アミノ酸配列のマルチプルアライメント計算における A*アルゴリズムの適用の効果

十時 泰[†] 秋山 泰[††] 鬼塚 健太郎[††]
野口 保[††] 斎藤 稔[††] 安藤 誠[††]

アミノ酸配列のマルチプルアライメント計算に、最良優先探索の反復改善法を適用し、最良優先探索における近傍探索をPVMライブラリを用いて並列実装した。さらに、A*アルゴリズムを適用して探索空間の効率的な刈り込みを実現した。これらの改良の結果、従来法よりも精度のよい、生物学的にも十分意味のあるマルチプルアライメントを、現実的な計算時間内で得ることを可能とした。

Employing A* Algorithm in Parallel Multiple Protein Sequence Alignment

Yasushi Totoki,[†] Yutaka Akiyama,[††] Kentaro Onizuka,[††]
Tamotsu Noguchi,[††] Minoru Saito[††] and Makoto Ando[††]

As a new strategy for multiple sequence alignment, we employed an iterative scheme with best-first search, and parallelized its search step using PVM library. Furthermore we implemented the A* pruning algorithm instead of dynamic programming, to drastically reduce the search space. As a result, our new parallel system enables biologically accurate multiple sequence alignment performed within reasonable time.

1. はじめに

マルチプルアライメントは、DNA／タンパク質の複数の配列の類似する部分を縦に揃えて並べ合わせる操作[1)2)]で、DNA／タンパク質配列の相同性解析の基本的手法として、分子生物学の様々な分野で利用される重要な処理である。マルチプルアライメントの結果からDNA／タンパク質配列の“保存領域”が同定され、DNA／タンパク質の未知の機能や構造を既知情報から予測することが可能となる。また、進化系統樹もマルチプルアライメントに基づいて作成される。DNA／タンパク質の機能・構造の研究や進化系統樹の研究には、精度の良いマルチプルアライメントが必要とされている。

従来、マルチプルアライメントの作業は、生物学者がエディタソフトなどを用いて手作業で行なうことが多く、その編集作業には、数週間から1カ月かかると言われており、非常に手間と時間のかかる作業である。そのため、その手間と時間を節約するための自動化が望まれているが、いまだに高精度かつ高速なシステムは存在せず、近似的な解法で解かれたマルチプルアライメントから、生物学者がさらに手間と時間をかけて編集を行なっているのが現状である。

実用的なマルチプルアライメントの課題を計算機で解くには、膨大な計算量を必要とするために、アライメントの精度と計算時間との間にトレードオフの関係が存在している。これまでに、種々の戦略に基づく方法が提案されてきた。現在、実用的なものとしては、GCGパッケージのPileUp[3)]やEMBLで開発されたClustalW[4)]を代表とする、マルチプルアライメントをペアワイズアライメントの組合せとして近似する「ツリーベース法」[3)4)5)](progressive method:累進法)が一般に用いられているが、その精度には限界がある。

広沢と十時ら[6)]は、グループ間でのペアワイズアライメントを繰り返して行なう「反復改善法」[7)8)]に注目し、上記の「ツリーベース法」と「反復改善法」を組み合わせた。「反復改善法」は、Bergerら[7)]が提案し、その後、後藤[8)]が、ギャップコスト計算を忠実に行う方法で再提案したものである。

我々は、反復改善法に、解空間の近傍解の中から最も良い解を選ぶ最良優先探索[1)]を適用し、この近傍探索を並列に行なわせることとして、PVMライブラリを用いて実装した。その結果、従来法よりも精度のよい、生物学的にも十分意味のあるマルチプルアライメントを現実的な計算時間内で得ることを可能とした。

† (株)情報数理研究所
Information and Mathematical Science Laboratory, Inc.
†† 新情報処理開発機構
Real World Computing Partnership

