情報処理学会論文誌　データベース　Vol.6　No.5　62–70 (Dec. 2013)

幾何学的なサフィックス木による高速類似構造検索手法

高橋 誉文[1,a)]　田村 慶一[1,b)]　黒木 進[1,c)]　北上 始[1,d)]

受付日 2013年6月21日, 採録日 2013年10月3日

概要：本論文では，大規模の蛋白質立体構造データベースに対する高速な類似構造検索を実現するために，従来の幾何学的サフィックス木の問題点を解決する方式を提案する．具体的には，提案手法は，大容量の幾何学的サフィックス木のノードが格納されたデータページのやりとりをするバッファ管理法，類似構造検索の高速化ための隠れ配列法より構成されている．データページのやりとりをするバッファ管理法については，木の上位レベルのノードがバッファ上に残るようにデータページの制御を行っている．隠れ配列法については，解をサーチするためのディスク I/O 回数を低減させるために隠れ配列と呼ばれるデータ構造を木の各ノードに追加している．提案方式の有効性を確認するために，提案方式を実装し，類似構造検索の速度や精度を測定し，従来の幾何学的サフィックス木と比較している．その結果，提案方式ではスケーラビリティに優れ，大規模データに対して従来の幾何学的サフィックス木よりも高速な類似構造検索を提供できることが分かった．

キーワード：データベース，蛋白質，立体構造検索

High-speed Similar Structure Search Method Using Geometrical Suffix Trees

Yoshifumi Takahashi[1,a)]　Keiichi Tamura[1,b)]　Susumu Kuroki[1,c)]
Hajime Kitakami[1,d)]

Received: June 21, 2013, Accepted: October 3, 2013

Abstract: In this paper, in order to achieve a similar high-speed structure search for protein three-dimensional structure databases on a large-scale, a method has been proposed to solve the existing geometric suffix tree problems. Specifically, the proposed method is consisted of a buffer management method has been proposed for the input and output of data pages containing the nodes of geometric suffix tree capacities, and a hiding sequence method for high-speed similar substructure searches. The buffer management method for the I/O data page controls the data page at a higher level node in order for the tree to be stored in the buffer. The hidden sequence method was added to each of the tree data structure nodes which are called hidden sequences for reduce the number of disk I/O operations for searching solutions. In order to verify the effectiveness of the proposed method, compared with the existing geometric suffix trees, the proposed method implements and measures the speed and accuracy of similar substructure searches. As a result, the proposed method is highly scalable, and can perform similar substructure searches quickly in large-scale data.

Keywords: database, protein, similar structure search

1. はじめに

蛋白質は生命活動の生体機能にかかわる重要な物質であり，生命科学の研究や創薬対象となっている．蛋白質は，アミノ酸配列が異なっていても，立体構造と生体機能の間には密接な関係があることが分かっている．しかしな

[1] 広島市立大学大学院情報科学研究科
Graduate School of Information Science, Hiroshima City University, Hiroshima 731–3194, Japan
[a)] dt65002@edu.ipc.hiroshima-cu.ac.jp
[b)] ktamura@hiroshima-cu.ac.jp
[c)] kuroki@hiroshima-cu.ac.jp
[d)] kitakami@hiroshima-cu.ac.jp

がら，近年，蛋白質の立体構造データが急増してきているため，蛋白質立体構造の類似部分構造を高速に見つけ出す方法の研究がさかんに行われている．その代表的な方法として，サフィックス木を利用した類似構造検索の方法 [1], [2], [3], [4], [5], [6] とフィルタリングなどに基づく高速化法 [7], [8] が提案されている．しかし，フィルタリングに基づく方法では，サフィックス木を用いた場合のように頻出な構造を見つけることができない．このため，サフィックス木による高速化，大規模化の研究が重要である．サフィックス木を利用した方法では，蛋白質立体構造データである座標配列を用いて幾何学的なサフィックス木 [1], [2], [3], [4] を構築する方法と，座標配列を符号化してサフィックス木を構築する PSIST [5], [6] と呼ばれる方法がある．前者の幾何学的なサフィックス木では，2 つの立体構造間の非類似度を求める RMSD（平均二乗偏差）を用いている．RMSD [9], [10], [11], [12], [13], [14] とは 2 つの立体構造に対して最適な重ね合わせを行うことで，2 構造間の非類似度を求める方法である．立体構造を利用することで，そのすべての情報がサフィックス木に組み込まれている．しかし，後者の PSIST は，座標配列を符号化しているため，立体構造の情報が一部失われた状態でサフィックス木が構築されている．このため，本論文では，立体構造のすべての情報が残されている幾何学的なサフィックス木に着目している．しかしながら，従来の幾何学的なサフィックス木には，大規模なデータに対しては，検索時間がかかることなどといった問題がある．これらの問題を解決するために，以下の 2 つの仕組みを提案する．

(1) 大容量の幾何学的サフィックス木のバッファ管理
　大容量の幾何学的サフィックス木のノードが格納されたバッファページのディスクとのやりとりをするバッファ管理の仕組みを実現し，木の上位レベルのノードがバッファ上に残るようにデータページの制御を行う．

(2) 類似構造検索の高速化のための隠れ配列
　問合せに合致した解の探索において，ディスク I/O 回数を低減させるために，隠れ配列と呼ばれるデータ構造を木の各ノードに追加する．

本論文の構成は以下のとおりである．2 章では関連研究として，類似部分構造検索手法について述べる．3 章では，従来手法である幾何学的なサフィックス木について述べる．4 章では，提案手法について述べる．5 章では提案手法の有効性を示すための実験による評価を行い，6 章では本論文のまとめを行う．

