunjong CHOI†††, and Katsuro INOUE†

† Graduate School of Information Science and Technology, Osaka University, 1-5, Yamadaoka, Suita, Osaka, 565-0871, Japan
†† Graduate School of Informatics, Nagoya University, Furo-cho, Chikusa-ku, Nagoya, Aichi, 464-8601, Japan
††† Graduate School of Information Science, Nara Institute of Science and Technology, 8916-5, Takayama, Ikoma, Nara, 630-0192,Japan
E-mail: †{s-tokui,inoue}@ist.osaka-u.ac.jp, ††yoshida@ertl.jp, †††choi@is.naist.jp

Abstract　LSH (Locality-Sensitive Hashing) is a near neighbor search algorithm that performs probabilistic hashing of high-dimensional data. For reducing the detection time of code clones, the execution time of LSH needs to be speeded up. In this study, we propose an approach to determine LSH parameters based on source code analysis in order to not only speed up code clone detection but also keep the preciseness of it.

Key words　Code Clone, Locality-Sensitive Hashing, Code block

1. まえがき

ソフトウェアの保守における問題の 1 つとして，コードクローンが指摘されている [1], [2]．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，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．大規模のソースコードからは多くのコードクローンが検出されるため，手作業でそれらを管理することは困難である．そこで，コードクローンを自動的に検出することを目的としたさまざまなコードクローン検出手法が提案されている．

横井らは情報検索技術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（ブロッククローン）を検出するブロッククローン検出法 [3] を提案し，有効性を確認した．ブロッククローン検出法では，まず，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，ソースコード中の識別子や予約語に利用される単語に対して重み付けを行い，各コードブロックを特徴ベクトルに変換する．特徴ベクトル間の余弦類似度が閾値以上のクローンペアを，局所性鋭敏型ハッシュ (LSH) [5] を用いた類似探索によって高速に求め，コードクローンの検出を行う．

しかし，20MLOC の大規模プロジェクトに対してブロッククローン検出法を適応した場合，各ステップ毎にかかる時間を

計測したところ，21 分の検出時間の内，特徴ベクトル間の類似探索に 18 分かかっていた事を確認した．LSH を用いた類似探索に約 90%の時間がかかっており，全体の計算時間を削減するために，類似探索の高速化が必要となる．ただし，LSH は，計算時間と探索精度がトレードオフの関係にある．

そこで，本研究では，ブロッククローン検出法において高速化をするための，LSH のパラメータ決定手法を提案する．評価実験では，これまでブロッククローン検出法にて使用されていたパラメータと，本手法を用いて決定したパラメータをブロッククローン検出法に適用した場合の，類似探索に費やす時間と探索精度の観点から比較を行った．

以降，2 章では，LSH ライブラリの 1 つである FALCONN [6](注1)とブロッククローン検出法のアルゴリズムについて述べる．3 章では，FALCONN に与えるパラメータの種類と，本研究で提案するパラメータの決定手法を述べる．4 章では，一部のパラメータについて，探索時間や再現性に与える影響の調査を行う．5 章では，本手法の評価実験について述べ，6 章で考察を行う．最後に，7 章でまとめと今後の課題について述べる．

2. 関 連 研 究

2.1 近似最近傍探索問題用ライブラリ FALCONN

Andoni らは，大規模高次元なベクトル集合の近似最近傍探索問題を解くための LSH である FALCONN を提案した [6]．FALCONN は，計算コスト及び空間コストに関して優れた Cross-Polytope LSH に基づいて，機械学習を用いた次元圧縮や，探索精度を向上させるための multi-probe LSH [7] を用いて，大規模高次元な探索問題に対応可能にしている．

2.1.1 局所性鋭敏型ハッシュ (LSH) の定義

LSH (Locality-Sensitive Hashing）とは，ハッシュ関数を用いて近傍点を探索するアルゴリズムである [5] では，データ間の距離が近い時にハッシュ値が一致する．正確な定義を以下の通りである．

ノルム $\|\cdot\|$ が定義された d 次元空間上の任意の 2 点 p, q をとる．距離 r 内の 2 点を求めるための LSH のハッシュ関数は，ある $c > 1$ と $P_1 > P_2$ を満たす確率 P_1, P_2 に対して，以下の 2 つの条件を満たす．