さらに、グループ間アライメントを求める方法として、従来の全探索のダイナミックプログラミングに代わり、A*アルゴリズム[10]を適用して探索空間の効率的な刈り込みを行なった。その結果、解の精度を落すことなく、さらなる計算時間の短縮に成功した。

2. ツリーベース並列反復改善法

2.1 マルチプルアライメント

遺伝情報は細胞の核にある染色体のDNAに格納されている。タンパク質は、このDNAの情報から翻訳合成されるアミノ酸配列が、空間的に折れ畳まって特異的な形状をとったものである。タンパク質は生物の体を形成し、生命の代謝反応を司る重要な物質である。

タンパク質の構成要素であるアミノ酸には通常20種類あり、それぞれ異なるアルファベットが割り当てられている。アミノ酸には、大きさ、水との親和性、酸性／塩基性、極性などの性質があり、どのような性質のアミノ酸がどのような順番に連なっているかで、タンパク質の構造と機能が決まる。タンパク質には大小さまざまなものがあり、短いものは数十残基、長いものは約千残基のアミノ酸が連なっている。タンパク質の構造や機能は、実験を積み重ねて初めてわかるものであるが、一方、そのアミノ酸配列のみを調べる技術は、すでに確立されており、現在では約6万種類のタンパク質の配列が決定されている。このように多くのタンパク質の配列データが集まると、新たな可能性が見えてきた。未知のタンパク質であっても、それと類似の配列をもつタンパク質の構造や機能が既知であれば、それから未知の構造や機能を推測することが可能になる。そこで必要になるのは、配列間の類似性を解析する情報処理技術である。

もっとも基本的な配列解析のひとつが、複数の配列の類似する部分を縦に揃えて並べ合わせる操作で、マルチプルアライメント (Multiple Alignment) である。たとえば以下のようなタンパク質のアミノ酸配列が4本あったとする。

```
CCHU    DVEKGKIFIMKCSQCHTVEKGGKHKTGPNL
CCFS    TTGAKIFKTKCAQCHTVKGHKQG
CCZP    DEKKGASLFKTAQCHTVEKGGANKVGPNL
CCCRCF  DAARGEKLRAAQCHTANQGGANGVG
```

ここで、左側の見出しが配列の名前で、右側の文字列がアミノ酸配列を表現する。最上段の配列の左端からDVEKは、それぞれアスパラギン酸、バリン、グルタミン酸、リシン残基を意味する。これら4本の配列をアライメントすると、次のようになる。

```
CCHU    DVEKG-KIFIMKCSQCHTVEKGGKHKTGPNL
CCFS    --TTGAKIFKTKCAQCHTV-KG--HKQG---
CCZP    DEKKGASLFKT--AQCHTVEKGGANKVGPNL
CCCRCF  DAARGEKL---RAAQCHTANQGGANGVG---
        ....G.........QCHT...G.....G...
```

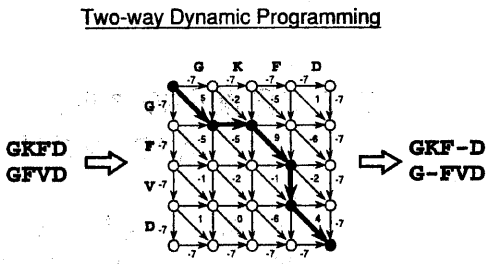

図1　2次元のダイナミックプログラミング[14]

配列のところどころにギャップ "-" を挿入することで、QCHTなどの共通文字が同じ列に並べられた。QCHTのように複数の配列で共通な部分文字列を "配列モチーフ" と呼び、タンパク質の重要な部分を指し示していると考えられる。この背景には、タンパク質の配列のうち生理的に重要な部分には遺伝的変異が蓄積しにくいという進化論的考え方[11]がある。