2. 関連研究

蛋白質の立体構造を表現する座標配列データベースに対する類似構造検索法の研究では，2 つの座標配列間の類似度あるいは非類似度を計算するために，さまざまな尺度が利用されている．2 つの座標配列が類似しているとは，類似度尺度が大きい値をとるか，あるいは，非類似度尺度が小さい値をとることを意味する．類似構造検索に利用されている尺度には，(1) 座標配列間の RMSD [1], [2], [3], [4], [7], [8], (2) 座標配列の符号化により得られる文字列間のハミング距離 [5], [6]，(3) 動的計画法を 2 回適用して座標配列間の局所的かつ大域的な対応づけ（DDP）により計算されるスコア [15], [16], [17], [18], [19]，(4) 座標配列を表現するグラフ間の対応づけ問題（CMO 問題）を解くことにより得られる辺の最大数 [20], [21], [22] などがある．上述の (1) に関する研究では，高速な類似構造検索のために，フィルタリングまたは索引構造の研究が行われている．また，(2) に関する研究では，(1) と同様の高速な類似構造検索のために，索引構造の研究が行われている．それに対して，(3) および (4) に関する研究では，尺度の計算精度を向上させる手法の研究が行われている．

上記の (1) および (2) の尺度を持つ類似構造検索では，高速化のための索引構造として，いずれもサフィックス木が利用されている．前者の尺度による類似構造検索では，RMSD に基づいて計算される MSSD（最小和二乗距離）[2] も非類似度による尺度として定義する．MSSD とは，最適な重ね合わせを行った 2 つの立体構造の距離の和を求める方法である．立体構造の長さが長くなるとき RMSD の値は，単調増加とならず減少する場合もあるため，2 つの構造間の類似する最大の長さを求めるサフィックス木には適さない．しかし，MSSD では単調増加となるため，サフィックス木に適している．この尺度を用いて構築されたサフィックス木は，幾何学的サフィックス木 [1], [2] と呼ばれている．後者の尺度による類似構造検索は，文字列データを対象にしたサフィックス木構築研究 [23], [24], [25], [26] に帰着しようとする取り組みである．そこで使用される尺度は，蛋白質のアミノ酸配列のバックボーンの構造を符号化することで座標配列を文字列配列に変換し，文字列間のハミング距離を非類似度尺度として定義されている．この尺度が利用されたサフィックス木は，PSIST [5], [6] と呼ばれている．これらのサフィックス木が用いられている研究は，いずれも蛋白質の立体構造検索の高速化をめざしており，本論文では，幾何学的な情報をサフィックス木に直接組み込んだ幾何学的サフィックス木に注目している．

従来の幾何学的なサフィックス木では，275 件の小規模な座標配列データセット（蛋白質の立体構造データ）に対して検索時間の評価が行われているのみ [2] で，実用規模の座標配列データに対する実験が報告されていない．また，検索精度の評価は行われていない．PSIST では，1810 件の座標配列データセット（蛋白質の立体構造データ）に対して検索時間と検索精度の評価が行われている [6] が，実用規模の座標配列データに対する実験が報告されていない．

3. 幾何学的なサフィックス木

従来の幾何学的なサフィックス木 [1], [2] は，RMSD に基づいて計算された MSSD を非類似度の尺度として，文字列データに対するサフィックス木を拡張し，座標配列を扱えるようにしたサフィックス木である．以後，文字列データに対するサフィックス木を文字列のサフィックス木と呼び，幾何学的なサフィックス木と区別する．幾何学的なサフィックス木では，2 つの座標配列の間の MSSD が閾値以下となる構造は同じ構造と見なして 1 本の枝で表されている．本章では，最初にその基本となる文字列のサフィックス木の説明を行った後，RMSD について触れ，従来の幾何学的なサフィックス木の構築法について述べる．

3.1 サフィックス木

文字列のサフィックス木 [27], [28] とは与えられた文字列 S のサフィックスの集合をトライ構造で表現したデータ構造である．文字列 S は，あるアルファベット Σ 上で定義された有限の文字の並びであり，S の終端記号を \$ とするとき，S のサフィックス木は，S\$ のサフィックスをもとに構築されたトライ木である．ただし，$\$ \notin \Sigma$ を満たし，S\$ は S の最右端に終端記号\$を追加した文字列とする．この木の各葉は文字列 S のサフィックス T の 1 つを表現し，根から葉に至るパス上の各ノードは T の部分文字列を表す．これ以降，文字列および座標配列を $\langle\rangle$ で囲んで表記する．

たとえば，文字列 $\langle ababcab\rangle$ に対するサフィックス木について考えてみよう．終端記号として文字列の最右端に \$ を追加する．文字 a で始まるものは $\langle ababcab\$\rangle$，$\langle abcab\$\rangle$，$\langle ab\$\rangle$ の 3 つあり，それらのどれも先頭 2 文字が $\langle ab\rangle$ である．このように，$\langle ab\rangle$ が 3 つの文字列の先頭 2 文字に共通に存在する点を利用し，サフィックス木では，**図 1** に示すように，これらの 3 つの文字列を辺 $\langle ab\rangle$ で表記する．その下に接続される辺は，$\langle abcab\$\rangle$，$\langle cab\$\rangle$，$\langle\$\rangle$ の 3 つ存在するが，これらのどの文字列も先頭から始まる共通部分文字列は存在しない．文字 b で始まるものは，$\langle babcab\$\rangle$，$\langle bcab\$\rangle$，$\langle b\$\rangle$ の 3 つあり，そのすべてが辺 $\langle b\rangle$ のみ共通である．その下に接続される辺は，$\langle abcab\$\rangle$，$\langle cab\$\rangle$，$\langle\$\rangle$ の 3 つ存在しするがこれらの 3 つの文字列の先頭に共通部分はない．文字 c で始まるものは 1 つのみなので，1 本の枝のみで表す．

