ICOT Technical Report: TR-0902

TR-0902

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

January, 1995

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191～5

Institute for New Generation Computer Technology

並列計算機を用いた
タンパク質配列のアライメント解析

石川 幹人

1994 年 12 月

もくじ

図一覧

1 緒論

本論文は、タンパク質配列の典型的な解析課題である、アライメント (alignment) の問題に対し、並列計算機を用いて、実用規模の高品質な処理結果を、実用的な時間内に与える技術についての研究報告である。まず、本章では、本研究の位置付けを明確化する。最初に、アライメントとは何か、それがどのような問題で、何に利用されるかを概説する。次に、その問題を解く従来の計算機技術を説明し、それがどのような点で不十分であったかを指摘する。そして、最後に、本研究の目的と意義を述べ、第2章以降で述べる研究内容への導入とする。

1.1 タンパク質配列のアライメント解析

本節では、タンパク質とその配列について述べた後、アライメントの具体例を挙げ、それがどんな生物学的な解析に利用されるかを概説する。

1.1.1 タンパク質とアミノ酸配列

よく知られているように、生物の細胞の内部にある DNA には、遺伝情報が格納されている。遺伝情報は、DNA 上の4種類の塩基の配列に符号化されており、ヒトの場合、全長は30億塩基にのぼる。遺伝情報の単位を遺伝子と呼び、それが DNA 上に点在している。ヒトの DNA には、およそ 10 万の遺伝子があるといわれる。ひとつの遺伝子は、RNA の断片に転写された後に、スプライシングなどの過程を経て、その塩基配列は3塩基ごとにアミノ酸に翻訳され、アミノ酸配列が生成される（図1）。このアミノ酸の鎖が、空間的に折れ畳まって特異的な形状になったものが、タンパク質である。いわば、タンパク質は、遺伝情報の内容が具体的に表現されたものであり、生物の体の形成、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、２０種類があり、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、どんな性質のアミノ酸がどんな順番に連なっているかで、タンパク質の構造や機能が決まってくる。すなわち、タンパク質は、アミノ酸を表すアルファベットの配列で特徴づけられる。ひとつのタンパク質を表現した配列は、短いもので数十文字、長いものでは千文字にものぼる。

タンパク質の研究で重要な事柄のひとつは、タンパク質の構造や機能をつきとめることである。しかし、それは生化学的な実験を積み重ねて初めてわかるものとされている。事

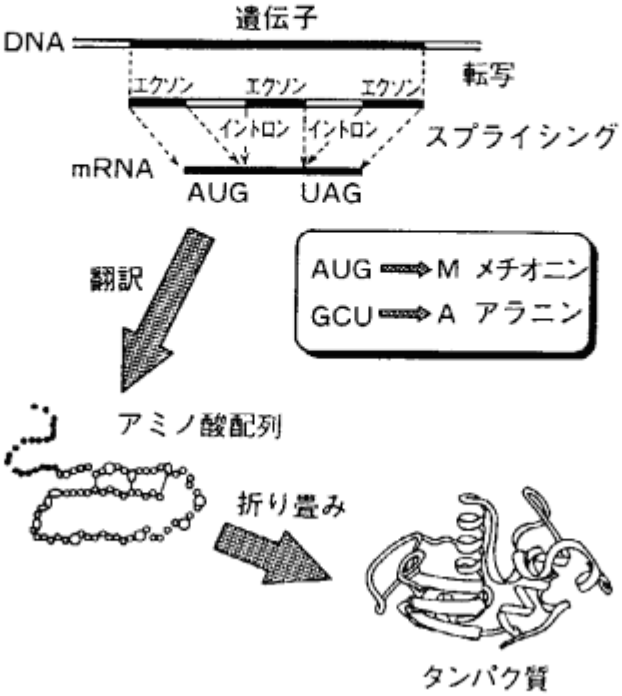


図 1: 遺伝子からのタンパク質の生成

実、タンパク質の正確な構造は、まだ4百種類程度しか知られていない（PDB データベース [1]）。一方、タンパク質のアミノ酸配列を調べる技術は、すでに確立されており、それを自動分析する機械も販売されている。現在では約3万種類のタンパク質について、その配列が決定されている（PIR データベース [2]）。タンパク質の構造や機能の解明は、その配列に比べ、十分に進んでいない。しかし、類似したアミノ酸配列をもつタンパク質は、類似した構造や機能をもつ傾向があるので、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である。このように、配列間の類似性を解析する情報処理技術の迅速な確立が望まれている [3]。

1.1.2 アライメントの例

もっとも基本的な配列の類似性解析のひとつは、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) と呼ばれる。まず、具体的な例を示そう。図 2(a) にはアミノ酸配列が6本示されている。これらはレトロウイルス (retro-virus) がもつエンドヌクレアーゼ (endonuclease) という酵素の一部分である。レトロウイルスは自分の遺伝情報を宿主細胞のＤＮＡに刷り込んで増殖する。このエンドヌ

クレアーゼは、そうしたレトロウイルスの増殖過程において、ＤＮＡを切る働きを担う酵素である。各配列の左側にある見出しは、それぞれの酵素断片が抽出されたレトロウイルスの名前を示している。たとえば、HTLV はヒトＴ細胞白血病ウイルスのことで、AIDS ウイルスに非常に近い仲間である。

```
(a)
copia  : ILDFHEKLLHPGIQKTTKLFGETYYFPNSQLLIQNIINECSICNLAKTEHRNTDMPTKTT
M-MuLV : LLDFLHQLTHLSFSKMKALLERSHSPYYMLNRDRTLKNITETCKACAQVNASKSAVKQGTR
HTLV   : LTDALLITPVLQLSPAELHSFTHCGQTALTLQGATTTEASNILRSCHACRGGNPQHQMPRGHI
RSV    : VADSQATFQAYPLREAKDLHTALHIGPRALSKACNISMQQAREVVQTCPHCNSAPALEAGVN
MMTV   : ISDPIHEATQAHTLHHLNAHTLRLLYKITREQARDIVKACKQCVVATPVPHLGVN
SMRV   : ILTALESAQESHALHHQNAAALRFQFHITREQAREIVKLCPNCPDWGSAPQLGVN

(b)
copia  : -------ILD--F-------HEKLLHPGIQKTTK-LF--GET-YY-FPNSQLLIQNIINECSICNL-AKT-EHR--N-TDMPTKTT
M-MuLV : -------LLD-FL-------HQ-LTHLSFSKM-KALLERSHSPYYMLNRDRTL-KNITETCKACAQ-VNA-SKS--A-VKQGTR--
HTLV   : LTDALL-ITP-VLQLSPAELHS-FTHCGQTAL-T-LQ---------GATTTEA--SNILRSCHACRG-GNPQHQMPRGHI-------
RSV    : VADSQATFQAYPLR-EAKDLHT-ALHIGPRAL-S-KA--------CNISMQQA--REVVQTCPHC---NSA-PALEAG-VN-------
MMTV   : --------ISD-PIH-EATQAHT-LHHLNAHTL-R-LL--------YKITREQA--RDIVKACKQCVV-ATPVPHL--G-VN------
SMRV   : --------ILT-ALE-SAQESHA-LHHQNAAAL-R-FQ--------FHITREQA--REIVKLCPNCPDWGSA-PQL--G-VN------
         ....................H....H.................................C..C.....................
```

図 2: アライメントの例：(a) アライメント前の配列群，(b) アライメント結果

図 2(b) には (a) をアライメントした結果が示してある。同じアミノ酸や性質の似たアミノ酸が縦に同じカラム位置になるように、ところどころハイフンが挿入されている。このハイフンの一連の横方向の連なりをギャップ (gap) という。各文字配列にギャップを入れた結果、図 2(b) では、H で示されるヒスチジン (histidine) が２個と、C で示されるシステイン (cysteine) が２個とが縦に揃っている。縦方向に揃った代表的文字で構成される配列を、そのアライメントのコンセンサス配列 (consensus sequence) という。図 2(b) の最下行は、コンセンサス配列である。

また、コンセンサス配列のなかに見られるパターンが、アライメントされた配列群を特徴づけるものと判断できるとき、そのパターンを、配列モチーフ (sequence motif) とか、単にモチーフ [4] と呼ぶ。C が２個と H が２個で特徴づけられるパターンは、「亜鉛の指」、ジンクフィンガー (zinc finger) と呼ばれる有名なモチーフである [5]。図 3は蛋白質のうちジンクフィンガー部分に相当する立体構造を示している。C が２個と H が２個で亜鉛イオンに結合することで、「指」に相当する構造が形成される。ジンクフィンガーをもつ蛋白質は一般に、何本ものそうした「指」をもち、ＤＮＡの２重らせんの溝に、それらの「指」が入り込むことで、特定のＤＮＡ配列を認識し、結合する機能をもつ。エンドヌクレアーゼは

ＤＮＡを切る働きをもつのであるから、その酵素に、ＤＮＡ結合機能をもつ部位が存在するのは当然予想されることである。すなわち、図 2(b) でアライメントにより抽出されたパターンは、ジンクフィンガーのモチーフである可能性が高い。

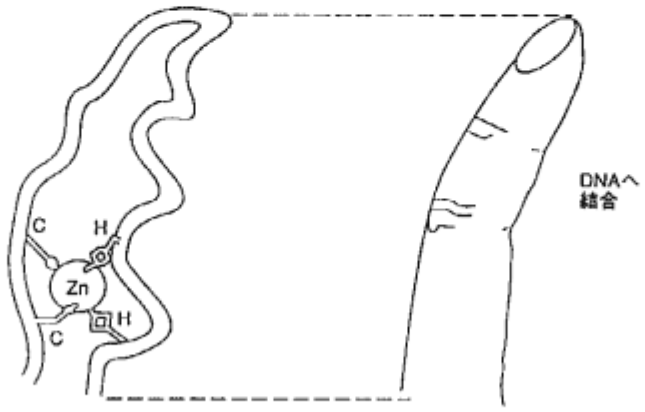


図 3: ジンクフィンガー（亜鉛の指）の構造：Zn は亜鉛イオン

アライメントのうち比較的類似したアミノ酸が縦に並んでおり、ギャップの入り方も少ない領域を、保存部位 (conservative site) という。図 2(b) では、２個の H の周辺、２個の C の周辺は保存性が高く、それに対し、それらの間の配列中央部は大きなギャップが入っており保存性が低い。保存性の高い部位は、配列のなかでも蛋白質の構造や機能の実現のうえで重要な部分であると推測できる。なぜなら、配列の多様性は、突然変異と自然淘汰からなる分子進化の結果で生まれたものだからである [6, 7]。蛋白質の構造や機能の実現のうえで重要な部分に突然変異が起きると、多くは蛋白質の異常を起こし、その生物は淘汰されてしまう（図 4）。すると、結果として、重要な配列部分は進化の過程のなかで保存されるので、生き残っている生物の同種の蛋白質をアライメントで調べると、自ずと保存部位が発見できるのである。

1.1.3　アライメントの利用目的

アライメントのうち、とくに配列の数が２本のものをペアワイズアライメント (pairwise alignment) と呼び、３本以上のものをマルチプルアライメント (multiple alignment) と呼んで区別する。本項では、これらのアライメントが主にどんなことに使用されているかを述べる。