[定義 1] $\|p - q\| < r \Rightarrow Pr(p, q) > P_1$

[定義 2] $\|p - q\| > cr \Rightarrow Pr(p, q) < P_2$

$Pr(p, q)$ とは p と q が同じハッシュ値を取る確率を表し，LSH では衝突確率と呼ばれる．

2.1.2 FALCONN のハッシュ関数

Andoni らが提案する FALCONN は，単位球面上のユークリッド距離に対して近似最近傍探索を行うため，余弦類似度にも対応している．余弦類似度を用いる TF-IDF 法や画像の類似探索を行う場合に適している．単位球面上の点の FALCONN のハッシュ値は以下のステップで求められる．

STEP A Feature Hashing [8] を用いて任意の次元に圧縮

STEP B 高速アダマール変換を応用した擬似ランダム回転 [9] を任意の回数実行

STEP C 単位球を一定の任意の大きさに分割し，その点が含まれる区画のラベルをハッシュ値として取得

STEP A はメモリ削減と高速化のために行う．STEP B, C は Cross-Polytope LSH に基づいた操作である．FALCONN のハッシュ関数において，距離 $r > 0$ に対し $\|p - q\| < r$ を満たす p, q が衝突する確率は式 1 を満たす．

$$\ln \frac{1}{Pr(p, q)} = \frac{r^2}{4 - r^2} \cdot \ln d + O_r(\ln \ln d) \quad (1)$$

2.2 ブロッククローン検出法

2.2.1 ブロッククローン検出法のアルゴリズム

横井らは，情報検索技術を利用することによって，意味的に処理が類似したブロッククローンを検出する手法を提案した [3]．ブロッククローン検出法では，ブロック単位の各コード片に対しベクトル化を行うため，コード片同士が完全に一致するタイプ 1 から意味的に類似するタイプ 4 までのコードクローン [10] を検出できる．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，類似ペアの探索 (以降，類似探索とする) を行い，クローンペアを検出

STEP4 の類似探索は以下のステップで実行される．

STEP4-1 ブロック全ての特徴ベクトルから FALCONN のハッシュ関数を用いてハッシュテーブルを作成

STEP4-2 特徴ベクトル毎に，STEP4-3，4-4，4-5 を実行

STEP4-3 特徴ベクトルのハッシュ値を求め，ハッシュテーブルにおいて衝突するベクトルを探索．また，multi-probe LSH に基づき，近いハッシュ値のベクトルも探索

STEP4-4 見つけた全ての近傍ベクトルとの距離を計算し，閾値以内のベクトルのコード片をクローンペアとして検出

2.2.2 ブロッククローン検出法の使用の問題点

2.2.1 節では，ブロッククローン検出法のアルゴリズムについ

表 1 ブロッククローン検出法の各ステップにかかる時間

プロジェクト名	行程	計算時間 [s]	相対時間
PostgreSQL 10.1	STEP1	4.2	0.06
	STEP2	0.4	0.01
	STEP3	0.2	0.01
	STEP4	65.6	0.93
	全行程	70.4	1.00
Linux Kernel 4.14	STEP1	157.7	0.12
	STEP2	1.3	0.01
	STEP3	5.8	0.01
	STEP4	1098.1	0.87
	全行程	1262.9	1.00