3.2 RMSD

RMSD [14] とは 2 つの座標配列間の幾何学的な非類似度を決定する手段の 1 つである．RMSD の閾値を ϵ とし，この値以下であれば類似構造とする．3 次元空間において，2 つの座標配列 $P = \langle p_1, p_2, \cdots, p_n\rangle$ と $Q = \langle q_1, q_2, \cdots, q_n\rangle$ を比較してみよう．ただし，この比較では，座標配列の同じ位置にある点座標 $\boldsymbol{p}_i$ と $\boldsymbol{q}_i$ を対応させる．

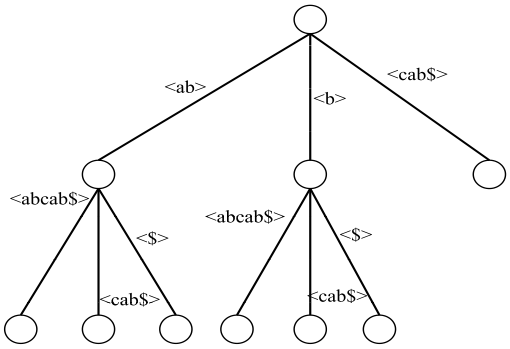

図 1　文字列 $\langle ababcab\$\rangle$ でのサフィックス木の例
Fig. 1　Example of the suffix tree of a string $\langle ababcab\$\rangle$.

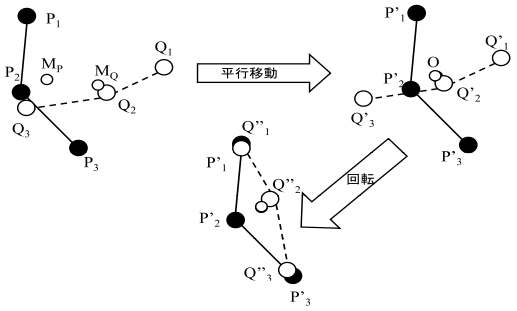

図 2　RMSD の計算例
Fig. 2　Calculation of RMSD.

$E(P,Q,R,v) = \sqrt{\sum_{i=1}^{n}\left|p_i - (R\cdot q_i + v)\right|^2/n}$ とおくと，P と Q の RMSD 値である RMSD(P,Q) は，RMSD$(P,Q) = \min_{R,v}\{E(P,Q,R,v)\}$ であり，$(R_0, v_0) = \mathrm{argmin}_{R,v}E(P,Q,R,v)$ を満たす．また，P と Q の MSSD 値である MSSD(P,Q) は MSSD$(P,Q) = n(\mathrm{RMSD}(P,Q))^2$ $(= n(E(P,Q,R_0,v_0))^2)$ を満たす．

図 2 に RMSD 値の計算イメージを示す．RMSD 値の計算は，2 つの座標配列の重心が同じ座標点になるように平行移動し，R を求めることで達成される．最初に P および Q の重心 M_P，M_Q を原点 O に位置するように座標配列を平行移動させる．$f(R) = \sum_{i=1}^{n}\left|p_i - (R\cdot q_i)\right|^2/n$ を最小化する R を求め Q を回転させることで，距離を最小化した P' と Q' を求めることができる．この R は，特異値分解（SVD）を以下のように使用することによって線形の時間でを見つけることができる．$H = \sum_{i=1}^{n} p_i{}^t q_i$ とし，H の SVD を行うと $U\Lambda{}^tV$ となり，$R = V{}^tU$ を計算することで，$f(R)$ を最小化する R となる．

幾何学的なサフィックス木の構築と検索のために RMSD を使おうとすると，RMSD は単調増加にならないため，構築や検索の枝刈りの閾値とは使えないので，MSSD を使う．

3.3 幾何学的なサフィックス木

幾何学的なサフィックス木とは，座標配列に対するサ

表 1　データセット DS
Table 1　Data set DS.

ID	座標配列
1	$\langle(3,5,2),(4,6,3),(4,5,3),(7,7,4),(4,4,2)\rangle$
2	$\langle(2,5,1),(3,5,3),(2,7,4),(6,7,3),(3,4,2)\rangle$

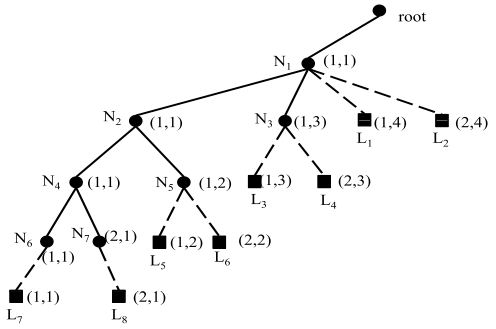

図 3　幾何学的なサフィックス木の構築例
Fig. 3　Example of construction of the geometrical suffix tree.

フィックス木を構築したものである．幾何学的なサフィックス木で同じ枝とする条件は，2 つの座標配列のサフィックスの間の MSSD の値が分割パラメータ b 以下となることである．分割パラメータは，MSSD の閾値である．たとえば，**表 1** の座標配列を用いて，MSSD の閾値を 20 とするとき，**図 3** に示されるような幾何学的サフィックス木が得られる．この図 3 では，幾何学的なサフィックス木の中間ノードを N，葉ノードを L，座標配列のサフィックスを (ID, 参照開始位置) $\in$ DS として表現されている．また，ノードや葉に付与されている (i,j) は，ID が i の座標配列の先頭から j 番目の座標点から最右端までの部分座標配列を意味する．たとえば，L_6 に付与されている $(2,2)$ は，$\langle(3,5,3),(2,7,4),(6,7,3),(3,4,2)\rangle$ であることを意味する．検索キーとの RMSD の値を d，RMSD の閾値を ϵ としたときの検索処理手順を以下に示す．この検索では (2) で木の探索を行い中間ノード N_l を見つけ，(3) で類似部分構造を探している．

