並列 DP マッチングによる大域アラインメントの高速化

山森　一人[1)] ・ 東　博志[2)] ・ 吉原　郁夫[3)]

High-speed Global Alignment by Parallel DP Matching

Kunihito YAMAMORI[1)], Hiroshi HIGASHI[2)], Ikuo YOSHIHARA[3)]

Abstract

Recent advance of genomic analysis technologies gives us a lot of DNA sequence data. These are used for further genome analysis. To find the optimum global alignment is investigated as one of the genomic analysis. For this purpose, Dynamic Programming (DP) matching is usually used. However, DP matching requires storage space in the square of the length of the sequences. For example, to find the optimum global alignment for one million bases, 1TB memory is needed. So, divide and conquer method is proposed to reduce required memory space. However, it leads increasing of the amount of calculation of DP matching. So we propose a method to achieve fast computation time with small required memory by combination of branch-cut and parallel processing. Our method succeeded to reduce computation time to 8% of conventional method.

Key Words:
Dynamic Programming, DP matching, global alignment, DNA sequence, parallel, branch-cut

1. はじめに

近年，ヒトをはじめとする様々な生物種のゲノム解析プロジェクトにより，大量のゲノム情報を得ることができるようになった [1]。得られたゲノム情報を利用し，遺伝子の機能や役割を解明する研究（ポストゲノム）が盛んに行われている。

ゲノム解析の分野では，計算機による解析が不可欠になっている。計算機で処理を行うことで，ヒトの手では不可能であった大量のデータを処理することが可能になった。さらに，計算機自身の処理性能向上も研究の発展の一翼を担ってきた。

[1)]情報システム工学科准教授
[2)]情報システム工学科（現在，テラインターナショナル株式会社）
[3)]情報システム工学科教授

ポストゲノムの一つとして，2 本の塩基配列の類似性に着目し，グローバル（大域）アラインメントを求め，解析に利用する研究がある。配列全体のアラインメントを得ることにより，異なる生物種間で塩基配列の類似部や非類似部を見つけることができる。つまり，アラインメントが塩基配列の意味を知る手掛かりになる可能性がある。

アラインメントを求める方法として，動的計画法 (Dynamic Programming：DP)を利用した DP マッチング [2]が使われる。しかし，塩基配列が長くなるにつれて計算時間が膨大になるという問題がある。また，アルゴリズムの性質上，塩基配列長の二乗のメモリ領域が必要となり，塩基配列が長くなればメモリ領域を確保することが困難になる。例えば，長さ約 100 万の DNA 塩基配列 2 本のアラインメントを行うとすると，計算回数は 1 兆回，使用するメモリ領域は 1T バイトに相当

する。ヒトの染色体塩基配列で塩基数が一番少ない21番染色体で約4500万，塩基数が一番多い1番染色体では2億8000万近くになる。ヒトの染色体塩基配列のような，長い塩基配列のアラインメントは計算に時間がかかり，メモリ確保に至っては通常の計算機では不可能であると言える。

Kleinberg ら [3]は分割統治法の考えを適用して計算に使うメモリ領域を削減する手法を提案した。しかし，Kleinberg らが提案した方法ではメモリ使用量が削減できる代わりに，アラインメントを行うための計算量が多くなる。

本研究ではメモリ使用量を削減しつつ，アラインメントを求めるうえで不要と考えられる部分の計算を削減すると共に，DP マッチング処理を並列化することで高速化する手法を提案する。

2. 塩基配列のアラインメント

2.1. DNA の構造と塩基配列

DNA(Deoxyribonucleic Acid：デオキシリボ核酸)は，遺伝子情報をコーディングする生体物質のことで，塩基とデオキシリボース（五炭糖），リン酸から構成される。DNA の塩基にはアデニン(adenine; A)，シトシン(cytosine; C)，グアニン(guanine; G)，チミン(thymine; T)の4つが存在する [2]。

DNA の塩基配列を決定することをシークエンシングという。シークエンシングにより得られた塩基配列を利用して，DNA の機能特定や生物種ごとの DNA の類似部，非類似部の特定など，さらなるゲノム解析が可能になる。

2.2. アラインメント

2 本の塩基配列に対して塩基の間にギャップと呼ばれる隙間を入れ，それぞれの塩基同士を対応付けることをアラインメントという。アラインメントを行うことにより，塩基配列の類似部や非類似部を発見できる。図 1 (a)は 2 本の塩基配列を並べたものである。また，図 1 (b)は(a)の 2 本の塩基配列にアラインメントを適用した例であり，塩基中に挿入されている‘－’はギャップを表す。網掛けされている部分は，上下の塩基が一致していることを表す。このようにアラインメントすることで塩基配列同士の類似性を正しく比較できる。