ペアワイズアライメントは、基本的に配列と配列との類似性の度合を求めることに使われる。そこで、ペアワイズアライメントは第１に、マルチプルアライメントの始めの段階で、

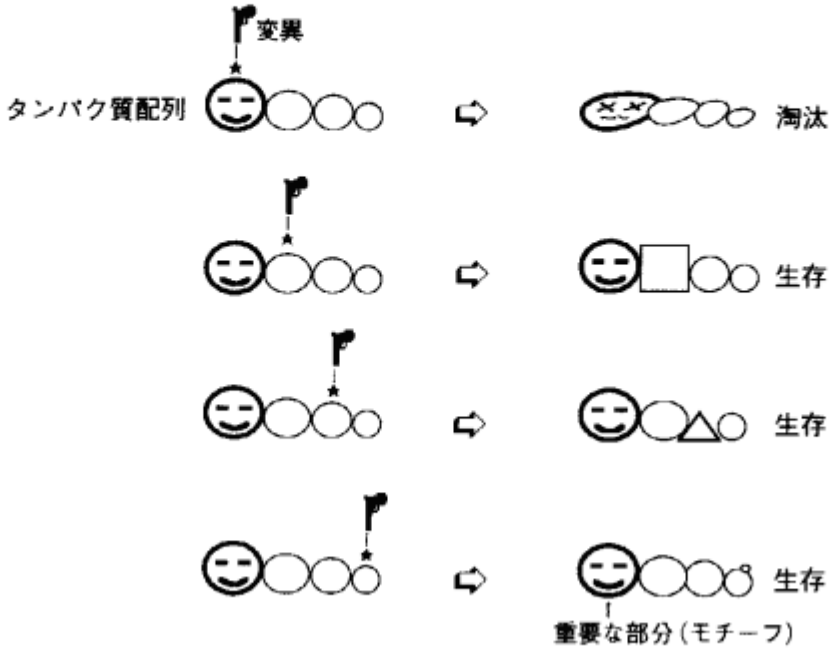


図 4: 重要な配列部分が分子進化で保存される仕組み

どの配列とどの配列が似かよっているかの判断に使用される。第2に、配列データベースから類似の配列を検索する手段に利用されている。たとえば、ペアワイズアライメントの手法を利用した高速配列検索ツールに FASTA[8] がある。

マルチプルアライメントは一度に複数本を比較するので、配列に存在するノイズの影響をあまり受けることなく、ペアワイズアライメントに比べて、より効果的に配列の共通性を見い出すことができる。

マルチプルアライメントの第1の利用目的は、共通配列の抽出である。すでに前項でアライメントからジンクフィンガーモチーフを共通配列として抽出する例を紹介した。モチーフ抽出には、すでに判明しているモチーフを同定する場合と、新たなモチーフを発見する場合とがある。新たなモチーフの発見には、単にアライメントのコンセンサスに見られたパターンというだけでは不十分で、生物学の知見から生物学的な意義づけがされることが必要とする考え方が主流である。また、PCR(Polymerase Chain Reaction) 法 [9] に使用するプライマー (Primer) の設計のために、共通配列を抽出することがある。PCR 法では、希望の DNA 断片を大量に増幅できるが、その際に希望する DNA 断片の両側の６～１０塩基がプライマーとして必要となる。あるタンパク質のグループに、一対の固有の共通配列が見つかれば、それの相当する DNA 配列をプライマーにして、そのグループに属する未知のタンパ

ク質のDNA断片を発見することができる。

第2の利用目的は、構造／機能の予測である。アライメントされた配列群のなかに配列の構造／機能がわかっているものがあれば、それとよくアライメントされる他の配列にも、同様な構造／機能があるに違いないと推測できる。図2(b)の最初の配列 copia は、実はエンドヌレアーゼではなく、ショウジョウバエのＤＮＡから見つかった配列をアミノ配列に翻訳したものである。copia は、他のレトロウイルスのエンドヌレアーゼとよくアライメントできるので、エンドヌレアーゼの活性をもつのではないかと推測できる。事実、この例では、生化学的に活性が確認されている[10]。

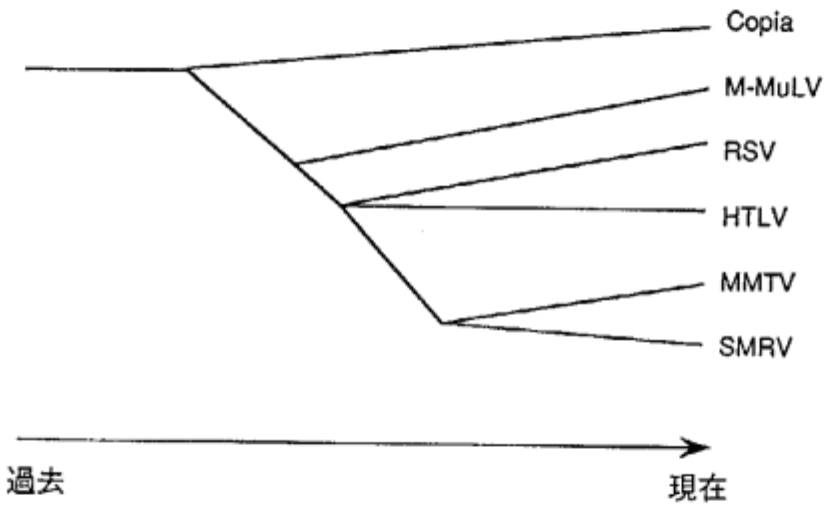


図 5: 進化系統樹の例

第3の利用目的は、進化系統樹の解析である。マルチプルアライメントの結果から、分子進化の過程でどのような突然変異（アミノ酸の置換、挿入、欠失）が起きたかを推定でき、たとえば図5のような系統樹が描ける。系統樹は過去から現在へ配列がどのような順で変化してきたか、生物種がどのように分化してきたかの歴史を示すものである。ただし、祖先の配列は現在残されていないので、いくつかの仮定から祖先の配列を推定しながら、系統関係を決めていく。図5のようなウィルスの系統樹からは、類縁のウィルスには同様な薬剤が効くなどの、医学的な検討にも役立つ。マルチプルアライメントの結果から系統樹解析をする手法には、距離行列法、節約法、最尤法などがある。手法によって仮定とするところが少し異なるので、得られる系統樹の形も少しずつ異なってくる[11]。

1.2 従来のアライメント解析技術

本節は、本論文の研究の背景となる、従来の解析技術についての総説[12]である。ペアワイズアライメント法、マルチプルアライメント法、そして、並列技術の応用とに分けて順に述べる。

1.2.1 ペアワイズのアライメント法

配列2本（ペアワイズ）のアライメントについて、アライメント全体の評価値を最適化するようなマッチング(global matching)問題は、ダイナミックプログラミング（以下DPと略す）の技術で厳密に解決できる。それに対して、配列の局所的な類似性に注目してアライメントする、ローカルマッチング(local matching)法も知られている。

グローバルマッチング グローバルマッチングの問題は、与えられた評価基準に沿った最適解が、DPにより高速に求められる。DPは、最適化問題の解法として古くから知られていた技術であるが、生物の配列マッチングに応用されたのも古く、1970年のことである[13]。

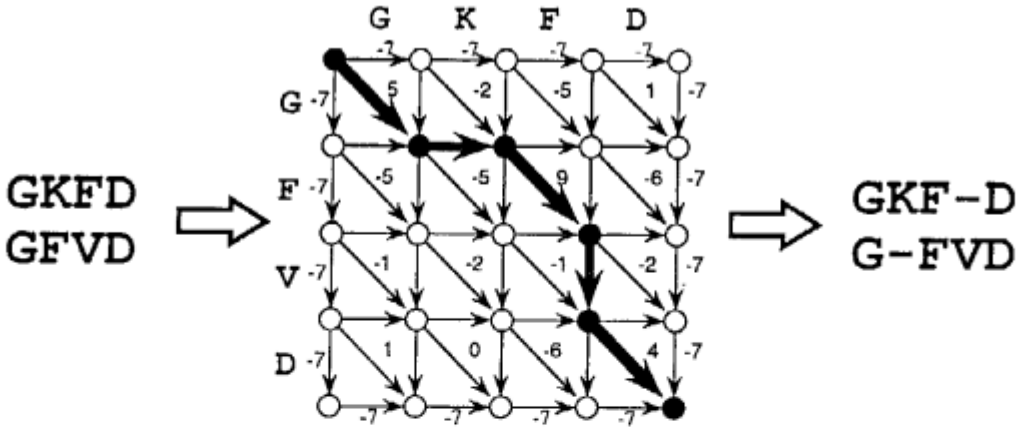
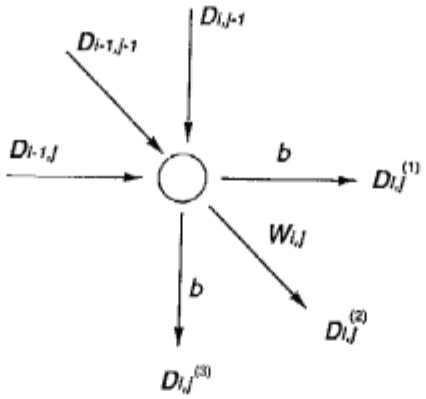


図6: ダイナミックプログラミングによるペアワイズアライメントの解法

DPによるペアワイズアライメントの解法について概念的説明を、図6を用いて行う。たとえば、GKFD, GFVDという2つの短い配列をアライメントする場合、この2つの配列を図のような2次元のネットワークの辺に対応させる。斜め方向のアーク（矢）は、そのアークの位置に対応する2つのアミノ酸の類似度が割り振られる。また、縦および横方向のアークは、ギャップに対応し、ギャップを挿入するときのコストが割り振られる。このよ

うに問題を定式化すると、最適なアライメントを求めることは、このネットワーク上の最良の経路を求めることに対応する。たとえば、太いアークで表された経路が最良となったとする。この太いアークで表された経路を順に見ていくと、GとGが対応し、Kに対応するもう片方のアミノ酸はなく（つまりギャップが対応し）、FにはFが対応し、という具合に解釈することができる。結果として図6の右側にあるアライメントが得られる。

最良の経路は、具体的には、左上の端から右下の端に向かって、各ノードに至る最良経路を段階的に決定していくことにより求めることができる。各ノードの計算を行うためには、直前の段階の各ノード（ペアワイズのときは3つある）で求まった、左上端からそこまでの最良経路の評価値を参照して、今求めたいノードに至る評価値をそれぞれ計算する。そして、それらの最良値を求めて、それをそのノードまでの評価値とすればよい。直前のどのノードを選択したかという情報も記憶しておく。この操作を右下のノードまで繰返し、最後に逆向きに、選択したノードをたどれば、ネットワーク全体の最良経路を求めることができる。



$$D_{i,j}^{(1)} = Max[(D_{i,j-1} + a), (D_{i-1,j-1} + a), D_{i-1,j}] + b$$

$$D_{i,j}^{(2)} = Max[D_{i,j-1}, D_{i-1,j-1}, D_{i-1,j}] + W_{i,j}$$

$$D_{i,j}^{(3)} = Max[D_{i,j-1}, (D_{i-1,j-1} + a), (D_{i-1,j} + a)] + b$$

図 7: ダイナミックプログラミング（DP）における処理（ギャップコスト $a + bk$）

分子進化を考えると挿入や欠失が、ある程度のアミノ酸の長さでまとめて起こることから、ギャップコストはギャップの長さに依存した値にすることが望ましく、DPにも通常そ

うしたギャップコストが適用される [14]。ギャップコストに伴う計算効率を考えると、ギャップの長さ k に対して、$a + bk$ のような一次式を与えるのが妥当である [15, 16]。図 7に、ギャップコストが $a + bk$ のときの、各ノードの処理を図示した。$a + bk$ の処理を実現するには、3方向に異なる累積評価値 D を送ると考えるとよい。斜め方向は、アミノ酸の類似度 W を最良の経路の評価値に加えるだけでよいが、縦や横方向は、注目ギャップが第一ギャップである場合、ギャップコスト a (opening gap cost) を加味して最良経路を判定したうえで、ギャップコスト b (extending gap cost) を加えねばならない。