(1) 分岐パラメータ b で幾何学的なサフィックス木を構築．
(2) 根から木をたどり $depth(N_k) < m$，$depth(N_l) \geq m$ かつ MSSD $\leq m \cdot (d + \sqrt{b/m})^2$ となる枝 $e = (N_k, N_l)$ を見つける．
(3) 中間ノード N_l 以下の葉ノードをすべて調べ，RMSD が ϵ 以下の部分構造を見つける．

従来の幾何学的なサフィックス木には 2 つの問題点がある．1 つ目は実用規模のデータを扱うための方法が提案されていないこと，2 つ目は類似構造検索を行ったとき，検索結果を得るためには木の探索が必要となることである．たとえば，図 3 の例に対して長さ 3 の検索キー $Q = \langle(2,4,1),(5,5,3),(4,7,5)\rangle$，RMSD の閾値 $\epsilon = 2.0$ で検索を行ってみよう．このとき N_2 まで探索を行うと，部分構造の情報は，N_2 以下の葉ノード L_5，L_6，L_7，L_8 に含まれているが，L_5，L_6 を調べるために N_5 の探索を行い，L_7，L_8 を調べるために N_4，N_6，N_7 の探索を行わなければならない．次章では，これらの問題点を解決する方法を提案する．

4. 提案手法

本章では，木を改良して高速な検索を行うために，従来の幾何学的なサフィックス木の 2 つの問題点を解決する方法について提案する．

サフィックス木を構築すると木のデータが元データの長さ n の 14 倍で構築する方法 [29] が提案されている．また，さらに，検索精度は低下するが，サフィックス木のサイズをさらに小さくした圧縮サフィックス木 [30] という実装方法もある．しかし，幾何学的なサフィックス木については，木のデータを小さくするための研究は行われておらず，今後の課題として残されている．我々の幾何学的なサフィックス木のデータの大きさを考えてみよう．元データの長さを n とすると，中間ノードの数は $n+1$ から $2n-1$，葉ノードは $n-1$ となる．我々が作成した幾何学的なサフィックス木では，1 座標が 24 バイト，中間ノードが 128 バイト，葉ノードが 64 バイトとしている．よって，中間ノードが 6 倍，葉ノードが 2.6 倍で表すことができる．よって，木のサイズは元データの 8.6 倍から 15.6 倍となる．

4.1 システム構成

大規模のデータを扱うためのシステムの構成の説明をする．大規模な文字列のサフィックス木を構築するために，ディスク上に文字列のサフィックス木を構築したディスクベースの文字列のサフィックス木の研究が行われている．ディスクベースの文字列のサフィックス木は，ノードがディスクのページに格納されていて，アクセスがあるとバッファを通して読み出される．このことは幾何学的なサフィックス木も同様で，wwPDB に登録されている蛋白質データは指数関数的に増加しているため，ディスクベースの幾何学的なサフィックス木を構築する必要がある．本研究では，幾何学的なサフィックス木のデータをデータページ，データページを管理するためのページテーブルを格納したページをカタログページと呼ぶ．ディスクベースの幾何学的なサフィックス木のシステム構成図を**図 4** で示す．点線で囲まれた範囲がバッファ管理システムである．提案するバッファ管理は，DPM（データページ管理モジュール）と CPM（カタログページ管理モジュール）の 2 つのモジュールから構成される．DPM はメモリ上のデータページのアドレスの管理とディスクとの読み書き，CPM はメモリ上のカタログページのアドレスの管理とディスクとの読み書きを行う．

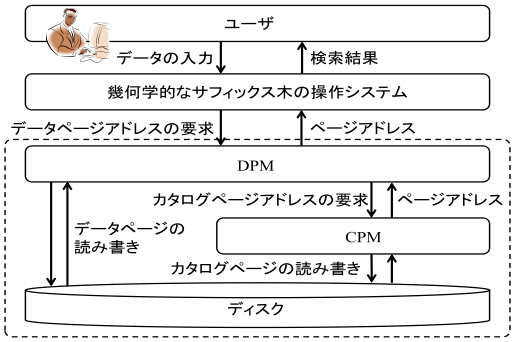

図 4　ディスクベースの幾何学的なサフィックス木のシステム構成
Fig. 4　System configuration of the disk-based geometrical suffix tree.

4.2　隠れ配列法

幾何学的なサフィックス木の検索における処理を削減するために隠れ配列の提案を行う．前章で説明したように，幾何学的なサフィックス木の検索では検索キーの長さに対応する深さまで木の探索を行っても，さらに深い葉ノードまで探索を行わなければならない．これを解決するために，ある中間ノード以下に存在するすべての葉ノードの情報をその中間ノードに持たせた隠れ配列と呼ばれるデータ構造を提案する．図 3 の N_2 以下の葉ノードを調べると，N_2 には $\{(1,1),(1,2),(2,1),(2,2)\}$ が含まれることが分かるが，N_2 自体は $(1,1)$ しか存在しない．すべての部分構造を調べるためには 4 つの中間ノードと 4 つの葉ノードの探索が必要となり，子の探索には時間がかかる．これを解決するために，$\{(1,2),(2,1),(2,2)\}$ を N_2 の前面に出ない隠れた部分構造として保持する．これを隠れ配列と呼んでいる．隠れ配列は 3.3 節で用いた座標配列のサフィックスを表す (ID, 参照開始位置) をページ内に書き込む．書き込みは構築時に中間ノードを通過するときに行われ，枝を分割するときには子ノードの隠れ配列を新しいノードにコピーする．ページサイズより大きくなる場合には，次ページと最終ページのアドレスを各ページに持たせる．そのため，ページ内にデータが数件しか入っていない場合がある．このとき，隠れ配列ページのバッファ領域がいっぱいになっていても，多くのページの中には空きが多くあるという状態になる．すると，バッファ領域へのページの読み込み，バッファ領域のページの削除が頻繁に起こることになるという影響を及ぼす．これを用いた検索処理手順を以下に示す．

(1) 分岐パラメータ b で幾何学的なサフィックス木を構築．
(2) 根から木をたどり $depth(N_k) < m$，$depth(N_l) \geq m$ かつ MSSD $\leq m \cdot (d + \sqrt{b/m})^2$ となる枝 $e = (N_k, N_l)$ を見つける．
(3) 中間ノード N_l の隠れ配列を調べ，RMSD が ϵ 以下の部分構造を見つける．

表 2　各ページのバッファ管理方式
Table 2　Buffer management scheme of each page.