実際のマルチプルアライメントでは、ひとつの列に同一の文字が揃うことは少なく、異なる文字でもそれらが表すアミノ酸の性質が似ていれば同じ列に置くことを許容して処理を行う。しかしアミノ酸の性質には親水性／疎水性、極性、酸性／塩基性、大きさなど複数あり、その類似性評価も多数の方法がある。現在最も広く使われている類似性評価尺度は、BLOSUM[12]やPAM[13]と呼ばれるマトリクスで、当時知られていたアライメントをもれなく調べ、同じ列に該当アミノ酸残基対が並んでいることが偶然に対していかに少ないかを数値化したものである。数値は確率の対数値になっているため、それらの足し算は複合事象の共起確率を算出したことに相当する。本論文のマルチプルアライメントシステムも、この評価尺度を使用している。この評価尺度を基に、マルチプルアライメントのスコアは、配列のすべての組合せの、すべてのアミノ酸残基ペアの評価値の総和 (sum of pairs) として与えられる。アミノ酸残基同士のペアの評価値は評価尺度マトリクスから算出されるが、残基とギャップとのペアの評価値はギャップコスト関数から算出する。

BLOSUMやPAMマトリックスのような評価尺度が与えられれば、その尺度において最適なマルチプルアライメントを求めるダイナミックプログラミング法[14] (Dynamic Programming: 以後DP)(図1は2本の場合の例) が知られている。N本の配列のマルチプルアライメントはN次元のDPで原理的には解決できる。しかし、N次元のDPは、配列の最大長をLとすると、$O(L^N)$の計算時間とメモリ空間を必要とするので、実用規模の問題 ($L = 100 \sim 1000, N = 10 \sim 100$) では計算量が爆発して、現実には計算不可能である。そこで、実用的には従来から近似的な解法がとられてきた。代表的な近似解法は、マルチプルアライメントをペアワイズアライメントの組合せとして近似したツ

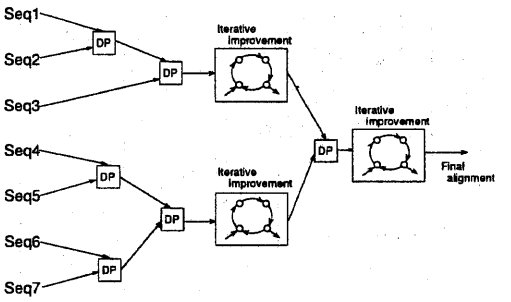

図2　ツリーベース法と反復改善法の融合[6]

リーベース法[3)4)5)]である。

2.2　ツリーベース法

ツリーベース法[3)4)5)]は、まず配列間の類似性に従ってクラスター分析を行ない、配列間のツリー状の階層関係を求める。ツリー状の階層関係に基づいて、類似性の高い配列から順にペアワイズアライメントを行なって、得られたアライメントを組み合わせてマルチプルアライメントを形成していく。類似度の高い配列から順に組合せれば、得られるアライメントの精度を大きく改善できる。類似した配列同士のアライメントは確実で，信頼性が高いからである。しかし、ツリーベース法では、一度できたアライメントは最後まで残り、後で修正されることがない。特に、比較される配列の類似性が低いと、初期段階の誤りが増幅される傾向があり、解の精度は必ずしも十分ではない。この欠点に対処したのが反復改善法[7)8)]である。

2.3　反復改善法

反復改善法[7)8)]は一種の山登り探索で、複数配列をランダムに2つのグループに分け、各グループを平均化した"プロファイル"をあたかも一本の配列の様に扱って、グループ間のペアワイズアライメントで最適化し、再結合してまた新たなランダム分割を行なう手続きを、評価値(アライメントのスコア)が十分に低下するまで何度も繰り返す。ツリーベース反復改善法[6)]では、ツリーベース法で得たマルチプルアライメントに反復改善法を適用することにより、初期段階の誤ったアライメントが修正され、アライメントの精度を大幅に改善できる(図2)。