アミノ酸の名称	略記法																			
システイン	C	12	ジスルフィド結合性																	
セリン	S	0	2	小型																
トレオニン	T	-2	1	3																
プロリン	P	-3	1	0	6															
アラニン	A	-2	1	1	1	2														
グリシン	G	-3	1	0	-1	1	5													
アスパラギン	N	-4	1	0	-1	0	0	2	酸性とそのアミド											
アスパラギン酸	D	5	0	0	-1	0	1	2	4											
グルタミン酸	E	-5	0	0	-1	0	0	1	3	4										
グルタミン	Q	-5	-1	-1	0	0	-1	1	2	2	4									
ヒスチジン	H	-3	-1	-1	0	-1	-2	2	1	1	3	6	塩基性							
アルギニン	R	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6							
リシン	K	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5						
メチオニン	M	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6	疎水性				
イソロイシン	I	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5				
ロイシン	L	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6			
バリン	V	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4		
フェニルアラニン	F	-4	-3	-3	-5	-4	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9	芳香族
チロシン	Y	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10
トリプトファン	W	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0	17
		C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W

図 8: アミノ酸の類似度を与える Dayhoff マトリックス (PAM250)

アミノ酸の類似度には、図 8の Dayhoff マトリックス (PAM250) を用いるのが最も一般的である [17]。この尺度は、進化の過程で該当アミノ酸間での置換がどの程度起こりやすいかを推定し、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。近頃、新しい配列データも増えてきたので、それに基づいたマトリックスの更新も試みられている [18, 19]。Dayhoff マトリックスには、ギャップコストについての指針はないが、図 8 の値を使用するときは、$a = -7, b = -1$ くらいが目安になる。もちろん、これらのパラメータを変えることによって、ギャップの入り方が変わるのであるから、問題に応じたパラメータ調整が必要となる [20]。アライメントの結果の両端に存在するギャップ（アウトギャップ）は、配列の内部に入るギャップとは異なったコストを与えられるようにするとなお良い。配列の類似部分が水平方向に大きくズレている配列対を効果的にマッチングするには、アウトギャップをゼロ（Dayhoff マトリックスにおける中立的な値）にすると有効である。

ローカルマッチング　ローカルマッチングは、配列全体としてはそれほど似てないが、局所的によく類似している部分を見い出すのに使われ、いくつかの手法が知られている。

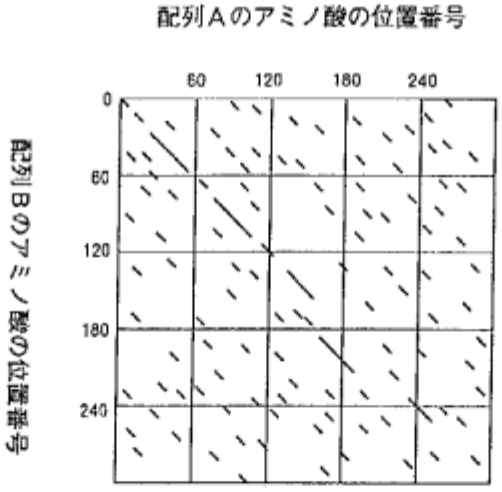


図 9: ホモロジーマトリックスの例

最も一般的なローカルマッチングは、k 個の文字の連なりであるセグメント（k-mer とか、k-tuple とも呼ばれる）を単位にして、類似セグメントを探すセグメント法である。比較される 2 本の配列のうちの部分セグメントを、順次照合していき、よく類似している配列位置の 2 次元平面に点を打っていくことで、図 9 のようなホモロジーマトリックス (homology matrix) を作成できる。対角線の方向に線が見える領域は、その 2 本の配列の類似性が局所的に高い部分に相当する。このように 2 本の配列の類似性が視覚的にわかりやすいため、ホモロジーマトリックスは人手で行うアライメントの補助としても、よく使用される表現である [21]。

$$D_{i,j}^{(1)} = Max[(D_{i,j-1} + a), (D_{i-1,j-1} + a), D_{i-1,j}, t] + b$$

$$D_{i,j}^{(2)} = Max[D_{i,j-1}, D_{i-1,j-1}, D_{i-1,j}, t] + W_{i,j}$$

$$D_{i,j}^{(3)} = Max[D_{i,j-1}, (D_{i-1,j-1} + a), (D_{i-1,j} + a), t] + b$$

図 10: 局所マッチングを行うときのノードの処理

ギャップを考慮した、精密なローカルマッチングの方法にGoad-Kanehisa 法がある [22]。この方法はDP を、局所マッチングが可能なように拡張したものである。DP では、図6に示すように、左上からの最良経路を各ノードの位置で決定していくものであるが、部分経路に注目して最良経路を決定していけば局所マッチングが可能となる。それを実現するため、図7の式を少し変形して、図10のようにする。つまり、最大和が、ある値 t（たとえばゼロ）に至らないときは、強制的に t にしてしまうのである。それと同時に、t が選ばれたときには、経路をリセットするようにしておくと、類似性の高いマッチングがなされたノードでは新たな経路が始まるので、とびとびの断片的な経路ができる。これを、左上から右下に行った後、右下から左上に向かって逆に同じ処理を行い、双方の経路の共通部分をとると、局所的な類似部分が精度よく分離できる。

マルチプルアライメントへの拡張　DP を多次元化することで、複数配列のマルチプルアライメントを行うことが、原理的には可能である。たとえば、配列3本のアライメントには、図11のように3次元DP の処理を行えば良い [23]。DP は、原理的には一度に何本の配列でも同時に処理でき、与えられた評価値における最適なマルチプルアライメントが得られるはずである。ところが n 本の配列を同時にアライメントする n 次元の DP は、概して、配列の n 乗の計算量と n 乗のメモリー量が必要であり、現実的に可能なのは3次元までである。また多次元化すると、ギャップコストの扱い方も複雑になる。カラムに1個でもギャップのある配列があったら、そのカラムの評価値はゼロにするという簡便な方法 [23] もなされているが、ギャップの多いアライメントの信頼性があまりよくない。ギャップコストの扱いは、処理時間とのトレードオフであり、議論の余地がある [24]。

アライメントに大幅なギャップが入ることはあまりないことから、ネットワークの対角部分（図6では、左上隅と右下隅を結ぶ部分）を適当な幅で残して、それ以外の部分をカットすると、処理をやや削減できる [25]。Carrillo-Lipman 法 [26] では、あらかじめ各組合せの

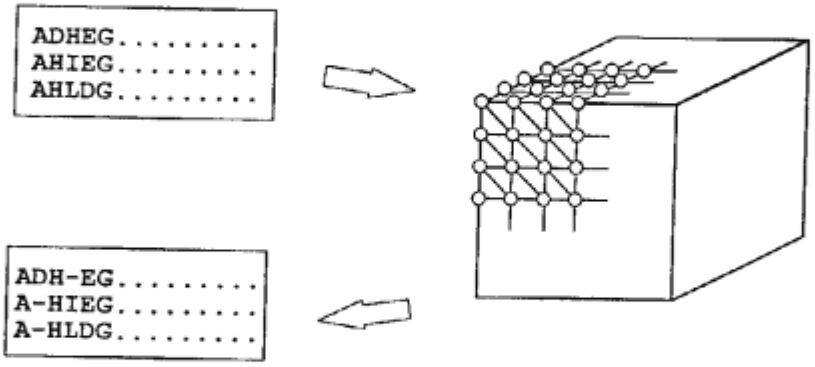


図 11: 配列３本のアライメントを行う３次元 DP の処理

ペアワイズアライメントを行い、各２次元平面上の最適経路を求めておき、ｎ次元立体のうちのカットする部分の度合を決めている。ｎ次元 DP を行うときには、ｎ次元立体中の最適経路の各２次元平面に射影した経路が、あらかじめ求めておいた最適経路の幅を越えないような範囲で処理をする。この方法により、問題によっては４次元以上の DP が可能となる。しかし、それでも、配列 10 本以上の実用的なマルチプルアライメントを行うと、膨大な計算時間を必要とし、DP の多次元化だけでは、一般のマルチプルアライメントの問題には対応できない。

一方、セグメント法を用いてマルチプルアライメントを行うことも可能である。セグメント法では、複数の配列にわたって共通した、あるいは類似した部分配列（セグメント）を見つけだし、その部分をピンで止め合わせるように縦に揃えることで、マルチプルアライメントを作る。この方法は、いわゆるローカルマッチングであり、配列群のなかに部分的に類似性の高い領域が存在するときには、その領域を貰くピン止めがすぐ発見でき、極めて有効である。しかし、その反面、類似性が低い領域を処理しようとすると、多数のピン止めが複雑に交差しあい、どれを優先すべきかという問題が発生する。これらの問題を解決するには、結局、配列全体を考慮せねばならず、グローバルマッチングとして対応せねばならない。実際、グローバルマッチングの前処理にセグメント法を使う試みもなされている [27]。

セグメント法のように、ローカルマッチングの観点からマルチプルアライメントを行う方法の開発もいくつかなされている [28, 29] が、実用的なマルチプルアライメントの方法は、おもにグローバルマッチングの観点から開発されてきた。次は、そのグローバルマッチングの諸手法を解説する。

1.2.2 組合せ法によるマルチプルアライメント

単純な組合せ法は図12に示すように、配列1と配列2、配列2と配列3、という具合に、配列を2本ずつペアワイズにアライメントし、その結果を次々と組合せてマルチプルアライメントを作る方法である[30]。ペアワイズのアライメントには、通常ダイナミックプログラミングが用いられる。この方法は画一的で高速ではあるが、素朴に適用してしまうと、結果のアライメントの品質はあまり良くない。一番大きな問題は、一緒に比較していない配列同士の類似部分が、組合せたときにズレてしまうことである。図12をみると、配列2と配列4とに同一な HIRA という部分があるが、配列2と配列4とは同時には比較されないので、マルチプルアライメントにおいて RA の部分が同一カラムに揃わないという現象が起きている。処理する配列群の類似性が低いときには、とくにこの現象が顕著に起きる。