ページ名	バッファ管理方式	構築	検索
中間ノードページ	TOP-Q	○	○
葉ノードページ	LRU	○	-
隠れ配列ページ	LRU	○	-
カタログページ	LRU	○	○

4.3　バッファ管理法

4.1 節で提案したバッファ管理法について説明する．データページとは中間ノード，葉ノード，隠れ配列を格納したページである．本研究では．幾何学的なサフィックス木のデータである中間ノード，葉ノード，隠れ配列のすべてをディスク上に格納しながら幾何学的なサフィックス木の構築，検索を行う．ここで用いる各ページのバッファ管理方式を**表 2** に示す．LRU とは，バッファ領域のページの優先度を参照された時間で管理する方法である．この方法はバッファ領域に空きがないとき，バッファ領域のページの中で参照された時間が最も古いページを削除する．TOP-Q とは，バッファ領域のページの管理を二分ヒープで行う TOP と，TOP から削除されたページを持つキューを組み合わせた方法である．この方法は，幾何学的なサフィックス木の深さを優先度として持たせることで，幾何学的なサフィックス木の浅いデータをバッファ領域に優先的に確保できる．

(1) 葉ノードページ
構築では，書き込み途中のページを管理するために LRU を使用する．検索では，提案手法では葉ノードを用いない．
(2) 隠れ配列ページ
構築では，隠れ配列は中間ノードごとに書き込み済みのページと書き込み途中のページが存在するため，書き込み途中のページを管理するために LRU を使用する．検索では，中間ノードごとにそれぞれの隠れ配列を持つため，1 度読み込んで検索に使用した隠れ配列ページを再度検索に使用することはないので，使用後は削除する．このため，バッファ管理は行わない．
(3) カタログページ
カタログページは，1 度使用したページをメモリ上に残すために LRU を使用する．
(4) 中間ノードページ
このバッファ管理方式は構築と検索で同じ方式を使用している．中間ノードは幾何学的なサフィックス木の構築と検索で，ルートノードに近いノードの参照が多くなるので，幾何学的なサフィックス木のノードの深さが浅いノードをバッファに残す必要がある．ここでは，TOP-Q [31] と呼ばれるバッファリング戦略を使用する．

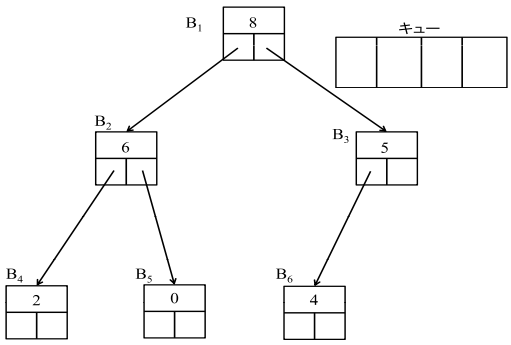

図 5　中間ノードページのバッファ管理例
Fig. 5　Examples of buffer management of an internal node page.

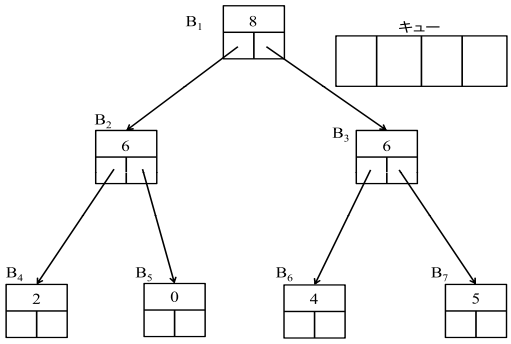

図 6　$m_B > m_U$ のときの新しいページの追加
Fig. 6　Adding a new page when $m_B > m_U$.

中間ノードで用いる TOP-Q については，**図 5** を用いて説明する．TOP-Q は 2 分木のデータ構造で，各ノードには最大 2 つの子ノードを持つ．また，各ノードには中間ノードのページを持ち，各ページに含まれる中間ノードの経路の長さの平均値のスコアを持ち，バッファの木は親ノードのスコアが子ノードより大きくなる．さらに，2 分木から除かれたページを一時的にメモリ上に残すためのキュー構造も持つ．また，ルートノードを持つページについてはスコアを 0 に設定する．

最大バッファサイズ $m_B = 7$，使用バッファサイズ $m_U = 6$ である図 5 に対して，スコアが 6 の新しいページを追加すると，新しいバッファの木のノード B_7 ができ，親ノードである B_3 と比較して交換される．この結果を**図 6** で示す．

さらに，図 6 に対してスコアが 3 の新しいページを追加すると，バッファ領域に空きがないため，ルートノード B_1 のページがキューに書き込まれる．新しいページはルートノード B_1 に格納される．木を最適な状態に保つために，子ノード B_2 と比較され B_1 より大きい B_2 と交換される．この結果を**図 7** に示す．

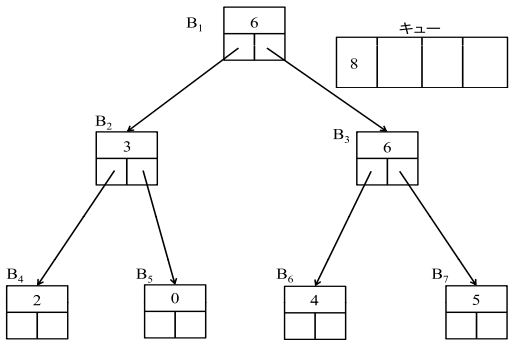

図 7　$m_B = m_U$ のときの新しいページの追加
Fig. 7　Adding a new page when $m_B = m_U$.