(注1)：https://falconn-lib.org/

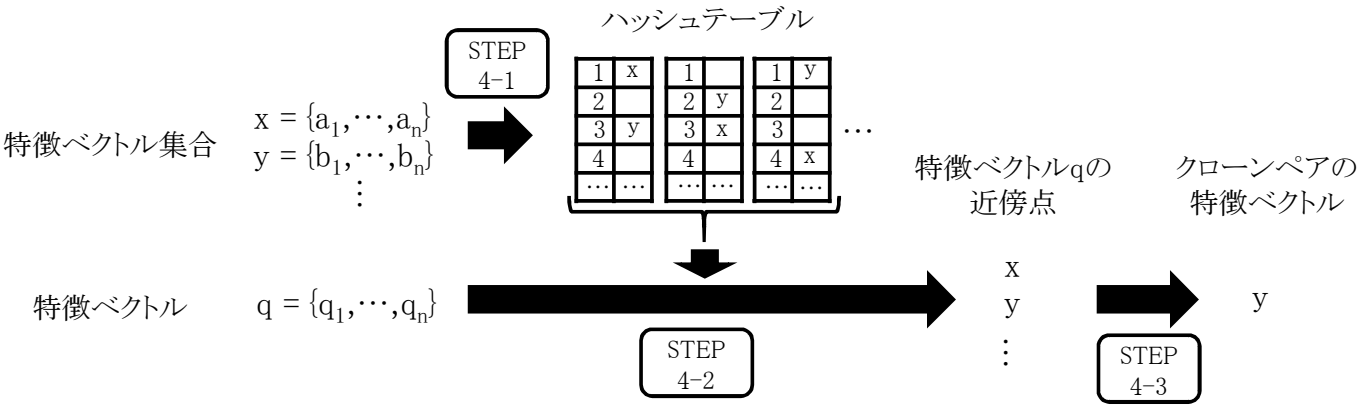

図 1　類似探索の実行フロー

て説明した．各ステップにかかる時間を PostgreSQL 10.1(注2) と Linux Kernel 4.14(注3) の 2 つのプロジェクトに対して計測した結果を表 1 に示す．この結果と LSH が確率的なアルゴリズムである事から，ブロッククローン検出法に対して以下の問題点が挙げられる．

1 つ目は，ブロッククローン検出法の STEP4 にかかる時間が検出全体の約 90%を占めている点である．実際，表 1 の結果から，それぞれ STEP4 に約 90%の時間がかかっている．

2 つ目は，STEP4 において LSH が用いるため，クローンペアの検出漏れを起こす可能性がある点である．実際，STEP4 において LSH を用いずに求めたクローンペアの数と，LSH を用いて求めたクローンペアの数を比較した結果，5%程度の検出漏れを起こしていた．

3. パラメータ決定手法

本研究では，2.2.2 節で述べた 2 つの問題点を踏まえて，ブロッククローン検出法における STEP4 (類似探索) の改善を行うために，類似探索に用いる FALCONN に対して，検出対象ごとに再現率を一定以上保証した上で最小の計算時間となる，適切なパラメータを決定する手法を提案する．本章では，パラメータ決定の方針と，その根拠を述べる．

TF-IDF 法により生成されたブロッククローンの特徴ベクトル間の類似探索は，d 次元実数空間において余弦距離に対して行われる．特徴ベクトルの数は，ソースコード中のコードブロックの数である，特徴ベクトルの次元は，ソースコード中に出現するワードの種類の数である．

ブロッククローン検出法において生成される特徴ベクトルの集合は，大規模多次元になりやすい．実際，ソースコード中のコードブロックの数は行数に比例して増え，1MLOC の Java プロジェクトには約 4 万ブロック含まれていた．TF-IDF 法では，変数名を含めたワードの種類の数がそのまま特徴ベクトルの次元となるため，プロジェクトの規模が大きくなるとワードの種類が増加し，実際 1MLOC の Java プロジェクトには約 1 万種類のワードが存在した．

(注2)：https://github.com/postgres/postgres
(注3)：https://github.com/torvalds/linux

表 2　FALCONN に与えるパラメータ

パラメータ名	内容
lsh_family	ハッシュ関数の種類
k	テーブル毎のハッシュ関数の数
stroge_hash_table	ハッシュテーブルのデータ構造
Threads	スレッドの数
distance_function	距離関数の種類
num_rotations	擬似ランダム回転の回数
FHD	次元圧縮後の次元数
probes	近接バケットの探索数
LCD	cross-polytope LSH で用いる球面の分割数
l	ハッシュテーブルの数

3.1 パラメータ決定の方針

あるベクトルに対し距離が閾値以内のベクトルを求める時，全ての距離を計算する前に，FALCONN により閾値以内のベクトルの候補として近傍ベクトルを求め，距離を計算するコストを削減する．近傍ベクトルに含まれないベクトルは，閾値以内のベクトルとして検出されることはない．だが，FALCONN に与えるパラメータを変更し，閾値以内のベクトルと衝突する確率を上げることができる．ただし，衝突確率を上げると，近傍ベクトルが増加し，距離の計算コストが増加する．