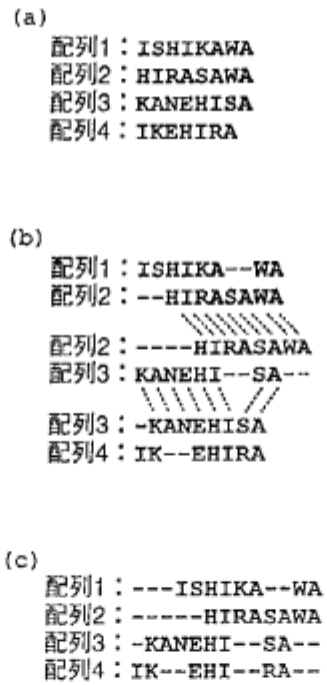


図 12: 単純組合せ法によるマルチプルアライメント

これを防ぐには2つのアプローチがあるが、ひとつは、一度に比較する配列を増やすことであり、もうひとつは、より類似した配列から順に組合せることである。

逐次組合せ法 逐次組合せ法[31]は図13(a)に示すような、4本の配列（seq1, seq2, seq3, seq4）のアライメントを行うとき、次の手順で処理をする。まず、(b) seq1 と seq2 をペアワイズDPにてアライメントし，(c) その結果と seq3 とを、グループ間のDPにてアライ

メントする。さらに、(d) その結果と seq4 とを再びグループ間 DP にてアライメントする。そうして、4本のマルチプルアライメントが得られる。5本以上のマルチプルアライメントも、この手順を繰り返すことで導くことができる。こうすると、新たにアライメントする配列は、すでにアライメントされているすべての配列と同時に評価されるので、信頼性が高い。

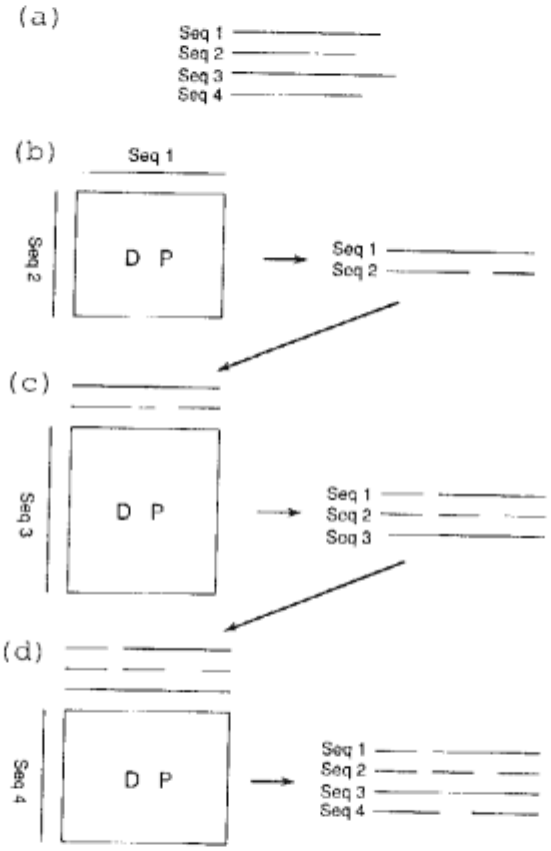


図 13: 逐次組合せ法によるマルチプルアライメント

グループ間の DP とは、図 14に示すように、配列群Aと配列群Bとを、配列群内の各々のアライメントはくずさずに、2次元の DP をするものである。グループ間 DP を行うときには、各アークに割当てられるアミノ酸の類似度が、そのアークの位置に対応する配列群Aのアミノ酸と、配列群Bのアミノ酸との類似度の総和になる。配列群Aが2本で、配列群Bが3本のときは、6通りのアミノ酸対に関する類似度の和をとることになる。類似度にはペアワイズ DP と同様に、Dayhoff マトリックスを使用するとよい。

ギャップコストの処理には注意を要する。ペアワイズ DP と違って、斜め方向のアークに

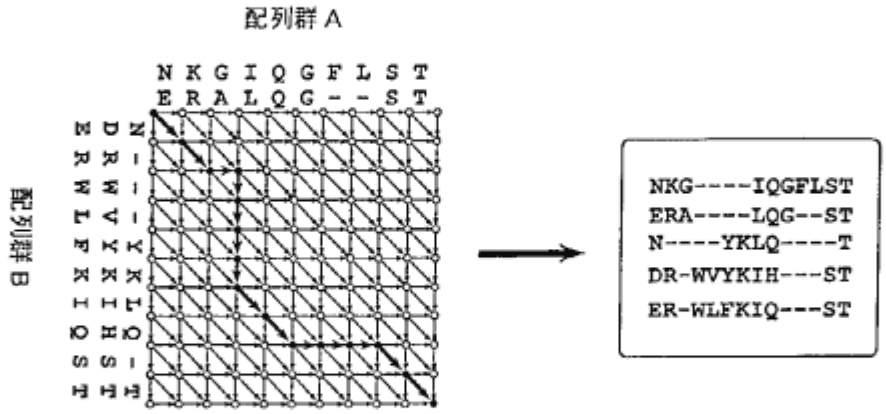


図 14: グループ間 DP によるアライメント

配列群Aのなかの1本　　LQERA^{0012}----IPG^{00}--TSF

配列群Bのなかの1本　　LQN$_{1200}$----YKIP$_{1002}$----SF

0：ギャップとはみなされない

1：第1ギャップとみなされる

2：第2ギャップとみなされる

図 15: グループ間 DP におけるギャップの判定

もギャップが対応することがあるので、注目しているギャップがどの種類のギャップかが、ノードをさかのぼらないと判断できない。図 15に示すように、配列対でギャップ同士が対応しているカラムは、普通、ギャップとはみなさない（無視する）ので、注目しているギャップが第1ギャップ ($a+b$ を加える opening gap) か、第2ギャップ以降 (b のみを加える extending gap) かは、ネットワークの経路を戻り、対応関係を調べてから判断する [32]。逐次組合せ法で使用するグループ間 DP は、必ず片側のグループの配列数が1本であるため、プロファイル化（カラムにおける各文字の発生頻度をテーブルにする方法 [33]）などの、DP 処理の効率化も可能である。

ツリーベース組合せ法　ツリーベース組合せ法は、配列間の類似性に従って描かれた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マルチプルアライメントを形成していく手法である。類似した配列から順に組合せれば、組合せ法によって得られるアライメント

の精度を大きく改善できる。なぜなら、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。

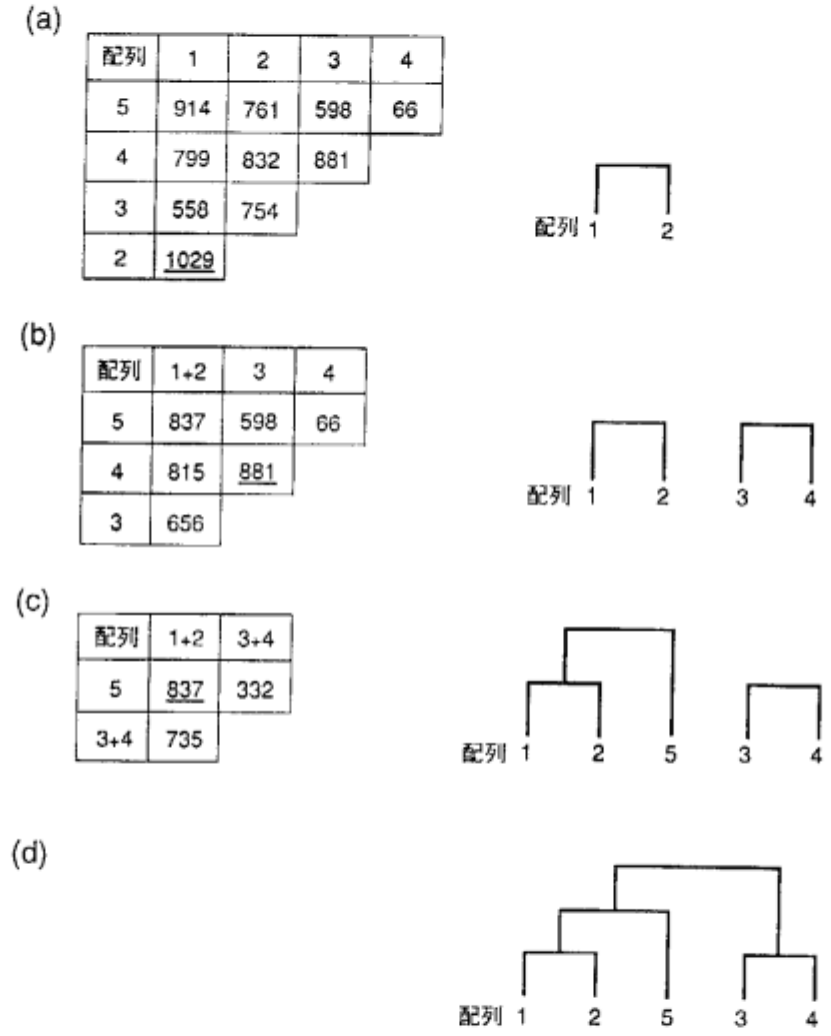


図 16: 組合せ順を決めるツリーの求め方

最初に図16を用いてツリーの作成法を説明しよう。すべての配列対についてペアワイズDP を行い、得られたペアワイズアライメントの評価値をそれぞれの配列間の類似性とする。そうして得られた類似性評価の三角表から、(a) まず、最も類似性の高い配列対（この場合は seq1 と seq2）をツリーの枝にして結線する。その結合した配列対の、残りの配列に対する類似性は、先の評価値の平均値として、三角表を更新する。そしてまた，(b) 最も類似性の高い配列対を三角表から探しだし、結線する。これを繰返す (c) と、ツリーが完成する

(d)。この手順は UPGMA[34](unweighted pair-group arithmetic average clustering) と呼ばれ、アライメントから系統樹を描くときの簡便な方法である平均距離法にも使われている[11]。

ツリーが完成したならば、ツリーの枝先から幹に向かって、枝が合わさる部分ごとに、ふたつの枝に相当するふたつの配列群を DP して、マルチプルアライメントを作っていく。そうすると最後には、全体のアライメントが得られる。その際に、ペアワイズ DP だけを行う方法と、グループ間 DP を使う方法とがある。

ペアワイズ DP だけを行う代表的な方法は、Doolittle らの、Progressive Alignment[35] である。配列群を組み合わせるときには、双方の配列群のうちから各1本をとった配列対のなかで、最も類似性の高い配列対のペアワイズアライメントに基づいて、配列群同士のアライメントを行う。たとえば、図16(d) で、seq5,1,2 と seq3,4 とのアライメントをするときには、図16(a) の表から、seq5,1,2 と seq3,4 との間でもっとも類似している組合せを選ぶ。するとこの場合、seq2 と seq4 間であるから、seq2 と seq4 との DP をとり、その結果を用いて、seq5,1,2 のアライメントと seq3,4 のアライメントとを融合させる。