表 3　検索キー
Table 3　Query sequence.

アミノ酸モチーフ	PDB ID
kringle	1A0H
homeobox	1AU7
leucine zipper	2ZTA

5. 評価実験

本章では新しく前章で提案した幾何学的なサフィックス木について評価を行う．

5.1 実験方法

提案手法の評価を行うために，wwPDB に 2012 年 1 月 11 日時点で登録されていた PDBML 形式の座標配列データファイルを，PDB の ID に含まれる鎖ごとにアミノ酸配列の Cα 原子の座標データを取り出し，評価実験データに使用した．PDB から取得した実験データは 366,146 件，データサイズは 1,400 MB である．評価実験では，検索スピードのスケーラビリティを検証するために，オリジナルの実験データを 6,400 MB になるまで複製して利用した．検索キーに使用したデータは**表 3** に示す．分岐パラメータは $b = 400$，検索パラメータは $\epsilon = 2.5$，ページサイズは 8 KB，1,400 MB のデータで構築した幾何学的なサフィックス木の各データサイズとバッファサイズを**表 4** に示す．検索バッファサイズの '-' は，バッファを用いないことを示す．実験環境は CPU：3.06 GHz × 12 コア，メモリ：72 GB，ディスク容量：38 TB である．提案手法の優位性を確認するために従来手法に対し比較を行う．

5.2 評価

ここでは，まず，評価のための実験環境が正しく構築されていることを確認するために，我々が作成したシステムの構築性能と検索精度を調べる．前者については，構築時間を調べることで幾何学的なサフィックス木が $O(n)$ で構築できていることを確かめる．後者については，検索精度

表 4　バッファサイズ
Table 4　Buffer size.

ページ名	データサイズ	構築バッファサイズ	検索バッファサイズ
中間ノードページ	78 GB	10 GB	10 GB
葉ノードページ	1.8 GB	1 GB	-
隠れ配列ページ	105 GB	10 GB	-
カタログページ	185 MB	25 MB	10 MB

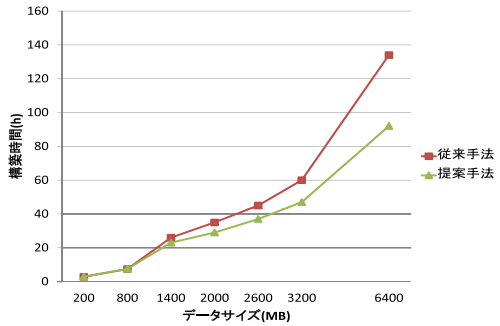

図 8　幾何学的なサフィックス木の構築の CPU 時間
Fig. 8　CPU time of construction of the geometrical suffix tree.

検索方法 ＼ データサイズ (MB)	200	800	1400	2000	2600	3200	6400
従来手法 (kringle)	1	1	1	1	1	1	1
従来手法 (homeobox)	1	1	1	1	1	1	1
従来手法 (leucine zipper)	1	1	1	1	1	1	1
提案手法 (kringle)	1	1	1	1	1	1	1
提案手法 (homeobox)	1	1	1	1	1	1	1
提案手法 (leucine zipper)	1	1	1	1	1	1	1

図 9　検索精度（F-値）
Fig. 9　Retrieval accuracy (F-measure).

を調べることで実用的な検索ができていることを確かめる．以上を確認した後に，検索速度を調べることで，提案手法が従来手法より高速であることを確かめる．

5.2.1　構築性能

ここでは幾何学的なサフィックス木を構築するための時間を従来手法と提案手法それぞれで調べる．従来の幾何学的なサフィックス木の構築と条件を合わせるために，提案手法では隠れ配列を含まない構築時間の CPU 時間の計測を行った．その結果，**図 8** のとおり構築時間が n に関して，ほぼ線形時間，すなわち，$O(n)$ となった．このことから，文献 [2] と同様に，データサイズと構築時間はデータサイズ n に関して線形オーダであることを確認した．

データサイズが 1,400 MB 以下と 1,400 MB 以上でデータの傾きが異なる結果となっているが，これは実験データによる結果である．1,400 MB のデータで構築した幾何学的なサフィックス木と 1,400 MB 以上のデータで構築した幾何学的なサフィックス木では，新たに中間ノードを作成することがなく葉ノードの件数の増加とそれぞれの中間ノードでの隠れ配列のサイズの増加だけが起こる．よって，中間ノードのデータの新規作成および更新にかかる処理の分だけ構築時間が減少している．

5.2.2　検索精度

ここでは，5.2.1 項で構築した幾何学的なサフィックス木に対して検索を行った．検索結果の正解は構築に用いたデータすべてに対して RMSD を計算し，閾値 ϵ 以下であるものとする．F 値を用いて検索精度を調査した．その結果，**図 9** のとおり検索精度の確保ができた．

従来手法と同等かつ検索もれやノイズのない検索ができていることから，実用に足る検索ができているといえる．

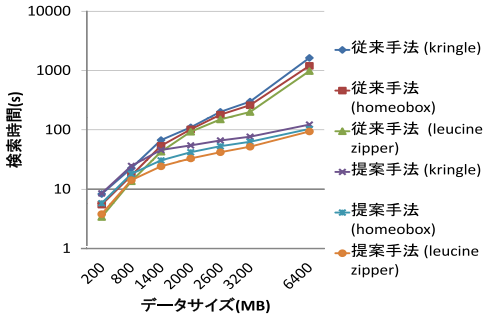

図 10　検索時間
Fig. 10　Retrieval time.