しかし試行錯誤的な山登り探索による反復改善法は、実用規模の大きな問題になると膨大な数の改善サイクルを要し、計算時間が障害となる。N本の配列からなる配列グループの分割の方法は$2^{N-1}-1$通りであるから、20本以上の配列をアライメントしようとすると、分割法は50万通り以上にのぼる。特に、アライメント過程の後半になると、ほとんどの分割が改善に寄与せずに、収束までの試行サイクル数が膨大になる傾向がある。これらの欠点を軽減したのが石川、十時らによる次の並列反復改善法[1)]である。

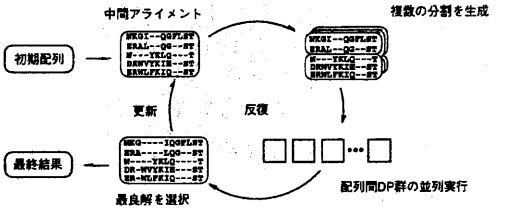

図3　最良優先探索の並列反復改善法[1)]

2.4　並列反復改善法

並列反復改善法[1)]では、最良優先探索を行なう。最良優先探索は、解空間において、注目する状態の近傍を全て探索し、そのうちから最も良い状態へ遷移する探索法である。最良優先探索に基づく並列反復改善法は、近傍探索の部分を並列化したもので、最もナイーブな方法では次に示す手順で改善を行う(図3)。

(1)　N本の配列の初期アライメントを与える。
(2)　初期アライメントに対し、可能なサブアライメントへの分割$2^{N-1}-1$通りを全て生成する。
(3)　分割されたグループ(サブアライメント)間でのペアワイズアライメントによりスコアを改善する。各分割法ごとに、これを並列に行う。
(4)　それらの結果を比較し、スコアの最も良い解を、次の改善サイクルの初期アライメントとする。

あるサイクルで、スコアに改善がみられなかったら、その時点のアライメントを最終解とする。

実用規模のアライメント問題では、(2)における分割法の数が膨大になるので、ヒューリスティクスにより探索空間を効率良く制限するために、次の限定分割[1)]を導入する。

2.4.1　1本抜き限定分割、1〜2本抜き限定分割

複数配列を2つのグループに分割する時に、片方のグループの配列数を、1本抜き限定分割[1)]では1本、1〜2本抜き限定分割[1)]では1本か2本とする。配列N本のアライメント問題の場合、分割法はそれぞれN通りと$N(N+1)/2$通りになる。ナイーブな反復改善法の分割法が、配列の本数に対して指数オーダーであったのに対して、それぞれ1乗オーダー、2乗オーダーとなるが、解のスコアはそれほど低下しない。1〜2本抜き限定分割は、1本抜き限定分割よりも、平均して若干スコアの良い解を生成する。こうした限定分割が有効なのは、アライメント途中の配列グループは、その配列数が多いとカラムの特徴が平均化されているのに対し、配列数が少ないとカラムの特徴が比較的よく表れるためと考えられる。カラムの特徴がはっきりしていると、配列グループ間のペアワイズアライメントの最適化が効率良くなされ、スコアの改善が大きい。

2.4.2　ツリー依存限定分割

ツリー依存限定分割[1)]は1乗オーダーの分割法で、1〜2本抜き限定分割をしのぐ効果をあげる。1〜2本抜き限定の、1本抜き限定に対する優位性は、アライ

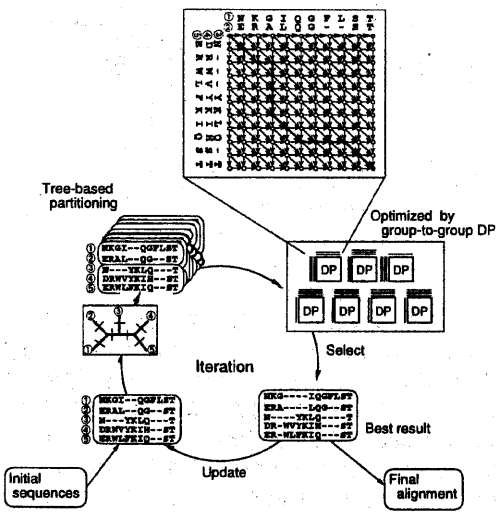

図4　ツリー依存限定分割 並列反復改善法[1)]