この方法は、DP を行うときに一度に比較する配列が2本であるため、単純組合せ法と同様に、初期の段階で起こったアライメントの誤りが、後々まで波及して、最後に得られるマルチプルアライメントの精度を落とす危険性がある。その危険性を極力さけるために、彼らは事前のツリーを非常に注意深く描いている。ペアワイズアライメントの評価値は、厳密には比較する配列対の長さやアミノ酸の構成比に依存する部分があるので、統計的な規格化を行う必要がある。その規格化された評価値を求めるには、ペアワイズアライメントのたびに、ランダムにシャッフルした配列対の平均評価値を求めるため、ペアワイズアライメントを他に100回ほど行わねばならない。さらに彼らは、三角表で類似性の高い配列対が、評価値の上では均衡してふたつ以上ある場合には、該当部分のツリーを複数作っておいて、マルチプルアライメントをすべてのツリーについて行い、その結果、評価値の一番良いアライメントを採用する方法を提案している。たとえば図16(c) の三角表では、seq1+2 に対して、seq5 を結線するのが妥当であるが、seq3+4 の評価値もそれほど悪くないので、seq1+2 と seq3+4 を先に結線したツリーも念のため作成しておき、マルチプルアライメントを行った結果で良い方を選ぶようにする。

一方、グループ間 DP を使う代表的なツールは、CLUSTAL[36] である。グループ間 DP を使うと、事前のツリーの不正確さの問題はやや軽減される。ただし、グループ間 DP を行うときには、配列ごとに評価値に重みづけをしたい場合があることに注意を要する。DP される配列群のなかには良く似た配列同士から、あまり似ていない配列同士が混在してくる。

とくに、ツリーに従ってアライメントが完成に近づくと、その傾向が大きくなる。良く似た配列がたくさん存在するときには、それらの影響がグループ間 DP の結果に強く出るので、それらの重みを下げる場合がある [27]。また、あまり似ていない配列同士を、むりやりアライメントするのは問題があるので、そちらの重みを下げる場合もある。どちらにしろ、そうした重みづけは、配列の近縁関係を意識したもので、重みをどう付与するかは、最終的には系統樹解析のなかで考慮されるべき課題である [37]。

また、ペアワイズアライメントでなく、3次元 DP を適用した3本のアライメントの結果を、次々にツリーに沿って組み合わせるマルチプルアライメント法も提案されている [38]。

組合せ法と系統樹 マルチプルアライメントは、分子進化解析に必要な系統樹を作成する前段階としても位置づけられることを前に述べた。にもかかわらず、マルチプルアライメントで、配列の組合せ順を求めるために、あらかじめツリーを描くのは、奇異に感じられる。実際、配列の近縁関係を正しく知るには、配列がマルチプルアライメントされてなければならないし、正確なマルチプルアライメントには、配列間の類似性の推定が不可欠である。つまり、マルチプルアライメントと系統樹は、鶏と卵のような関係になっている。

理想的には、マルチプルアライメントと系統樹が同時に得られる手法がよいのであろうが、それを厳密に実現するには、祖先にあたる配列の任意性が発生するので、n 次元 DP よりもさらに膨大な処理が必要である [39]。祖先にあたる配列を近似的に推定しながら、マルチプルアライメントと系統樹が同時に得られる手法が、いくつか開発されている [40, 41]。さらに、マルチプルアライメントと系統樹とを交互に修正していく手法も提案されている [42]。この手法では、ツリーをもとに作成したマルチプルアライメントについて、対応する系統樹を作成し、その系統樹に沿って組合せ法を行って得られたアライメント結果について、また系統樹を作成するというサイクルを、もはや系統樹が更新されなくなるまで続ける。

1.2.3 並列計算機の応用

最も古くからなされている並列計算機の応用は、データベース検索である [43, 44, 45, 46]。注目するひとつの配列と、類似の配列をデータベースから見つけるには、数万のアミノ酸配列、あるいは数百万の塩基配列と類似性を評価しなければならない。前に述べたように類似性評価はペアワイズアライメントで行うのだが、並列計算機で同時にいくつものペアワイズアライメントを実行できると、類似性検索の高速化が実現できる。また、DNA の断片配列を繋ぎ合わせて一連のゲノム配列データに構成する問題（Genome Assembly）にも、大量

のペアワイズアライメントの実行が必要である。それに並列実行を導入する方法 [47] もなされている。

こうしたペアワイズアライメントに使用する、2次元 DP の内部を並列化するという検討もなされている [48, 49]。さらに、マルチプルアライメントの実現のため、高次元の DP の内部を並列化する試みもなされている [50, 51]。しかし、2次元の DP は、実行にそれほど時間がかからないため、並列化の利益があまりない。高次元の DP を分散メモリー型の並列計算機上に並列化した場合には、要素プロセッサ間のデータ参照が増え、通信のオーバーヘッドが高くなるため、並列性が十分に出ない傾向がある。

マルチプルアライメントの問題部分に並列性を見い出す研究もあった [52]。その研究では、始めローカルマッチングで、ピン止めにあたる共通配列部分を複数同定し、その間にある部分配列群をそれぞれ別の要素プロセッサで並列実行した。ローカルマッチングで、どのくらいの並列が見つかるかが問題依存である難点はあったが、通信のオーバーヘッドが低くく、問題によっては並列効果は高い。

その他には、並列計算機の利用とは言い難いが、配列マッチングの際に現れる細かな並列性を、専用ハードウェアの中で並列実行しようというアプローチもある [53, 54]。

1.3 本研究の位置付け

本節では、アライメント手法の開発の歴史を簡単に振り返った後、その歴史を踏まえて、本研究の目的を述べる。また本研究の目的が達成された場合の計算機工学的、生物学的意義を述べる。

1.3.1 歴史的背景

機械的にアライメントを行うアルゴリズムは７０年代に開発された。そのアルゴリズムの原理は、ダイナミックプログラミング (dynamic programming) であった。当時はまだ、計算機の技術が十分でなかったので、ペアワイズアライメントに応用されていたにすぎなかった。７０年代後半から８０年代にかけては、アライメントの評価値の検討や、ダイナミックプログラミングを用いたアルゴリズムの拡張の検討が加えられた。

８０年代後半になると、計算機の性能向上と、生物学分野への計算機の普及のため、ダイナミックプログラミングがマルチプルアライメントへと積極的に応用されてきた。それでも、実用的なマルチプルアライメントの問題を厳密に解くには膨大な計算時間を必要とし、主流はペアワイズアライメントを組み合わせるタイプの手法であった。この手法では、精度

が十分でなく、難しいところは、もっぱら熟練した生物学者が手作業で、アライメントの課題に取り組んでいた。

近年、Genbank[55], PIR[2] などの配列データベースが急速に膨らんでおり、解析すべき配列の数も増加してきたため、アライメント処理の機械化が不可避となってきた。それに呼応して、並列計算機などの高速計算機の登場や、高性能なユーザーインターフェースツールの開発により、マルチプルアライメントの機械化の要望に応えられる環境も整ってきた。また、計算機工学の進歩から、生物学的知識を導入した知的な解析処理も可能になりつつある[56, 57]。

1.3.2 本研究の目的と意義

本研究では、アミノ酸配列（あるいは核酸配列）のマルチプルアライメントのグローバルマッチング問題を取り挙げた。そして、この問題を、より最適な評価値が与えられるアライメント状態を探索する問題（いわゆる組合せ最適化問題）として捉え、取り組んだ。評価値には、マルチプルアライメントを構成する配列群内に存在する、すべてのペアワイズアライメントのスコアを足し合わせる、SP 体系 (Sum of Pairs[24]) を使用した。従来のツリーベース組合せ法を越える高品質のマルチプルアライメントを、SP 体系の最適化によって求めることが本研究の目的である。この際の大きな課題は、問題解決を実用的な時間内に行うべく、高能率のアルゴリズムを、いかに適用して高速な実装を実現するかである。

本研究の目的が達成された場合、生物学的な面の大きな意義がある。従来の計算機によるマルチプルアライメントは、ツリーベース組合せ法を用いていたため、精度が十分でなかった。それに対し、手作業による修正をあまり要しない高品質なアライメントが、本研究により得られる。こうした、高品質のアライメントは、配列群からモチーフなどの共通性を見い出し、構造／機能予測をする問題に、そして、系統樹解析の問題に有用である。

ただし、注意すべき点が２つある。第１に、本研究では、グローバルマッチングを採用しているため、部分的にしか類似性のない配列データのアライメントには、向かない。たとえば、DNA 断片を次々と繋ぎ合わせる Genome Assembly の問題には適用できない。第２に、スコアに SP 体系を使用していることから、類似性が極端に高い配列と低い配列が混在している問題にも向かない。系統樹解析には、よくこうした問題があるが、こうした問題を SP 体系でアライメントする場合には、類似性を事前に調べて配列を取捨選択しておく必要がある。

また、本研究により、計算機工学的に重要な知見も得られる。シミュレーテッドアニーリ

ングや、遺伝的アルゴリズムなどの最適化手法を、実用的な問題に適用する場合の課題は何か。それらのアルゴリズムを並列化などで実行時間の削減を図るときに留意すべき点は何かである。本研究は、実用規模の問題を扱う計算機応用の工学分野へも貢献する研究である。

1.3.3 本論文の構成

前節で述べた目的に従って、並列シミュレーテッドアニーリング、並列反復改善法、並列遺伝的アルゴリズムの3つの最適化手法を、それぞれマルチプルアライメントの問題に適用した研究報告が、第2～4章に記述されている。第2章のシミュレーテッドアニーリングのシステムでは、与えられたスコア体系において簡便に準最適な解が得られ、小規模な問題では従来法よりも良い解を生成する。第3章のシステムは、ダイナミックプログラミングをもとにした反復改善法を並列実装したものであり、実用規模のアライメント問題にも、従来法を上回る高品質の解を与える。第4章のシステムでは、遺伝的アルゴリズムをマルチ個体群の方式で並列実装しており、マルチ山登りの並列反復改善法よりも早期に収束に至り、かつ、最良優先探索のような局所最適解に陥ることが少ない。これらのシステム開発のうちの最大のポイントは、並列反復改善法における限定分割手法の考案であった。遺伝的アルゴリズムにおいても、独自のマルチ個体群方式を開発し、効率の良い並列探索を実現した。

続く第5，6章では、以上の技術を用いた応用システムを報告する。第5章では、ユーザーのインタラクティブな操作でアライメントの問題を解く、ワークベンチについて述べる。現在のアライメントのスコア体系は、それを最適化しても、必ずしも生物学的に意味のあるアライメントになるとは限らない。本ワークベンチでは、画面に表示されているアライメント途中の配列を指定し、並列反復改善法を用いた高速で高品質なアライメントを部分的に実行できる。さらに、アライメント途中に、ユーザー指定のアミノ酸を揃えるという制約を課したうえでの、再アライメントも可能である。そのため、ユーザーが持つ事前知識の盛り込みや、配列の特徴を捉えながらのインタラクティブなアライメント操作が実現でき、生物学の研究に有用なツールとなっている。第6章では、RNAのステム（茎）構造を考慮しながらアライメント問題を解決するシステムについて述べる。本システムは、並列反復改善法で暫定的なRNA配列のアライメントをした後に、並列シミュレーテッドアニーリングの方式で、それらのステム構造を考慮しながらアライメントの精緻化をする。立体構造の観点から信頼性の高いアライメント結果が得られるうえに、配列群が共通に持つステム構造を推定できる。

2 並列シミュレーテッドアニーリングによるアライメント

本章では、マルチプルアライメントの問題を組合せ最適化問題と捉え、並列シミュレーテッドアニーリング (simulated annealing) の手法を応用したシステム [58, 59] について、報告する。本システムは、シミュレーテッドアニーリングを温度並列の方式で並列計算機上に実装したものであり、簡便に準最適なマルチプルアライメントを得られる特徴を持つ。小規模な問題では、従来のツリーベース組合せ法よりも良い解を得られる。