パラメータは以下の要件を満たすものを指定する．

p 類似探索

再現率 p 以上
できるだけ小さい探索時間の類似探索

再現率

p 類似探索の再現率とは，LSH を使わずにすべての組み合わせの類似度を計算して検出したクローンペアの検出数の内，FALCONN が検出したクローンペアの検出数の割合を表す．

探索時間

p 類似探索の探索時間とは，ブロッククローン検出法のアルゴリズムの STEP4-1 から STEP4-5 までにかかる時間とする．再現率を p 以上にできるパラメータの中で，できるだけ少ない探索時間となるパラメータを決定する．

3.2 FALCONN に与えるパラメータ

本節では，表 2 に示された FALCONN に指定する 10 種類のパラメータについて説明し，パラメータの決定手順とそれにより決定したパラメータを説明する．FALCONN に与えるパラメータは，以下のステップで決定する．

STEP I FALCONN 製作者が推奨するパラメータに固定

STEP II ブロッククローン検出法に適したパラメータに固定

STEP III 規模に応じて変更するパラメータの決定

3.2.1 節から 3.2.3 節まで，パラメータの決定手順に基づきパラメータを決定する手順を説明する．

3.2.1 FALCONN 製作者が推奨するパラメータに固定

FALCONN に与えるパラメータの内，FALCONN 製作者が推奨する値に固定するパラメータを表 3 に示す．以下で，パラメータに関する説明と，Andoni らが推奨する固定値について述べる．

lsh_family では，ハッシュ関数として cross-polytope LSH か，hyperplane LSH の 2 種類を選択できる．Andoni らの手法は cross-polytope LSH に基づいて設計しているため，cross-polytope LSH に固定する．

k では，テーブル毎のハッシュ関数の数を決定する．ハッシュ関数が多いほど，近いベクトルの候補の数を減らすことができるため，大規模なデータセットから最近点を求める場合には k を大きくし，近傍点をできるだけ減らす必要がある．しかし，本研究では閾値以内のペアを全て見つけることを目的としているため，k は Andoni らが推奨する 2 に固定する．

storage_hash_table では，ベクトルの集合の規模や次元の大きさに合うテーブルのデータ構造が用意されており，省メモリな LSH を実現するために必要である．ブロッククローン検出法において特徴ベクトルの集合は大規模であるため，Andoni らが推奨しているできるだけ少ないビット数でハッシュテーブルのバケットを構築するように指定する BitPackedFlatHashTable にする．

Threads では，FALCONN が使用可能なハードウェアのスレッドの数を指定できる．ただし，0 を与えると，最大数のスレッドを自動で指定させることができるため，Threads は 0(最大数) を指定する．

3.2.2 ブロッククローン検出法に適した値に固定するパラメータ

FALCONN に与えるパラメータの内，ブロッククローン検出法に適した値に固定するパラメータを表 4 に示す．以下で，パラメータに関する説明と，ブロッククローン検出法に適した固定値について述べる．

distance_function では，距離関数として余弦距離かユークリッド距離の 2 種類を選択できる．TF-IDF 法では余弦類似度を用いるため，類似探索の前にすべての特徴ベクトルに対し正規化を行う．正規化された 2 つのベクトル p, q に対して，なす角を θ とすると，式 3 で示されるように，ユークリッド距離は余弦距離に対し単調減少する．

$$\|p - q\|^2 = 2 - 2 \cdot \cos\theta \tag{2}$$

このため距離関数として選ぶものはどちらでもよい．しかし，余弦距離を指定した場合，再現率が 5%程度減少することが実験により分かったため，高い再現率を実現するためユークリッド距離を指定する．

num_rotations では，擬似ランダム回転の回転数を指定できる．cross-polytope LSH では，ベクトルをランダムに回転させ，理論上の衝突確率を一定に保つ．高速アダマール変換を応用した擬似ランダム回転を用いて，理論上のランダム回転と同じ振る舞いにするために，3 回転させる．このとき，高速アダマール変換を用いるため，次元圧縮後の次元数は，2 の累乗を指定しなければならない．