メントの過程で良く揃った2本が一緒に処理されることにある。そこで、ツリー依存限定分割では、反復改善の過程で良く揃った配列グループを、改善サイクルごとに抽出し、それらと残りの配列間でアライメントを行なう(図4)。ツリー依存限定分割では、各改善サイクルの始めにその時点でのアライメントのツリーを配列の類似性に基づいて作成し、ツリー上で近い配列は、なるべく同じ配列グループに含まれるように、分割を決める。具体的には、得られたツリーに対して、ツリーの枝の任意の1か所を切断して分けられる2つの配列グループを、改善の対象とする分割に選ぶ。この分割法はツリーの形状によらず、配列N本に対して$2N-3$通りある。末端の枝が切断される場合は、1本と残りの分割になるので、ツリー依存限定分割には、1本抜き限定分割が含まれている。

3. 並 列 化

3.1 実　　装

我々は、PVMライブラリを利用して、マスター・スレーブ方式でツリーベース並列反復改善法の近傍探索の部分を並列化し、これをを Hitachi SR2201(PU数: 256、主記憶:64GB 分散) 上に実装した。近傍探索におけるヒューリスティクスとしてツリー依存限定分割を用いた。アルゴリズムは文献1で既に提案済みであったが、A*アルゴリズムによる探索空間の効率的な刈り込みを実現し、汎用の通信ライブラリを用いて、unix 環境上で並列実装されたのは初めてである。

具体的には、まずマスタープロセスは、使用するPU台数分のスレーブプロセスを生成する。次にマスタープロセスは、初期アライメントからサブアライメントへの2分割をツリー依存限定分割で複数組生成し、それぞれの組を各スレーブプロセスに配る。ここでは、マスタープロセスは、各スレーブプロセスに対して1タスクずつを投げ、処理が終ったスレーブプロセスに、新しい処理を投げるようにする。各スレーブプロセスは、グループ間のペアワイズアライメントによる改善を行なう。計算結果はマスタープロセスに集められ、マスタープロセスは最良だった近傍を選択する。

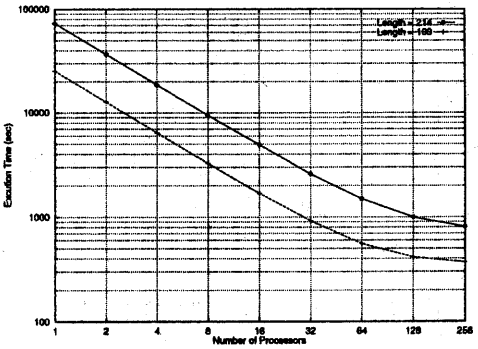

図5　SR2201 上での処理時間

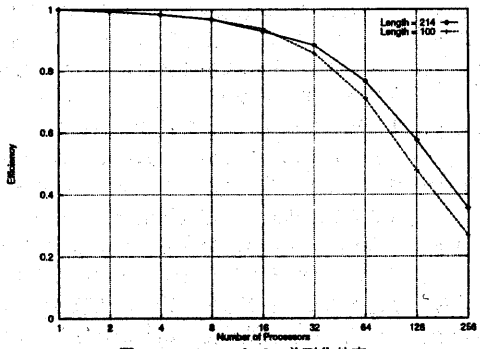

図6　SR2201 上での並列化効率

3.2 並列化の評価及び考察

129本のリボカリンファミリーのアミノ酸配列から、配列長の異なる2種類のデータを作成し、それらを用いて並列化効率の評価を行なった(図5, 図6)。ツリー依存限定分割では、サブアライメントへの分割法は配列N本に対して$2N-3$通りであるので、129本に対しては分割法(タスク数)は255通りである。