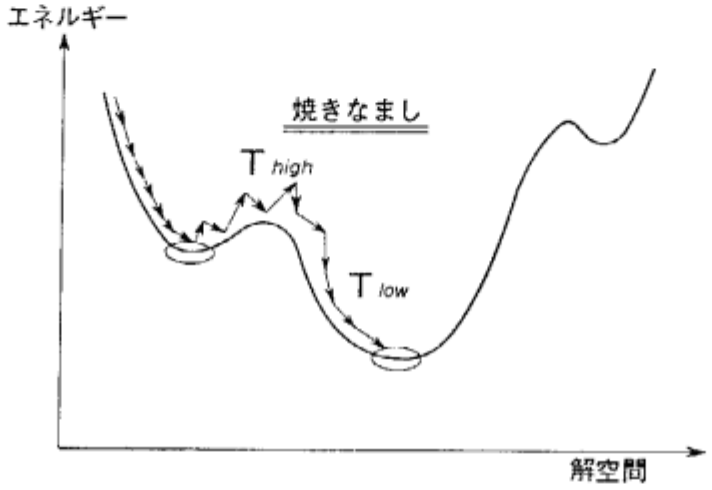


図 17: シミュレーテッドアニーリング (SA) の概念

本章の構成は次の通り。まず、並列シミュレーテッドアニーリングの手法を紹介したのち、我々が開発した、マルチプルアライメントへの適用法について解説する。次に、シミュレーテッドアニーリングの実装法を説明し、続いて、その実験結果と考察を示す。

2.1 並列シミュレーテッドアニーリングのアルゴリズム

シミュレーテッドアニーリング (以下SA と略す) のアルゴリズムは、ローカルミニマム (局所的にのみエネルギー最小な点) につかまらずに、グローバルミニマムを探索することを可能にするものである (図 17) 。古くは物理化学のシミュレーションに使われていた [60] が、計算機上の組合せ最適化問題にも応用されるようになった [61]。最近になって、生物学の問題を組合せ最適化問題と捉え、SA を応用する研究が徐々に出てきてはいる [62, 63, 64, 65] が、並列計算機を利用した本格的なものは、我々の研究が初めてである。

本節では、SA のアルゴリズムについて、その基本アルゴリズムと、我々が採用した並列

化アルゴリズムとに、分けて説明する。

```
begin
  Xo := 初期状態;
  {Tn}n=0,...,N-1 := 温度スケジュール;
  for n :=0 to N-1 do
  begin
      X'n := 変形操作を施したXn;
      ΔE := E(X'n) - E(Xn) ;
      if ΔE<0  then
          Xn+1 := X'n;
      else
          if exp(-ΔE/Tn) ≧ 区間[0,1]の乱数 then
              Xn+1 := X'n;
          else
              Xn+1 := Xn;
  end;
  解出力 Xn;
end;
```

図 18: SA の基本アルゴリズム

2.1.1 シミュレーテッドアニーリングのアルゴリズム

そもそも、アニーリングとは、物理系の焼きなまし過程を意味する。つまり、ある物質を高温から徐々に温度を下げることにより、結晶などの非常に安定な物質が得られる過程を指している。SA とは、この焼きなまし過程を模擬したアルゴリズムで、温度パラメータに依存して探索の範囲が決定される、組合せ最適化問題の解法である。この枠組では、各解の状態に、その状態を評価するエネルギー値が割り当てられた解空間を探索する。高温時においては、温度に依存したボルツマン分布に従って、エネルギーの悪化する（大きくなる）方向の探索をも確率的に許し、徐々に温度を下げていくにつれて探索範囲を絞り込む。つまり、エネルギーの良くなる（小さくなる）ような方向にしか探索しない戦略をとる。

具体的にアルゴリズムを説明すると、初期解 X_0 から順に次の様に解系列を生成していき、徐々に最適解に近い解を得ていく（図 18）。まず、ある解 X_n にランダムな微小変形を行うことで次の解の候補 X'_n を作る。最小化を目的とするエネルギー関数を $\mathbf{E}$ とすると、エネルギー値の変化は $\Delta E = \mathbf{E}(X'_n) - \mathbf{E}(X_n)$ となる。$\Delta E \leq 0$ ならば、良い変形である

ので無条件に $X_{n+1} = X'_n$ とし、また、$\Delta E > 0$ のような場合には、エネルギーが悪化する変形である。そのため、確率値として、$P = \exp(\frac{-\Delta E}{T_n})$ を採用し、温度パラメータ T_n に依存させて、次の解を決定する。つまり、確率 P で、$X_{n+1} = X'_n$ とし、確率 $(1 - P)$ で $X_{n+1} = X_n$ とする。このオペレーションをエネルギー値が収束するまで、多数回繰り返す。

ここで温度パラメータ列 $\{T_n\}$ は温度スケジュールと呼ばれ、それを適切に設定すれば、十分な時間ののち、最適解を求めることが理論的には可能である。しかし現実には、限られた時間内に準最適な解を求める場合が多い。そのための適切な温度スケジュールは、扱う問題ごとに異なり、それを設計するには少々骨がおれる。次に述べる SA の温度並列化は、温度スケジュールの設計を不要にすると同時に、より安定して高速に準最適解が得られる手法である。

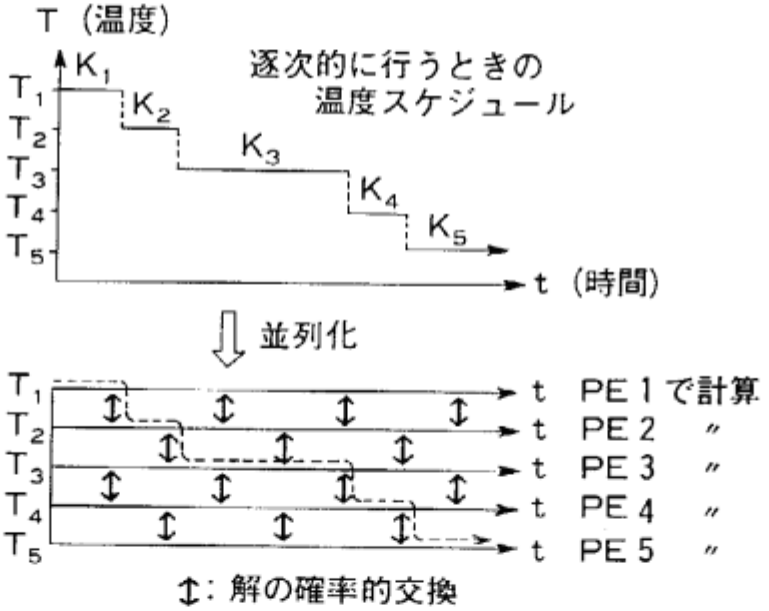


図 19: 温度並列 SA の方式

2.1.2 温度並列シミュレーテッドアニーリング

SA を並列に行うには、いくつかの方法がある。最も単純な方法は、通常の逐次的に動く SA を、利用可能な要素プロセッサ（PE）の数だけ独立に（異なる乱数で）行い、そのうちの最もよい解を選ぶものである（単純並列 SA）。それに対して、今回採用した方式は、各 PE ごとに異なる温度を割り当て、温度スケジュールがなかば自動的に行われる方式 [66] で

ある（温度並列SA）。温度並列SAは単純並列SAに比べ、同一時間で同等程度か、場合によっては同等以上の解を得られる。とくに、その時々における最良の解が刻々と得られる利点を持つ。この温度並列の方式を、以下に詳しく述べる。

各PEごとに初期解を与え、それぞれのPEにおいては担当する温度パラメータで、ひとつの解に対して一定温度のアニーリングを行う。そして、ある間隔置きに、隣接する温度を担当しているPE間で、解の交換を確率的に行う。この解交換は、より良い解はより低温のPEに移動するようになっており、それを十分な頻度で行うことにより、適当な温度スケジュールを経た解が最終的に、低い温度のPEに至る（図19）。

この解の交換確率は、温度パラメータ$T1$において得られた解のエネルギーが$E1$、温度パラメータ$T2$において得られた解のエネルギーが$E2$の時$\Delta E = E1 - E2$、$\Delta T = T1 - T2$とおけば、次式で定義される。

$$\mathbf{p}(T1, E1, T2, E2) = \begin{cases} 1 & \text{if } \Delta E \cdot \Delta T < 0 \\ \exp(\frac{-\Delta E \cdot \Delta T}{T1 \cdot T2}) & \text{otherwise} \end{cases}$$

この関数により得られた確率値に従って、解を実際に交換するか、交換を見送るかの決定を下せば、各温度に対するボルツマン分布に従う平衡状態をくずさずに、解の交換を行うことが可能になる。すなわち、従来のSAにおける理論がそのまま適用でき、十分に長い時間をかければ最適解が得られることが保証される[66]。

2.2 マルチプルアライメントへの適用

マルチプルアライメントにシミュレーテッドアニーリングを適用するためには、マルチプルアライメントの問題を、組合せ最適化問題として定式化しなければならない。つまり、各状態における評価尺度にあたるエネルギー関数と、ある解の状態から近隣の状態へと移るオペレーションである微小変形とを定義する必要がある。我々が考案した定式化について、以下に記す。

2.2.1 エネルギー関数の定義

マルチプルアライメントにシミュレーテッドアニーリングを適用した場合、取り扱う全配列にわたって同時に評価を行える利点がある。こうした評価の値を各状態に対して与えるのがエネルギー関数であり、以下の式で算出される。

$$Energy = - \sum_{i<j}^{seq.pair} \sum_{k}^{column} MatchScore(A_{ik}, A_{jk})$$

$$MatchScore(A_{ik}, A_{jk}) = \begin{cases} Dayhoff(A_{ik}, A_{jk}) & A \text{がともにアミノ酸} \\ 0 & A \text{がともにギャップ} \\ 0 & A \text{の一方がアウトギャップ} \\ -5 & A \text{がアミノ酸と opening gap} \\ -1 & A \text{がアミノ酸と extending gap} \end{cases}$$