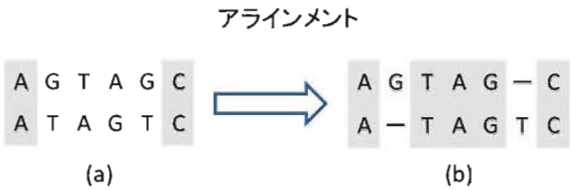

図 1： 塩基配列のアラインメント

表 1: ペナルティ計算テーブル

状態	ペナルティ
Match	+0
Mismatch	+2
Gap	+3

アラインメントを行い塩基配列の類似部や非類似部を見つけるには，適切なギャップの挿入パターンを見つける必要がある。そこで，塩基の一致，不一致，ギャップの挿入にペナルティを設け，アラインメントの類似性を判定する。本研究ではペナルティの和を計算するため，表 1 のようにペナルティを定義した。ペナルティの和のことをペナルティスコアと呼び，ペナルティスコアが最も少なくなるようなアラインメントを最適アラインメントと呼ぶ。

2.3. DP マッチング

図 2 は 2 本の塩基配列 GTAGA と GTGA の最適アラインメントを，DP マッチングを利用して求めた例である。左上の G と G から，右下の A と A までのペナルティスコアが最小になるような経路を求めれば，それが最適アラインメントとなる。

2 本の塩基配列の長さをそれぞれmとn，各配列上の塩基の位置をそれぞれ$i(1 \le i \le m)$，$j(1 \le j \le n)$とし，2 つの 2 次元配列$P[m][n]$，$B[m][n]$を用意する。$P[\ ][\ ]$はペナルティスコアを保存する領域，$B[\ ][\ ]$はバックトレース情報を保存する領域である。式(1)，式(2)，式(3)により，(m,n)までのペナルティスコアとバックトレース情報を求めることができる。ここで(1)式のgはギャップのペナルティを表す。その後，$B[\ ][\ ]$に保存されている情報を(m,n)からバックトレースすることでアラインメントされた塩基配列を求めることができる。

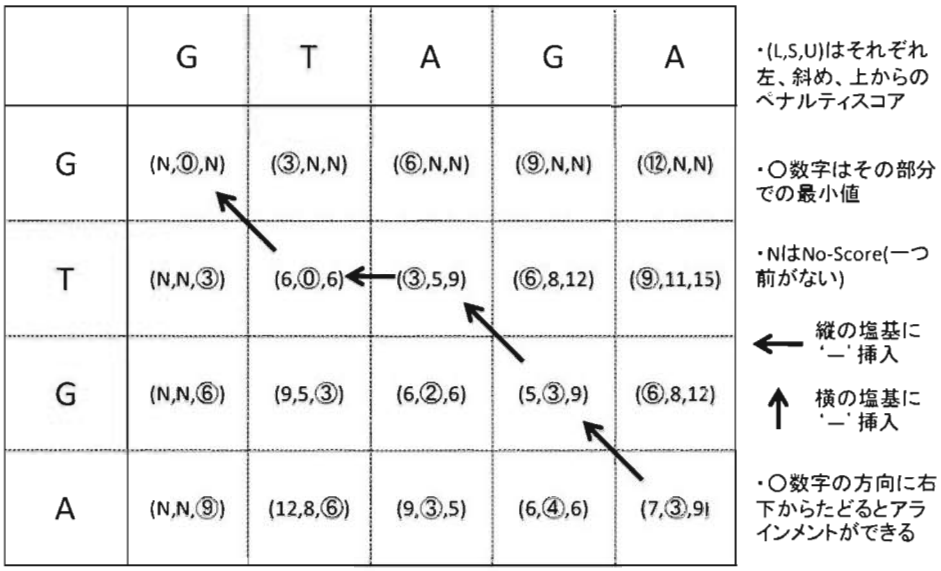

図 2: DP マッチング

$$P[i][j] = \begin{cases} P[1][1] = d(1,1), \\ \quad min \begin{cases} P[i-1][j-1] + d(i,j), -① \\ P[i-1][j] + g, \quad -② \\ P[i][j-1] + g, \quad -③ \end{cases} \end{cases} \tag{1}$$

$$d(i,j) = \begin{cases} 0, & if\ Match, \\ 2, & if\ Mismatch, \end{cases} \tag{2}$$