図5、図6に示すようにPU数が64以上になると、並列化効率が低下する。その原因としては、1つのマスタープロセスから逐次的に同期通信を行なっているためのボトルネックが考えられ、タスク分配機構の改善が課題である。また、2種類のデータを比較すると、タスクの粒度の大きいデータ(最大配列長214)の方が、粒度の小さいデータ(最大配列長100)よりも並列化効率が良い。実用的な問題では、今回の評価データに比べて配列長が長い場合が多く、タスクの粒度も大

きくなるので、並列化効率はやや改善されると考えられる。

4. A*アルゴリズムによる探索空間の効率的な刈り込み

ツリーベース並列反復改善法の中で行なっている、配列グループ間のペアワイズアライメントは、2次元DPの技術で厳密に解決できる[8)9)]。DPを用いると、最適解を求めるには、2次元のグラフ(図1)上を盲目的に全探索する必要がある。全探索DPでは最適解が保証されるが、無駄な計算が多くなる。ペアワイズアライメントでは、最適解のパスは2次元グラフの対角線近傍に存在する場合が多いため(配列間の類似性が高い場合)、対角線から遠い領域を強制的に削除して、計算量を削減する近似も提案されている。しかしこの方法では、最適解のパスが対角線から遠い領域に存在する場合(配列間の類似性が低い場合)は、最適解が求まらず、解が極端に悪くなる危険性がある。ペアワイズアライメントの解が悪いと、反復改善法の中で悪い局所解に陥る。最終的なマルチプルアライメントの精度を良くするには、ペアワイズアライメントで良い解を求めることが必要である。

ここでA*アルゴリズム[10)]を用いると、2次元グラフ上で必要な部分だけを探索して最適解を求めることができる。全探索DPと比較すると、不必要な部分を探索しないので、処理時間を節約できる。また、必ず最適解が求まるため、強制的に探索空間を削除するDPのように、解が極端に悪くなる危険性がない。

なおギャップコスト関数を単純な比例コスト(linear cost)ではなく、切片つきコスト(affine cost)にすると、ギャップコスト計算やA*アルゴリズムにおけるヒープ管理が複雑化するが、我々は後者を実現した。

4.1 A*アルゴリズムのペアワイズアライメントへの適用

DPでの探索では、2次元グラフのアークに正負のコストが混在して与えられ(図1)、コストの合計が最大になる経路を求めた。A*アルゴリズムでの探索では、コストは全て正として、コストの合計が最小になる経路を求めるように、探索空間を変換する必要がある。ここでは、コストの正負を逆転させて、コストが全て正になるようにコストの“かさ上げ”を行った。各アークに対して同量の“かさ上げ”では経路によって“かさ上げ”の量が異なるので、斜め方向のアークには縦・横方向のアークの2倍の“かさ上げ”を行い、経路による“かさ上げ”の量を同じにした。

また、グラフの任意のノードnから到着ノードまでの最短経路のコストとコストの推定値をそれぞれ$h(n)$と$\hat{h}(n)$とすると、A*アルゴリズムでは、最適解を保証するためには、$\hat{h}(n) \leq h(n)$が成り立つような$\hat{h}(n)$を設定する必要がある。このような$\hat{h}(n)$が与えられると、展開の候補ノードの中で、出発ノードからの最短経路のコストに$\hat{h}(n)$を足し合わせた値が最小のノードを展開していくと、少ないノード数展開で最短経路が求まる。もしも、$\hat{h}(n) = h(n)$ならば、不必要なノードを展開せずに最短経路を求めることができるので、$\hat{h}(n)$を$h(n)$に近づけることが、効率化の鍵である。ここでは、ノード$n(i,j)$における$\hat{h}$を次のように定義した。横、縦方向に配置させる配列の長さをそれぞれl_x、l_y、アークのコストを$C(i,j)$、ギャップペナルティをP'とすると、$l_x - i \geq l_y - j$の時、

$$\hat{h} = \sum_{m=j+1}^{l_y} \min_{i+1 \leq k \leq l_x} C(k,m) + ((l_x - i) - (l_y - j)) \times P'$$