エネルギー関数はSP体系に従って、ある配列ペアにおいて、各カラムにおけるアミノ酸ペアについてDayhoffマトリックス（図8）の値を総和し、それをすべての配列ペアについて、合計したものを使用している。エネルギー関数は小さい程よいとする慣例から、最終的に符号を反転して使用している。ギャップコストは、ギャップの長さkに対して、一次式$a+bk$を計算しているが、本章で扱う規模のデータでは、$a=-4, b=-1$にしている。ギャップ対ギャップのペアは無視される。さらに、配列の頭部や尾部にあるギャップがアウトギャップとして特別扱いされており、アウトギャップ対アミノ酸の評価を特別に指定できる。その値を通常は0にしておくことによって、配列の水平位置がどんなに食い違っていても、ペナルティが与えられることがなくなる。そのため、類似部分の位置が配列によって、先頭部分にあったり、後尾部分にあったりする難しいマルチプルアライメントが効果的に行える。

```
(a) 初期状態          1         2         3         4         5         6
          12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789
   SMRV  :-------GFILATPQTGEASKNVISHVIHCLATIGKPHTIKTDNGPGYTGKNFQDFCQKLQI-------
   MMTV  :-------YSHFTFATARTGEATKDVLQHLAQSFAYMGIPQKIKTDNAPAYVSRSIQEFLARW-------
   IAP   :-------GVMFATTLTGEKASYVIQHCLEAWSAWGKPRIKTDNGPAYTSQKFRQFCRQMDVT-------
   RSV   :-------IVVTQHGRVTSVAVQHHWATAIAVLGRPKAIKTDNGSCFTSKSTREWLARWGIAH-------
   HTLV-1:-------SGAISATQKRKETSSEAISSLLQAIAHLGKPSYINTDNGRAYISQDFLNMCTSLA-------
   HTLV-2:-------DTFSGAVSVSCKKKETSCETISAVLQAISLLGKPLHINTDNGPAFLSQEFQEFCT-------
   BLV   :-------HASAKRGLTTQTTIEGLLEAIVHLGRPKKLNTDQGANYTSKTFVRFCQQFGVSLS-------
   (energy = 877)

(b) 1回変形後         1         2         3         4         5         6
          12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789
   SMRV  :-------GFILATPQTGEASKNVISHVIHCLATIGKPHTIKTDNGPGYTGKNFQDFCQKLQI-------
   MMTV  :-------YSHFTFATARTGEATKDVLQHLAQSFAYMGIPQKIKTDNAPAYVSRSIQEFLARW-------
   IAP   :-------GVMFATTLTGEKASYVIQHCLEAWSAWGKPRIKTDNGPAYTSQKFRQFCRQMDVT-------
   RSV   :-------IVVTQHGRVTSVAVQHHWATAIAVLGRPK-AIKTDNGSCFTSKSTREWLARWGIAH------
   HTLV-1:-------SGAISATQKRKETSSEAISSLLQAIAHLGKPSYINTDNGRAYISQDFLNMCTSLA-------
   HTLV-2:-------DTFSGAVSVSCKKKETSCETISAVLQAISLLGKPLHINTDNGPAFLSQEFQEFCT-------
   BLV   :-------HASAKRGLTTQTTIEGLLEAIVHLGRPKKLNTDQGANYTSKTFVRFCQQFGVSLS-------
   (energy = 866)
```

図 20: アライメントの初期状態と微小変形

2.2.2 微小変形の定義

マルチプルアライメントの結果は内側に不定数のギャップを含むので、微小変形において、ギャップをどのように扱うかが鍵となる。我々は、配列の頭部や尾部に、あらかじめ十分な数のギャップを付加する方法をとった。たとえば、アライメントの初期状態として、まったくギャップの入ってない状態を採用するならば、図20(a)のようなアライメントが初期状態となる。

そして、状態に対する微小変形は次のように定義する。複数の配列のうちのある1本の配列に対して、任意のギャップと任意のカラム位置をそれぞれランダムに選択し、選択されたギャップを選択されたカラム位置に移動させる。そして、間の部分の配列を、移動したギャップがあった方へ1カラム分移動する。ただし、両サイドのアウトギャップはどれを選んでも同一効果であることを考慮して、アウトギャップがいくつあっても、両サイドそれぞれひとつと見なして、確率的に選択する。また、移動先のカラム位置を選ぶときも、両端のアミノ酸の位置と同じ効果をもたらすという理由から、アウトギャップの位置は除く。微小変形の例をあげると、先の初期状態において、RSVの配列が選ばれ、その配列の右端のギャップと、中央部のAなるアミノ酸のあるカラム位置（37番カラム）が選ばれた場合、図20(b)のように変形される。

また、マルチプルアライメントにはギャップが固まりで入りやすいことから、ギャップを長方形の固まりで動かすようにオペレーションを拡張すると効果的である。ここでは、前述の微小変形を、単独ギャップ移動モードと呼び、ギャップを長方形の固まりで動かす微小変形を、ブロック移動モードと呼ぶ。ブロック移動モードには、縦長ブロック移動モード、横長ブロック移動モード、縦優先矩形ブロック移動モード、横優先矩形ブロック移動モードの4種類を設けた。各ブロック移動モードとも、選ばれたギャップに隣接ギャップがなければ、単独ギャップ移動モードと同等の微小変形となる。

1. 縦長ブロック移動モードは、同一カラムに存在するギャップを一緒に移動する微小変形である。たとえば図21(a)において、カラム21番にあるギャップが選ばれ、カラム15番が選ばれたとすると、カラム21番にある全てのギャップはカラム15番へ移動し、図21(b)の状態になる。

2. 横長ブロック移動モードは、横方向に一連のギャップを一緒に移動する微小変形である。たとえば図21(b)において、カラム38番にあるギャップが選ばれ、カラム21番が選ばれたとすると、カラム38番にあるギャップから右に隣接する一群のギャップは、

カラム 21 番の位置へ移動し、図 21(c) の状態になる。（隣接ギャップは右にしか調べない。その理由は、たとえば 6 文字分のギャップは、どのギャップが選ばれるかにより、6 分の 1 ずつの確率で、1 文字から 6 文字の長さの 6 種類のギャップが生成されるからである。）

```
(a) n回変形後      1         2         3         4         5         6
          12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789
   SMRV  :----GFILATP--QTGEASK-NVISHVIHCLATIGKPHTIKTDNGPGYTGKNFQD--FCQ-KLQI----
   MMTV  :---YSHFTFAT-ARTGEATK-DVLQHLAQSFAYMGIPQKIKTDNAPAYVSRSIQE--FLA-RW------
   IAP   :----G-VMFAT-TLTGEKAS-YVIQHCLEAWSAWGKPR-IKTDNGPAYTSQKFRQ--FCR-QMDVT---
   RSV   :-------IVVTQHGRVTSV--AVQHHWATAIAVLGRPKAIKTDNGSCFTSKSTREWLA-RWGIAH----
   HTLV-1:----S-GAISA-TQKRKETSSEAISSLLQAIAHLGKPSYINTDNGPAYISQDFLN--MCT-SLA-----
   HTLV-2:---DTFSGAVSVSCKKKETSCETISAVLQAISLLGKPLHINTDNGPAFLSQEFQE--FCT---------
   BLV   :------HASAKRGLTTQTTI-EGLLEAIVHLGRPKKL---NTDQGANYTSKTFVR--FCQQFGVSLS--
   (energy = -978)                                  ^^

(b) n + 1回後      1         2         3         4         5         6
          12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789
   SMRV  :----GFILATP--Q-TGEASKNVISHVIHCLATIGKPHTIKTDNGPGYTGKNFQD--FCQ-KLQI----
   MMTV  :---YSHFTFAT-AR-TGEATKDVLQHLAQSFAYMGIPQKIKTDNAPAYVSRSIQE--FLA-RW------
   IAP   :----G-VMFAT-TL-TGEKASYVIQHCLEAWSAWGKPR-IKTDNGPAYTSQKFRQ--FCR-QMDVT---
   RSV   :-------IVVTQHG-RVTSV-AVQHHWATAIAVLGRPKAIKTDNGSCFTSKSTREWLA-RWGIAH----
   HTLV-1:----S-GAISA-TQKRKETSSEAISSLLQAIAHLGKPSYINTDNGPAYISQDFLN--MCT-SLA-----
   HTLV-2:---DTFSGAVSVSCKKKETSCETISAVLQAISLLGKPLHINTDNGPAFLSQEFQE--FCT---------
   BLV   :------HASAKRGL-TTQTTIEGLLEAIVHLGRPKKL---NTDQGANYTSKTFVR--FCQQFGVSLS--
   (energy = -990)                                  ^^

(c) n + 2回後      1         2         3         4         5         6
          12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789
   SMRV  :----GFILATP--Q-TGEASKNVISHVIHCLATIGKPHTIKTDNGPGYTGKNFQD--FCQ-KLQI----
   MMTV  :---YSHFTFAT-AR-TGEATKDVLQHLAQSFAYMGIPQKIKTDNAPAYVSRSIQE--FLA-RW------
   IAP   :----G-VMFAT-TL-TGEKASYVIQHCLEAWSAWGKPR-IKTDNGPAYTSQKFRQ--FCR-QMDVT---
   RSV   :-------IVVTQHG-RVTSV-AVQHHWATAIAVLGRPKAIKTDNGSCFTSKSTREWLA-RWGIAH----
   HTLV-1:----S-GAISA-TQKRKETSSEAISSLLQAIAHLGKPSYINTDNGPAYISQDFLN--MCT-SLA-----
   HTLV-2:---DTFSGAVSVSCKKKETSCETISAVLQAISLLGKPLHINTDNGPAFLSQEFQE--FCT---------
   BLV   :------HASAKRGL-TTQTT---IEGLLEAIVHLGRPKKLNTDQGANYTSKTFVR--FCQQFGVSLS--
   (energy = -1227)                                 ^ ^   ^^

(d) n + 3回後      1         2         3         4         5         6
          12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789
   SMRV  :----GFILATP--Q-TGEASKNVISHVIHCLATIGKPHTIKTDNGPGYTGKNFQDFCQ-KL--QI----
   MMTV  :---YSHFTFAT-AR-TGEATKDVLQHLAQSFAYMGIPQKIKTDNAPAYVSRSIQEFLA-RW--------
   IAP   :----G-VMFAT-TL-TGEKASYVIQHCLEAWSAWGKPR-IKTDNGPAYTSQKFRQFCR-QM--DVT---
   RSV   :-------IVVTQHG-RVTSV-AVQHHWATAIAVLGRPKAIKTDNGSCFTSKSTREWLA-RWGIAH----
   HTLV-1:----S-GAISA-TQKRKETSSEAISSLLQAIAHLGKPSYINTDNGPAYISQDFLNMCT-SL--A-----
   HTLV-2:---DTFSGAVSVSCKKKETSCETISAVLQAISLLGKPLHINTDNGPAFLSQEFQEFCT-----------
   BLV   :------HASAKRGL-TTQTT---IEGLLEAIVHLGRPKKLNTDQGANYTSKTFVRFCQQFG--VSLS--
   (energy = -1300)                                 ^ ^   ^^
```

図 21: ブロック移動モードによる微小変形

3. 縦優先矩形ブロック移動モードは、同一カラムに存在する全てのギャップに対し、隣接する同じ長さの一連のギャップを移動する微小変形である。たとえば図 21(c) において、カラム 56 番にあるギャップが選ばれ、カラム 62 番が選ばれたとすると、カラ

ム56番にある全てのギャップから隣接する一群の矩形状のギャップが、カラム62番の位置へ移動し、図21(d)の状態になる。

4. 横優先矩形ブロック移動モードは、横方向に一連のギャップと、同じ長さのギャップが他の配列にも同じ位置にあった場合、それらを一緒に移動する微小変形である。選択ギャップが完全な矩形状のギャップ領域の内部にあった場合、横優先矩形ブロックは、縦優先矩形ブロックと同じであるが、一般には、ギャップ領域の形状は不定形であるので、両者は異なったブロック形状を指定する。