FHD(feature_hashing_dimension) では，次元圧縮後の次元数を指定できる．機械学習を用いた次元圧縮を用いており，2 点間の距離をほとんど変化させずに次元圧縮できる．ただ，圧縮後の次元を小さくしすぎると 2 点間の距離の誤差が大きく生じる可能性があるため，圧縮後の次元は Andoni らが推奨している 1024 に固定する．FHD が計算時間や再現率に与える影響は 4. 章で述べる．擬似ランダム回転に用いられる高速アダマール変換のために，圧縮後の次元は 2 の累乗を指定しなければならない．

probes では，multi-probe に基づいて近接バケットの探索数を指定できる．各テーブルにつき 4 つずつ近接バケットを探索させることにより検出漏れを減らす．4. 章で詳細について述べる．

3.2.3 規模に応じて変更するパラメータ

本手法がプロジェクトの規模毎に決定する FALCONN に与える 2 種類のパラメータを表 5 に示す．2 種類のパラメータについて述べる．

LCD (last_cp_dimension) では，cross-polytope LSH で用いる球面の分割数を指定できる．分割数を多くするほど衝突確率は下がる．球面を T 個の区画に分割した時，cross-polytope LSH の性質上，距離 r 以内の 2 点 p, q の衝突確率は式 3 で計算される．この衝突確率が 0.8 以上になるような最大の T を閾値に対して計算して決定する．4. 章で LCD と計算時間や再現率の関係を述べる．

$$\ln \frac{1}{Pr(p,q)} = \frac{r^2}{4 - r^2} \cdot \ln T + O_r(\ln \ln T) \tag{3}$$

l ではハッシュテーブルの数を指定できる．ハッシュ関数が全て異なるハッシュテーブルを複数用意し，少なくとも l つのハッシュテーブルで衝突したベクトルを近傍点とし，衝突確率を上げる．ただし，4. 章で詳しく述べるが，ハッシュテーブルの数を増やすと，線形に近傍点が増加するため，計算コストが線形に増加する．ハッシュテーブルが L 個の時，2 点 p, q の衝突確率 $Pr_L(p,q)$ は式 4 で求められる．

$$Pr_L(p,q) = 1 - (1 - Pr(p,q))^L \tag{4}$$

ここで，N 個の特徴ベクトル全てが閾値内の点と衝突する確率として，$Pr_R(p,q)$ を以下のように定義する．Pr_R は．再現

表 3　FALCONN 製作者が推奨するパラメータ

パラメータ名	値
lsh_family	cross-polytope LSH
k	2
storage_hash_table	BitPackedFlatHashTable
Threads	0(最大数)

表 4　ブロッククローン検出法に適した値に固定するパラメータ

パラメータ名	値
distance_function	Euclidean distance
num_rotations	3 回転
FHD	1,024
probes	テーブル毎に 4 つのバケットを探索

表 5　規模に応じて変更するパラメータ

パラメータ名	値
LCD	$Pr(p,q) > 0.8$ & 最小
l	$Pr_R > p$ & 最大

率の理論上の期待値である．再現率を p 得るために，期待値である Pr_R が p 以上になるような最小のテーブル数 L を特徴ベクトルの数に対して計算して決定する．Pr_R は式 5 で求められる．

$$Pr_R = Pr_L(p,q)^N \tag{5}$$

4. パラメータが探索時間や再現性に与える影響の調査

FALCONN に与えるパラメータの内，FHD，LCD，l，probe の 4 つのパラメータについて，探索時間や再現性に与える影響の調査を PostgreSQL 10.1 を対象に行う．調査結果を図 2 から図 5 に示す．調査結果は，調査対象以外のパラメータを全て固定した状態で調査対象のパラメータを変化させる．探索時間は，3 回の実行時間の平均を取る．

FHD と探索時間と再現性の関係を表すグラフを図 2 に示す．FHD は主に探索時間に大きな影響を与える．FHD の値を小さくすると，次元圧縮の影響により近傍点が多く見つかり，距離の計算に多大な時間がかかる．その反面，FHD の値をを大きくすると，近傍点が減少するが，次元が大きくなることにより，擬似ランダム回転にかかるコストが大きくなる．