$l_x - i < l_y - j$の時、

$$\hat{h} = \sum_{m=i+1}^{l_x} \min_{j+1 \leq k \leq l_y} C(m,k) + ((l_y - j) - (l_x - i)) \times P'$$

但し、アークのコストの最小値の候補には、ギャップコストが与えられている横(縦)方向のアークも含めて、そのコストには縦(横)方向のギャップコストの最小値を加算した。また、ギャップペナルティP'はギャップコストの最小値とした。

4.2 A*アルゴリズムの評価及び考察

2つのサブアライメント間のペアワイズアライメントをA*アルゴリズムと全探索DPを用いて行ない、処理時間を比較した(図7)。テスト用のデータは22本のキナーゼ配列(平均ホモロジー29%)のアライメント結果を、11本の2つのサブアライメントに分けて使用した。図7に示すように、全探索DPの処理時間に比べてA*アルゴリズムの処理時間は、配列長が長くなるほど短縮される。長さ392では0.44倍であるが、長さ2352では0.21倍まで低下する。また、A*アルゴリズムの効率は、配列の類似性が高いほど、向上することが容易に予想される。なぜなら、配列の類似性が高いと最適パスが2次元グラフの対角線周辺の領域に限局されるからである。このように、A*アルゴリズムの効率は、配列長と配列の類似性に依存する。本稿では詳細は省略するが、種々のデータを用いた実

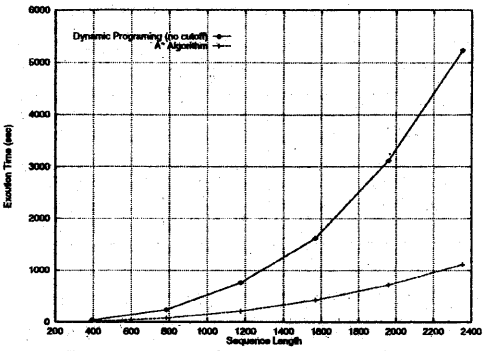

図7 A*アルゴリズムとDPの処理時間の比較

表1　各手法のアライメントスコアと処理時間の比較

手法	スコア	処理時間 (sec)	PU 数
ツリーベース法	-1052897	163	1
ツリーベース反復改善法 山登り探索 (逐次)	-682549	10251	1
本研究の手法 最良優先探索 (並列)	-674054	73224 1493 807	1 64 256

験を通じて、図7とほぼ同じ傾向を示し、全探索 DP に対して処理時間が 0.2 ～ 0.5 倍程度に短縮できることが計測された。

5. 従来法との比較及び考察

従来法の「ツリーベース法」[3)4)5)] 及び「ツリーベース反復改善法」[7)8)] と、本研究の「ツリーベース並列反復改善法」のアライメントの精度と処理時間の関係を比較した (表1)。従来法の「ツリーベース反復改善法」では山登り探索を1本抜き限定分割でラウンドロビンに逐次に行なった。本研究の「ツリーベース並列反復改善法」では最良優先探索をツリー依存限定分割で並列に行なった。データは、129本のリポカリンファミリーのアミノ酸配列 (最大配列長 214、平均ホモロジー 24%) を使用した。表1の例に示されるように、従来法の「ツリーベース法」は処理時間は速いが、精度が極端に悪い。本研究の手法が精度が一番よく、並列化により処理時間も抑えられており、最も実用的な手法といえる。本研究の手法は保存領域以外の領域もかなり正確にアライメントするので、見かけのスコア差以上に生物学的な価値があり、特に進化系統樹の研究に非常に有効であると生物学者から評価を得ている。