$$B[i][j] = \begin{cases} "\nwarrow", & if\ ①, \\ "\leftarrow", & if\ ②, \\ "\uparrow", & if\ ③, \end{cases} \tag{3}$$

$$1 \le i \le m, 1 \le j \le n$$

3. メモリ使用量削減とアラインメント高速化

3.1. 分割統治法による領域分割

アラインメントする 2 本の塩基配列の長さをそれぞれm, nとする。従来の DP マッチングは経路の保存のために$O(mn)$の領域が必要になる。塩基配列の長さが長くなればメモリ上に領域を確保することが難しくなる。そこで DP マッチングに分割統治法 [3]を適用して計算に利用するメモリ使用量の削減を行う。

図 3 は DP マッチングに分割統治法を適用した例である。まず図3 (a)のように計算領域を上下に2分割し，左上と右下からペナルティスコアの計算を行う。ペナルティスコアは直前の 1 行分だけを保存して次の行の計算を行うことにより，$O(mn)$の使用メモリ領域を$O(m)$にすることができる。分割線までのペナルティスコアまで求めたら，上下のペナルティスコアを足して最小値となった位置を分割点とし，図 3 (b)のように計算領域を左右に分割して白の領域で再度分割統治法を行う。この分割点は最適アラインメントが通る点であり，2 つの分割線の右上，左下の領域では DP マッチング処理を省略できる。一方，メモリ使用量削減のため 1 行分のペナルティスコアしか保存しないため，再度分割された領域内のペナルティスコアを計算しなおす必要がある。その後，図 3 (c)のようにペナルティスコアを計算する領域を小さくしていき，オンメモリで DP マッチングが可能になるまで分割統治法による領域分割を行う。適切な分割点をあらかじめ求めることによって，経路保存に使うメモリ使用量を削減する。しかし，同じ領域のペナルティスコアを何度も計算するので計算量は増大するという問題がある。これは分割後にペナルティスコアの計算を行うときに，ペナルティスコア計算のスタート位置が変わるため，先に求めたペナルティスコアを再利用できないからである。また，使用するメモリ削減のために，直前に計算した 1 行分のペナルティスコアだけしか確保しないため，領域分割の度にペナルティスコアを再度計算し直さざるを得ず，計算量が限りなく元の 2 倍に近くになるという問題がある。

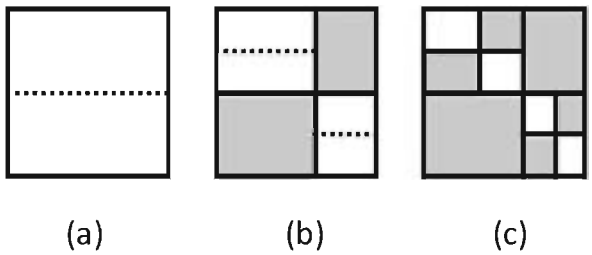

図 3: 分割統治法の例

3.2. 枝切り処理による高速化

3.1 節で示した問題点を解決するため，枝切り処理を行って計算量の削減を図る。

図 4 (a)はアラインメント対象となる塩基配列を，DP マッチングを行うために並べたものである。このとき，2 次元領域の左上から右下に伸びている対角線は，塩基配列が完全一致する場合のアラインメントの経路である。比較する塩基配列が類似していれば，アラインメントの経路は対角線に近づく。よって，領域の右上や左下は計算が不要な領域である可能性が高い。図 4 (b)は対角線から左右に塩基配列長の 50%を計算領域としている。この状態を探索領域 50%と定義する。探索範囲 50%では，領域全体の 4 分の 1 の計算領域を削減したことになる。さらに探索範囲を狭めて 10%とすると図 4 (c)のようになる。全く異なる塩基配列間でアラインメントを行うことは考えづらいため，探索範囲のみを対象にペナルティスコアを計算することで計算量を削減する。