実験手法 ＼ データサイズ (MB)		200	800	1400	2000	2600	3200	6400
従来手法	構築時間(h)	2.8	7.4	26	35	45	60	134
	検索時間 (kringle)(s)	8.3	22.6	67.2	110.2	201	299.3	1633.1
	検索時間 (homeobox)(s)	5.5	17	53.8	102	180	260	1200
	検索時間 (leucine zipper) (s)	3.4	13.6	42.8	93	150	200	980
提案手法	構築時間(h)	2.9	7.5	23	29	37	47	92
	検索時間 (kringle) (s)	8.5	24.7	45.8	54.7	66.2	76.6	122.7
	検索時間 (homeobox) (s)	5.8	18.1	30.6	42	53	63	103.7
	検索時間 (leucine zipper) (s)	3.8	14.2	24.3	33	42	52	94

図 11　評価実験にかかる時間
Fig. 11　Experiment time.

5.2.3　検索時間

検索時間の実験結果を**図 10** に，構築時間と検索時間の内訳を**図 11** に示す．データサイズが大きくなるにつれて，従来手法の検索時間が長くなっている．これは，表 4 で

示されているとおり実験環境のメモリサイズを超え，データの読み込みに時間がかかっているからだと考えられる．データサイズを 6,400 MB まで大きくすると，各検索キーで提案手法が従来手法に比べ 10 倍以上の高速化を達成した．

検索では，検索キーの長さと等しい幾何学的なサフィックス木の深さである中間ノードまで探索を行うが，従来手法では検索結果の候補である部分構造を列挙するためにその中間ノードからさらに葉ノードまでの探索を行わなければならない．しかし，提案手法では隠れ配列を用いることでその中間ノードにある隠れ配列を調べることで，検索結果の候補である部分構造を列挙することができる．その結果，従来手法に比べて高速な検索ができていると考えられる．

6. おわりに

本研究では，幾何学的なサフィックス木 2 つの問題点を解決する方法を提案した．提案手法は，大容量の幾何学的サフィックス木のノードが格納されたデータページのやりとりをするバッファ管理法，類似構造検索の高速化ための隠れ配列法より構成されている．提案手法の評価を行うために，wwPDB からアミノ酸で構成された蛋白質の全座標配列を取得し，類似部分構造検索実験を行った．評価実験の結果，高精度性を維持しつつ，大規模なデータに対して高速な類似部分構造検索ができることを確認した．

今後の課題として，ページへのデータの格納法の改良によるさらなる高速な検索や幾何学的なサフィックス木のサイズを小さくする方法などがあげられる．

謝辞　本研究の一部は，日本学術振興会・科学研究費補助金（基盤研究（C），課題番号：20500137）の支援により行われた．

参考文献

[1] Shibuya, T.: Geometric Suffix Tree: A New Index Structure for Protein 3-D Structures, *Combinatorial Pattern Matching 2006*, LNCS 4009, pp.84–93 (2006).
[2] Shibuya, T.: Geometric Suffix Tree: Indexing Protein 3-D Structures, *J. ACM*, Vol.57, No.3, pp.15:1–15:17 (2010).
[3] 高橋誉文，黒木　進，田村慶一，北上　始：複数の蛋白質立体構造データに対するディスク版索引構造の構築方式，データ工学と情報マネジメントに関するフォーラム DEIM2009，電子情報通信学会データ工学研究専門委員会，p.8 (2009).
[4] Takahashi, Y., Kuroki, S. and Kitakami, H.: Efficient Query Processing in Protein Structure Databases, *Proc. 2nd International Workshop with Mentors on Databases, Web and Information Management for Young Researchers* (*iDB Workshop 2010*), pp.47–55 (2010).
[5] Gao, F. and Zaki, M.J.: PSIST: Indexing Protein Structures Using Suffix Trees, *Proc. IEEE Comput. Syst. Bioinform. Conf.*, pp.212–222 (2005).
[6] Gao, F. and Zaki, M.J.: PSIST: A Scalable Approach to Indexing Protein Structures Using Suffix Trees, *J. Parallel Distrib. Comput.*, Vol.68, No.1, pp.54–63 (2008).
[7] Shibuya, T., Jansson, J. and Sadakane, K.: Linear-time Protein 3-D Structure searching with Insertions and Deletions, *Algorithms Mol. Biol.*, Vol.5:7, p.8 (2010).
[8] Terashi, G., Shibuya, T. and Takeda-Shitaka, M.: LB3D: A Protein Three-dimensional Substructure Search Program Based on the Lower Bound of a Root Mean Square Deviation Value, *J. Comput. Biol.*, Vol.19, No.5, pp.493–503 (2012).
[9] Arun, K.S., Huang, T.S. and Blostein, S.D.: Least-Squares Fitting of Two 3-D Point Sets, *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.9, No.5, pp.698–700 (1987).
[10] Eggert, D.W., Lorusso, A. and Fisher, R.B.: Estimating 3-D Rigid Body Transformations: A Comparison of Four Major Algorithms, *Mach. Vision Appl.*, Vol.9, No.5-6, pp.272–290 (1997).
[11] Kabsch, W.: A Solution for the Best Rotation to Relate Two Sets of Vectors, *Acta Crystallographica Section A*, Vol.32, No.5, pp.922–923 (1976).
[12] Kabsch, W.: A Discussion of the Solution for the Best Rotation to Relate Two Sets of Vectors, *Acta Crystallographica Section A*, Vol.34, No.5, pp.827–828 (1987).
[13] Schwartz, J.T. and Sharir, M.: Identification of Partially Obscured Objects in Two and Three Dimensions by Matching Noisy Characteristic, *Int. J. Rob. Res.*, Vol.6, No.2, pp.29–44 (1987).
[14] Umeyama, S.: Least-Squares Estimation of Transformation Parameters Between Two Point Patterns, *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.13, No.4, pp.376–380 (1991).
[15] Toh, H.: Introduction of a Distance Cut-off into Structural Alignment by the Double Dynamic Programming Algorithm, *Computer Applications in the Biosciences*, Vol.13, No.4, pp.387–396 (1997).
[16] Standley, D.M., Toh, H. and Nakamura, H.: GASH: An Improved Algorithm for Maximizing the Number of Equivalent Residues between Two Protein Structures, *BMC Bioinformatics*, Vol.6, pp.221–239 (2005).
[17] 日本蛋白質構造データバンク（PDBj: Protein Data Bank Japan），入手先 ⟨http://www.pdbj.org/⟩.
[18] 三村賢太，北上　始，森　康真，高橋誉文：蛋白質立体構造データベースに対する類似構造検索性能の考察，第 14 回 IEEE 広島支部学生シンポジウム（HISS），pp.529–532 (2012).
[19] 三村賢太，北上　始，森　康真，高橋誉文：蛋白質立体構造検索のための類似度計算性能に関する考察，第 63 回電気・情報関連学会中国支部連合大会，pp.448–449 (2012).
[20] Godzik, A. and Skolnick, J.: Flexible Algorithm for Direct Multiple Alignment of Protein Structures and Sequences, *Computer Applications in the Biosciences*, Vol.10, No.6, pp.587–596 (1994).
[21] Andonov, R., Malod-Dognin, N. and Yanev, N.: Maximum Contact Map Overlap Revisited, *Journal of Computational Biology*, Vol.18, No.1 pp.1–15 (2011).
[22] 中田章宏，田村慶一，北上　始，高橋誉文：CMO 問題に対する改良版 EO を用いた発見的解法，情報処理学会論文誌 数理モデル化と応用，Vol.6, No.3, p.12 (2013).
[23] Ching-Fung, C., Jeffrey, X.Y. and Hongjun L.: Constructing Suffix Tree for Gigabyte Sequences with Megabyte Memory, *IEEE Trans. Knowledge and Data Engineering*, Vol.17, No.1, pp.90–105 (2005).
[24] Tian, Y., Tata, S., Hankins, R.A. and Patel, J.M.: Practical Methods for Constructing Suffix Trees, *VLDB*