閾値 0.9 において，LCD と探索時間と再現性の関係を表すグラフを図 3 に示す．LCD を増加させると，再現率は減少し，$LCD \geqq 64$ において探索時間はほぼ一定である．$LCD = 32$ において $Pr(p,q) = 0.803$ となる．

l と探索時間と再現性の関係を表すグラフを図 4 に示す．l の値を増加させると，各ハッシュテーブルでほとんど同じ数の近傍点を取得し，距離の計算時間が線形に増加するため，探索時間は線形に増加する．ハッシュテーブル毎にランダムなハッシュ関数が生成されるため，再現率は l の増加とともに増加する．

probe と探索時間と再現性の関係を表すグラフを図 5 に示す．probe の値を増加させると，探索時間はほぼ線形に増加しながら，再現率も増加する．確認する近接バケットを増やす毎に，近傍ベクトルとみなすベクトルが増え，距離の計算時間が増加するためである．

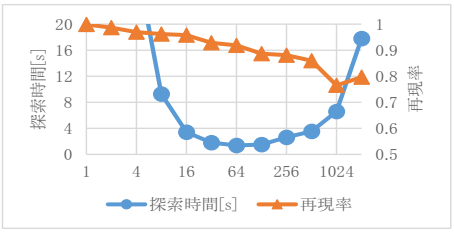

図 2　FHD が探索時間や再現性に与える影響

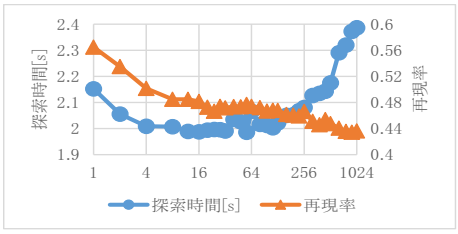

図 3　LCD が探索時間や再現性に与える影響

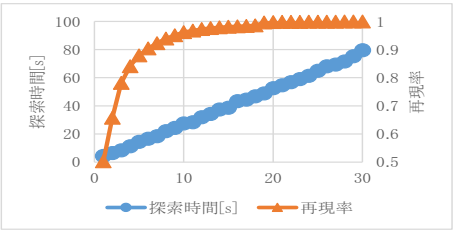

図 4　l が探索時間や再現性に与える影響

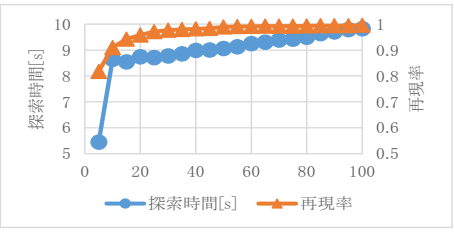

図 5　probes が探索時間や再現性に与える影響

5. 提案手法の評価実験

本章では，本手法で提案したパラメータ決定手法の評価実験(注4)について述べる．本手法で決定したパラメータとこれまでブロッククローン検出法で使用していたパラメータとの比較を，類似探索にかかる時間と再現率の観点で，異なる規模の 2 つの C プロジェクトの PostgreSQL 10.1 と Linux Kernel 4.14 に対して行う．対象のプロジェクトの情報を表 6 に示す．

実験は以下の手順で行う．まず，LSH を使わずにすべての組み合わせの類似度を計算してブロッククローン検出を行う．既に使用中のパラメータと本手法で決定したパラメータを，それぞれブロッククローン検出法に適用する．各プロジェクトに対

(注4)：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 6　プロジェクトの情報

プロジェクト名	メトリクス	値
PostgreSQL 10.1	行	1,136,684
	コードブロック	30,537
	ワード	11,366
	閾値 0.9 のクローンペア	12,629
Linux Kernel 4.14	行	17,407,666
	コードブロック	508,982
	ワード	77,319
	閾値 0.9 のクローンペア	163,535

表 7　実 験 結 果

		前パラメータ	本手法
PostgreSQL 10.1	探索時間 [s]	65.64	14.78
	再現率 [%]	93.10	99.21
Linux Kernel 4.14	探索時間 [s]	1098.1	334.73
	再現率 [%]	89.62	99.86

してそれぞれのパラメータでのブロッククローン検出を 3 回ずつ行い，類似探索の平均探索時間と，再現率を比較する．

実験結果を表 7 に示す．規模が異なる 2 つのプロジェクトに対して，前パラメータの場合から，それぞれ 60%以上の探索時間の削減を行うことができ，本手法で決定したパラメータでは，$p = 0.99$ の p 類似探索の要件である 99%の再現率を満足している．