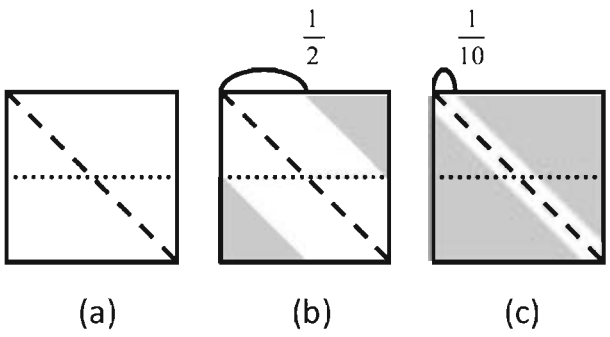

図 4: 枝切り処理による計算領域削減

3.3. 並列処理による高速化

枝切り処理に加えて並列処理を併用してさらなる高速化を図る。図 5 は CPU4 つによる並列化の様子を表している。まず，図 5(a)のように CPU1 と CPU3 を利用して分割点を求める。図 5(b)のように分割点で領域を分割し，それぞれの領域で CPU1 と CPU2，CPU3 と CPU4 で分割点をそれぞれ求める。図 5(c)のようになったらそれぞれの領域ごとに 1 個の CPU を割り当て，各領域ごとにオンメモリで DP マッチングが可能になるまで分割統治法による領域削減を行う。最後に分割した領域ごとにアラインメントを行い，求めた塩基配列を一つにつなげて大域アラインメントを完成させる。

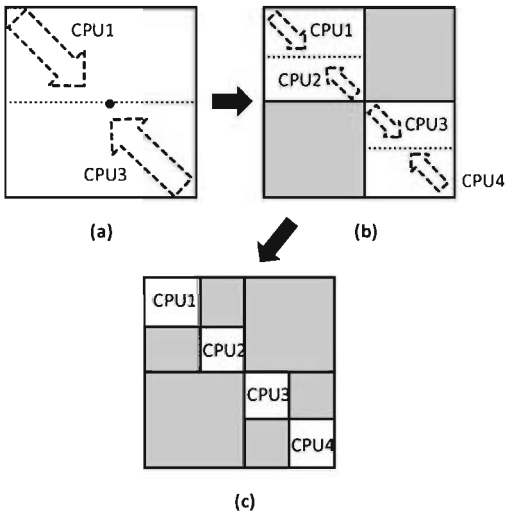

図 5: CPU4 つによる並列化

4. 提案手法による実験と結果

4.1. 実験データと実験・開発環境

本研究では，DDBJ(DNA Data Bank of Japan)で公開されているヒト 21 番染色体 GL000151 の塩基配列を利用する。塩基配列の 200 万塩基目から，長さ 200 万塩基を切り出す。切り出した塩基配列から 3%の確率で変化（置換，交換，削除）した擬似塩基配列を作成し，元の切り出した塩基配列とアラインメントを行う。

今回の実験使用した計算機は表 2 のとおりである。表 2 のように CPU2 個を搭載した計算機 16 台で構成されるクラスタ計算機を使用した。プロセッサ間通信には MPI(Message Passing Interface) [4]を使用した。MPI とは，並列プログラミングを行うための規格であり，

表 2： 本実験に使用した計算機の仕様

CPU	Intel® Xeon® 2.80GHz CPU
CPU の数	2 個/計算機
キャッシュ	2048 KB/CPU
メインメモリ	2 GB/計算機
計算機台数	16 台
計算機間通信	MPICH2 Ver1.0.7

CPU 間でメッセージの通信を行うことにより並列処理を可能とするライブラリとして実装されている。今回のアラインメントプログラムでは，分割点を求める場所，大域アラインメントを完成させる場所の 2 か所でプロセッサ間通信を行う。

4.2. 実験結果

図 6 は 1CPU で枝切り処理のみを行った時の実行時間を示したものである。縦軸は実行時間（千秒），横軸は探索範囲である。枝切りなしの状態から探索範囲を 50%，40%，…，10%と変化させてアラインメントを実行した場合，枝切りなしに比べ探索範囲 50%では理論値通りに実行時間を約 4 分の 3 に短縮できた。また，探索範囲 10%では枝切りなしに比べ実行時間が約 4 分の 1 に短縮できた。また，全ての探索範囲で求めたアラインメントの塩基配列は，枝切りなしのアラインメントの塩基配列と一致した。

図 7 は枝切り処理に加え並列化を行ったときの実行時間を示したものである。縦軸は実行時間（千秒），横軸は使用した CPU 数である。また，探索範囲は 10%で実験した。逐次処理時と比較して 2 個の CPU による並列化では実行時間が約 2 分の 1 に短縮できた。4 個の CPU による並列化時も 2 個の並列化と比べて 4 分の 3 程度の処理時間になった。しかし，6 個以上の CPU を使用してもあまり実行時間の短縮は見られなかった。この原因として，一番時間がかかる最初の分割点を求める計算を 2CPU のみで行っており，他の CPU に割り当てられる計算が少なくなり，並列化の効果が小さくなったと考えられる。