Journal, Vol.14, No.3, pp.281–299 (2005).
[25] Benjarath, P. and Mohammed J.Z.: TRELLIS+: An Effective Approach for Indexing Genome-Scale Sequences Using Suffix Trees, *Pacific Symposium on Biocomputing*, pp.90–101 (2008).
[26] Watanuki, Y., Tamura, K., Kitakami, H. and Takahashi, Y.: Parallel Processing of Approximate Sequence Matching Using Disk-based Suffix Tree on Multi-core CPU, *IWCIA2013, IEEE SMC Hiroshima Chapter*, p.6 (2013).
[27] McCreight and Edward, M.: A Space-Economical Suffix Tree Construction Algorithm, *J. ACM*, Vol.23, No.2, pp.262–272 (1976).
[28] Weiner, P.: Linear Pattern Matching Algorithms, *Proc. 14th Annual Symposium on Switching and Automata Theory* (*SWAT 1973*), *SWAT '73*, pp.1–11 (1973).
[29] Stefan, K.: Reducing the Space Requirement of Suffix Trees, *Software – Practice and Experience*, Vol.29, pp.1149–1171 (1999).
[30] Sadakane, K.: Compressed Text Databases with Efficient Query Algorithms Based on the Compressed Suffix Array, *Proc. ISAAC'00*, number 1969 in LNCS, pp.410–421 (2000).
[31] Bedathur, S.J. and Haritsa, J.R.: Engineering a Fast Online Persistent Suffix Tree Construction, *ICDE '04 Proc. 20th International Conference on Data Engineering*, Vol.29, pp.720–731 (2004).

高橋 誉文　（学生会員）

2008 年広島市立大学情報科学部知能情報システム工学科卒業．2010 年同大学大学院情報科学研究科博士前期課程修了．同年同博士後期課程進学．日本データベース学会会員．

田村 慶一　（正会員）

広島市立大学情報科学研究科准教授．博士（情報科学）．1998 年九州大学工学部情報工学科卒業．2000 年同大学大学院システム情報科学研究科知能システム学専攻修士課程修了．2003 年同大学院システム情報科学府知能システム学専攻博士後期課程単位取得のうえ満期退学．広島市立大学情報科学部助手，広島市立大学大学院情報科学研究科助教，同講師を経て，2011 年より現職．データマイニングとその並列処理，進化計算に関する研究に従事．IEEE（SMCHiroshima Chapter Secretary），日本データベース学会，人工知能学会，日本知能情報ファジイ学会各会員．

黒木 進　（正会員）

広島市立大学大学院情報科学研究科准教授．博士（工学）．1990 年東京大学大学院工学系研究科博士前期課程修了．九州大学助手，広島市立大学助教授を経て 2007 年より現職．主に時空間データ検索の研究に従事．ACM，IEEE，電子情報通信学会，日本データベース学会，人工知能学会等各会員．

北上 始　（正会員）

広島市立大学情報科学研究科教授．博士（工学）．1976 年東北大学大学院工学研究科博士前期課程修了．同年富士通株式会社，以後，富士通研究所，新生代コンピュータ技術開発機構（ICOT）主任研究員，国立遺伝学研究所客員助教授を経て，1994 年広島市立大学情報科学部教授，2007 年より現職．主な著書「データベースと知識発見」（共著），「生命情報学」（共著）ほか．1985 年情報処理学会 25 周年記念論文賞，2003 年日本工学教育協会論文・論説賞ほか．IEEE，ACM，情報処理学会（論文誌数理モデルと応用編集委員，一般情報教育委員会委員），電子情報通信学会，人工知能学会（評議員），日本バイオインフォマティクス学会，日本データベース学会（論文誌編集委員）各会員．

（担当編集委員　宝珍 輝尚）