6. ま と め

精度の良いマルチプルアライメントを求めるため「ツリーベース並列反復改善法」を PVM ライブラリを用いて並列化した。また A*アルゴリズムを適用して探索空間の効率的な刈り込みを実現した。今回行なった並列化と探索空間の刈り込みにより処理時間が大きく抑えられた。並列化により約 90 倍、探索空間の刈り込みにより 5 倍の速度向上が計測された。併せて約 450 倍の速度向上となる。さらに配列長が長い問題ではより大きな速度向上が期待できる。高速化により、従来は計算時間が膨大なために困難とされていた本数の多いアライメントの精度の向上を、現実的な計算時間内で実現することを可能とした。特に配列の類似性が低い場合に、アライメントの精度の向上が大きく、進化距離が遠い配列の研究の有力な道具となる。

謝　辞

本研究の性能評価のために貴重なアミノ酸配列データをご提供頂いた、生物分子工学研究所 藤 博幸 博士に深謝します。

参 考 文 献

1) 石川幹人, 十時泰, 戸谷智之, 星田昌紀, 広沢誠, "並列反復改善法によるタンパク質の配列解析," 情報処理学会論文誌, Vol.35, No.12, pp.2816-2830 (1994).
2) 石川幹人, "並列計算機を用いたタンパク質の配列のアライメント解析," *Institute for New Generation Computer Technology, Technical Report,* TR-0902 (1994).
3) Feng, D.F. and Doolittle, R.F., "Progressive Sequence Alignment as a Prerequisite to Correct Phylogenetic Trees," *J. Mol. Evol.,* Vol.25, pp.351-360 (1987).
4) Higgins, D.G., Bleasby, A.J. and Fuchs, R., "CLUSTAL V: improved software for multiple sequence alignment," *Comput. Applic. Biosci.,* Vol.8, pp.189-191 (1992).
5) Barton, J. G., "Protein Multiple Sequence Alignment and Flexible Pattern Matching," *Methods in Enzymology,* Vol.183, Academic Press, pp.403-428 (1990).
6) Hirosawa, M., Totoki, Y., Hoshida, M. and Ishikawa, M., "Comprehensive Study on Iterative Algorithms of Multiple Sequence Alignment," *Comput. Applic. Biosci.,* Vol.11, No.1, pp.13-18 (1995).
7) Berger, M.P. and Munson, P.J., "A novel randomized iterative strategy for aligning multiple protein sequences," *Comput. Applic. Biosci.,* Vol.7, pp.479-484 (1991).
8) Gotoh, O., "Optimal Alignment between Groups of Sequences and its Application to Multiple Sequence Alignment," *Comput. Applic. Biosci.,* Vol.9, pp.361-370 (1993).
9) Gotoh, O., "Further improvement in methods of group-to-group sequence alignment with generalized profile operations," *Comput. Applic. Biosci.,* Vol.10, no.4, pp.379-387 (1994).
10) Araki, S., Goshima, M., Mori, S., Nakashima, H., Tomita, S., Akiyama, Y. and Kanehisa, M., "Application of Parallelized DP and A* Algorithm to Multiple Sequence Alignment," *Proc. Genome Informatics Workshop IV,* pp.94-102 (1993).
11) 木村資生, "分子進化の中立説," 紀伊国屋書店 (1986).
12) Henikoff, S. and Henikoff, J.G., "Amino acid substitution matrices from protein blocks," *Proc. Natl. Acad. Sci. USA,* Vol.89, pp.10915-10919 (1992)
13) Dayhoff, M.O., Schwartz, R.M. and Orcutt, B.C., "A Model of Evolutionary Change in Proteins," *Atlas of Protein Sequence and Structure,* Vol.5, No.3, Nat. Biomed. Res. Found., Washington DC, pp.345-352 (1978).
14) Needleman, S.B. and Wunsch, C.D., "A general method applicable to the search for similarities in the amino acid sequences of two proteins," *J. Mol. Biol.,* Vol.48, pp.443-453 (1970).