全体としては，枝切り処理と並列化の併用により，逐次処理に比べ 8CPU 利用時で実行時間は約 12 分の 1 に短縮できた。

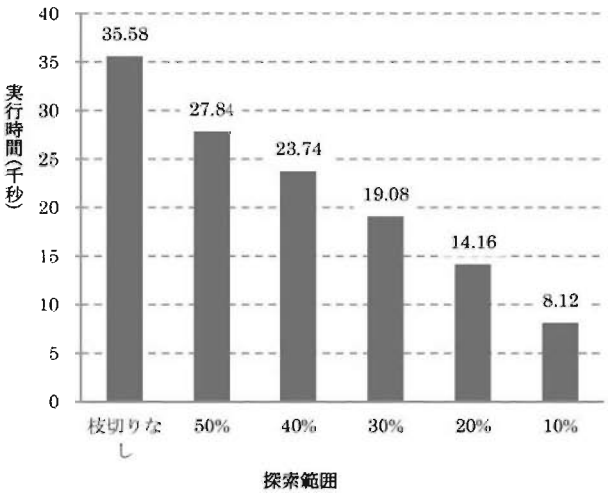

図 6: 枝切り処理による実行時間の変化

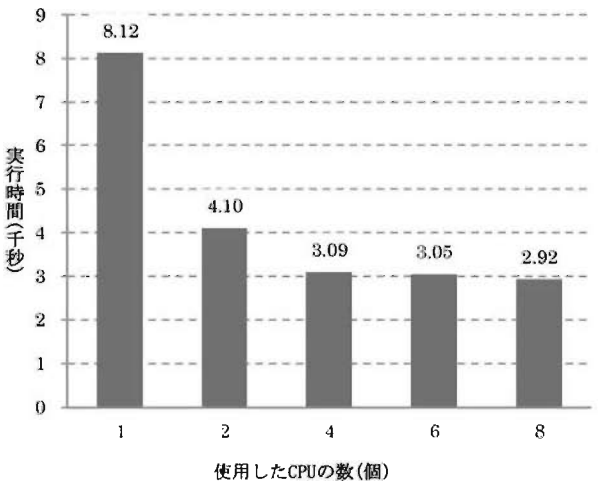

図 7: 並列処理による実行時間の変化

5. おわりに

塩基配列のグローバル（大域）アラインメントを，動的計画法を利用した DP マッチングを使用して行った。しかし，DP マッチングの計算量とメモリの使用領域は$O(mn)$であり，長い塩基配列になると処理時間とメモリ領域が必要になる。そこで，分割統治法を利用して使用するメモリを削減することにより，長い塩基配列のアラインメントを可能にする手法が提案されていた。しかし，分割統治法を利用することで計算量が限りなく 2 倍近くになってしまう課題があった。

本研究では，塩基配列のアラインメントを行うときに用いられる DP マッチングに，枝切り処理と並列化

を適用する高速化手法を提案した。アラインメントに不要と考えられる部分を枝切りし，さらにペナルティスコア計算部分と分割統治法で二分割した後の処理を並列化することにより，アラインメントの高速化を行う手法である。

枝切り処理の実験により，削減した領域の量に比例して実行時間も短縮されることが確認できた。また，今回実験した全ての探索範囲でアラインメントの精度を落とさずに高速化できることが分かった。並列化の実験では4CPUまでは実行時間の短縮が確認できたが，6CPU以上では，あまり効果が見られないことが分かった。この原因として，一番時間がかかる最初の分割点を求める計算を2CPUで行っており，この時間が支配的であったためであると考えられる。全体としては，枝切り処理と並列化の併用により，通常の逐次処理時に比べ8CPUの使用で実行時間は約12分の1に短縮できた。

今後の課題として，最初の分割点導出を全CPUを用いて高速に行う手法の開発が挙げられる。

参考文献

[1] ナショナルバイオリソースプロジェクト情報運営委員会，バイオリソース&データベース活用術，1st ed.: 秀潤社，2009.

[2] 榊原 康文，"バイオインフォマティクス概説 -比べることで生命は解明できるか?-," 情報処理，vol. 46, no. 3, pp. 230-238, 2005.

[3] J.Kleinberg, E.Tardos, "分割統治法による線形の領域の系列アライメント," in アルゴリズムデザイン.: 共立出版，2008, ch. 6.7, pp. 253-258.

[4] P.パチェコ，MPI並列プログラミング，1st ed.: 培風館，2001.