6. 考　　察

本章では，評価実験の結果に対する議論を行い，他の検出手法や他のベクトル化手法に対する本手法の有用性について考察を行う．

再現率と探索時間

評価実験では 2 つの異なる規模のプロジェクトに対し実験を行った．いずれの規模でも再現率 0.99 を満足する結果が得られ．期待する再現率を得られることが確認できた．また，いずれの規模でも 60%程度の探索時間の短縮が確認できた．

本手法の有効性

本手法は余弦距離の下での p 類似探索に対して提案している．そのため，検出手法に依存しておらず，ブロッククローン検出法以外の手法においても，余弦距離の下での p 類似探索を行う手法には，本手法で決定するパラメータを用いる FALCONN を適用可能である．

本研究では FALCONN を対象にパラメータ決定を行った．本研究は，閾値以内の衝突確率を一定以上にするパラメータを決定し，さらに再現率 p を満たすパラメータを決定する．本手法を他の LSH に適用するためには，衝突確率や再現率を調整するためのパラメータがある事，またはユーザーがそれらを外部から調整する方法がある事が必要である．

7. まとめと今後の課題

本研究では，p 類似探索のためのパラメータ決定手法を提案した．パラメータを，再現率が理論上 p 以上となるようなパラメータ決定を行うことで，p 類似探索の要件である，再現率 p 以上を満足し，類似探索の時間の削減を実現した．

評価実験では，異なる規模の 2 つの C プロジェクトの PostgreSQL 10.1 と Linux Kernel 4.14 に対し，再現率と探索時間に関して，既にブロッククローン検出法で使用中のパラメータと本手法で決定したパラメータとの比較を行った．その結果，本手法で決定したパラメータは，前パラメータよりも探索時間を短くすることに成功した．また，$p = 0.99$ として p 類似探索をしているが，要件である再現率 99% を満足している．

今後の課題として，以下が挙げられる．

- 関数クローン検出法 [11] など，ブロッククローン検出法と同様の類似探索を行う手法に対して適用し，有効性を評価する必要がある．
- 本研究では TF-IDF 法に対して評価実験を行ったが，他のベクトル化手法に対しても実験を行い，有効性を評価する必要がある．
- 本研究では FALCONN に対してパラメータ決定を行ったが，他の LSH にはないパラメータが存在するため，本手法を他の LSH に適用するための拡張性について再考する必要がある．

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

文　　献

[1] 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，vol.J91-D，no.6，pp.604–613，1998.
[2] Ratta D., Bhatia R., and Singh M., “Software clone detection: A systematic review,” Information and Software Technology, vol.55, no.7, pp.1165–1199, 2013.
[3] 横井一輝，吉田則裕，崔　恩瀞，井上克郎，“情報検索技術に基づくブロッククローン検出，” 情報処理学会研究報告，vol.2017-SE-196，no.19，pp.1–8，2017.
[4] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval: The Concepts and Technology behind Search, Addison-Wesley, 2011.
[5] A. Alexandr and I. Piotr, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions,” Foundations of Computer Science, vol.51, no.1, pp.117–122, 2006.
[6] A. Alexandr, I. Piotr, L. Thijs, R. Ilya, and S. Ludwig, “Practical and optimal LSH for angular distance,” Proc. of NIPS 2015, pp.1225–1233, 2015.
[7] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li, “Multi-probe LSH: efficient indexing for high-dimensional similarity search,” Proc. of VLDB, pp.950–961, 2007.
[8] K.Q. Weinberger, A. Dasgupta, J. Langford, A.J. Smola, and Josh Attenberg., “Feature hashing for large scale multitask learning,” Proc. of ICML 2009, pp.1113–1120, 2009.
[9] N. Ailon and B. Chazelle, “The fast johnson-lindenstrauss transform and approximate nearest neighbors,” Society for Industrial and Applied Mathematics, vol.39, no.1, pp.302–322, 2009.
[10] C.K. Roy, J.R. Cordy, and R. Koschke, “Comparison and evaluation of code clone detection techniques and tools,” Science of Computer Programming, vol.74, no.7, pp.470–495, 2009.
[11] 山中裕樹，吉田則裕，崔　恩瀞，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.