2.3 実験と結果

前節で述べた温度並列SAと、単純並列SAとを、並列推論マシンMulti-PSI[67]の上に実装し、2種類の比較実験を行った。

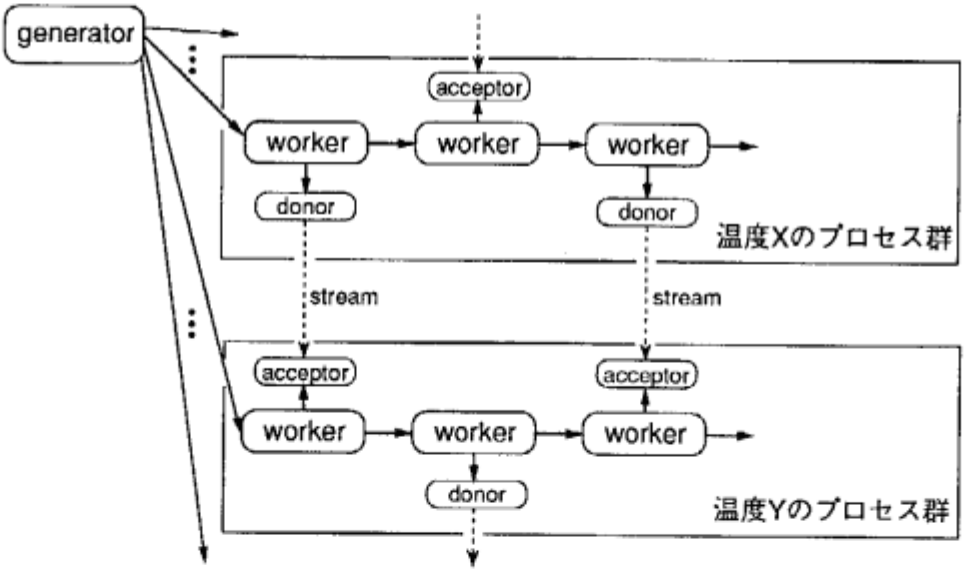


図22: KL1による温度並列の実装概念

2.3.1 並列計算機への実装

Multi-PSI上に、その上で動作する並列論理プログラミング言語であるKL1[68]を用いて、温度並列SAと、単純並列SAとを実装した。Multi-PSIとは、64台の要素プロセッサ（PE）が8×8のメッシュ構成で結合された分散メモリマシンで、それぞれのPEは、80MBのメモリを有している。KL1は、形式上はPrologに似た論理型言語であるが、プロセス指向の

プログラミングが可能な並列プログラミング言語である。KL1 では、並列実行時の同期処理が言語の枠組に含まれており、並列プログラム開発が効率良く行える。Multi-PSI は、KL1 を高速に実行するための専用マシンであり、データの型をファームウェアレベルで管理するなどの、独自のアーキテクチャを持っている。

図 22は、KL1 で実装した温度並列の概念を表している。generator のプロセスが、各温度に対応するプロセス群を生成し、それぞれのプロセス群が微小変形を繰り返す worker を中心に、並列に動作する（プロセス生成の具体的プログラミングなどは、3.2.2 を参照されたい）。プロセス群間では、共有変数をストリームにして、情報の授受を行う。図 23は、解交換の部分の KL1 プログラムであるが、共有ベクター変数 Message に donor が値を入れ、acceptor がそれを読むようになっている。donor と acceptor は異なる PE で実行されているので、実際には通信が発生する。acceptor での実行結果は、やはり同一の変数を介して戻される。

```
donor(TempX,EnergyX,EnergyX_out,AlignX,AlignX_out,PEX,PEX_out,Stream,Stream_out) :-
        Message = {TempX,EnergyX,EnergyX_out,AlignX,AlignX_out,PEX,PEX_out},
        Stream = [Message|Stream_out].

acceptor(TempY,EnergyY,EnergyY_out,AlignY,AlignY_out,PEY,PEY_out,Stream,Stream_out,Random,
                                                                        Random_out) :-
    Stream = [Message|Stream_out],
    Message = {TempX,EnergyX,EnergyX_out,AlignX,AlignX_out,PEX,PEX_out} |
        decision(TempX,TempY,EnergyX,EnergyY,Random,Random_out,ANSWER),
    ( ANSWER = exchange -> (EnergyX_out = EnergyY,
                            EnergyY_out = EnergyX,
                            AlignX_out = AlignY,
                            AlignY_out = AlignX,
                            PEX_out = PEY,
                            PEY_out = PEX);
      ANSWER = nochange -> (EnergyX_out = EnergyX,
                            EnergyY_out = EnergyY,
                            AlignX_out = AlignX,
                            AlignY_out = AlignY,
                            PEX_out = PEX,
                            PEY_out = PEY)
    ).
```

図 23: KL1 による解交換のプログラム

Multi-PSI のような疎結合型マシンは、大規模化が容易な反面、PE の性能に比べた通信性能が低い。プログラミングにあたっては、その点に注意しなければならない。温度並列 SA の実装で PE 間の通信が発生するのは、解交換のときであるから、その通信を最小限に抑える工夫が必要であった。そこで、データ量の多い解を PE 間で交換するのではなく、逆

にPE間で温度を交換するようにしている。図23のプログラムでは、decisionがexchangeの場合、エネルギーとアライメント解とPE番号が、プロセス間で交換されている。各プロセスは、一定の温度を持っているので、PEから見ると、プロセス（つまり温度）が交換されたこととなる。

計算時間を比較すると、微小変形の時間よりも、そのあとの評価（エネルギー値算出）に数十倍の時間がかかる。評価時間の短縮のため、$\sum_k^{column} MatchScore(A_{ik}, A_{jk})$ の値をテーブルにとっておき、微小変形で変更されたところのみの値を更新し、エネルギー計算するように工夫している。

温度並列SAでは、高温から低温まで63個の異なる温度を、63台のPEに割り当てた。残りの1台のPEは、SA実行時の実績データをディスクに格納する作業や、実行中の内容をインタラクティブにモニタする作業などの、入出力管理の目的で使用した。図24は、モニタツールの表示例であるが、このようなツールで実行中の解がモニタできる。上部に表示されているアライメントが、モニタ解であり、その解のエネルギー値がENERGYウィンドウに、その解の温度履歴（小数点は省略されている）が中央のウィンドウに表示されている。また、下部には、最良エネルギーと実行時間の履歴が表示されている。通常は、最低温度の解を常時モニタすることで、所望の解が得られているかどうかを判断するのであるが、MONITORウィンドウで指定すると、任意の温度の解が表示できる。

図25では、十分な時間を経たあとの、高温（99.9度）、中高温（47.8度）、中低温（27.0度）のモニタ解を表示した。高温では、ギャップがほとんどランダムに配列中に挿入されている。中高温では、配列の中央部分で、類似部分が揃い始めているのがわかる。さらに、中低温では、周辺部を除いて、類似部分が揃っている。このような解のモニタで、高温でシャッフルされた解が低温で落ち着く様子が良くわかる。63個の温度の割り付けは、こうした作業を通して、99.9度から1～2割ずつ等比的に減温させて決めており、最低温度は0.3度にしている。

単純並列SAは、逐次SAの処理を63台のPEで個々独立に行ったものから、各時点での最も良い解を選び、全体としての解とするものである。逐次SAの処理は、あらかじめ指定した回数の微小変形を最高温の99.9度で行ったならば、次の温度に下げるという操作を、温度並列SAと同様な63個の温度にわたって、最低温の0.3度まで行うように実装した。単純並列SAでは、入出力用のPEが、その時点での最良解を探すべく、定期的に情報を集めるときにのみ通信が発生する。通信量は極めて少なく、通信のオーバーヘッドの問題はない。

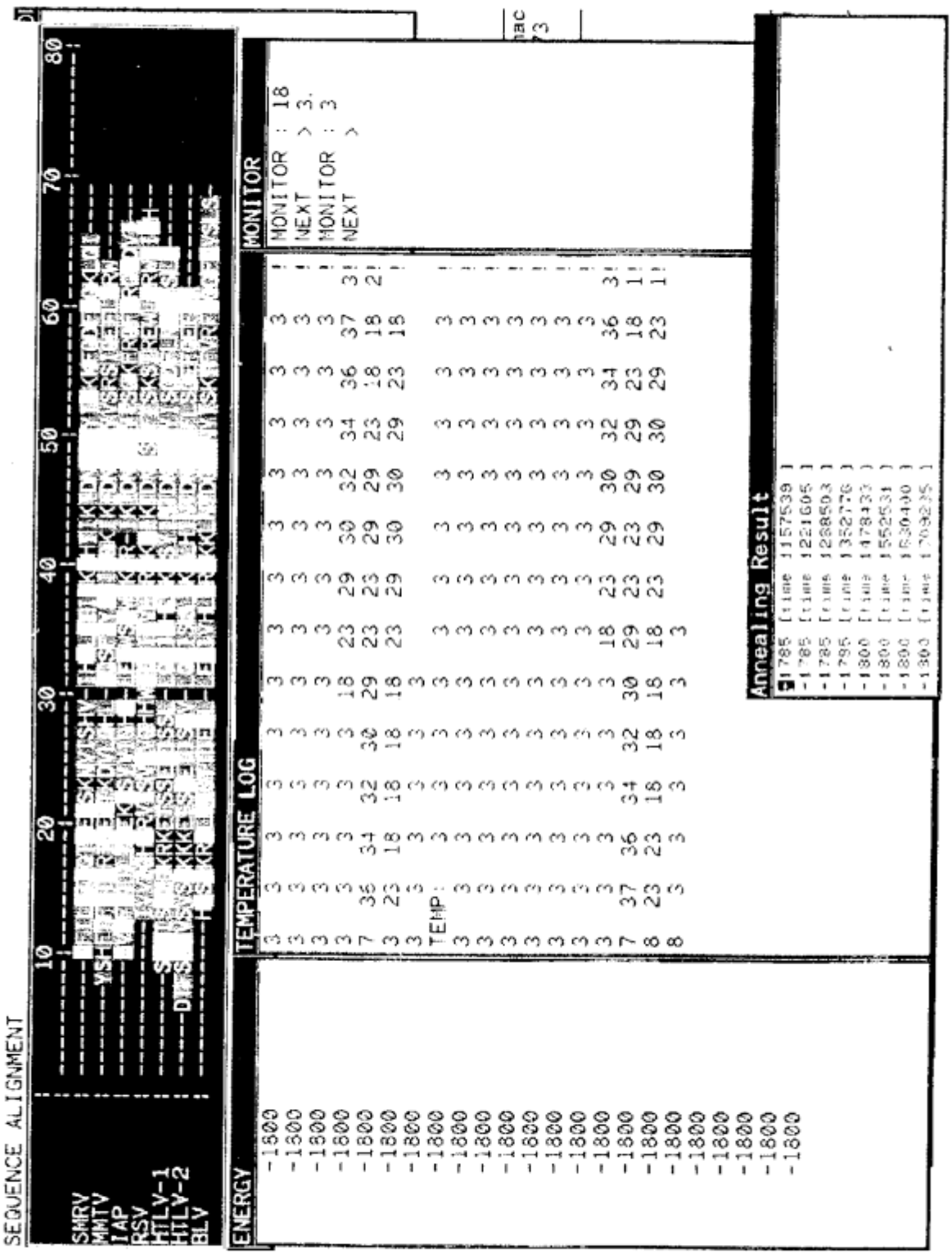


図 24: 温度並列 SA のモニタツール

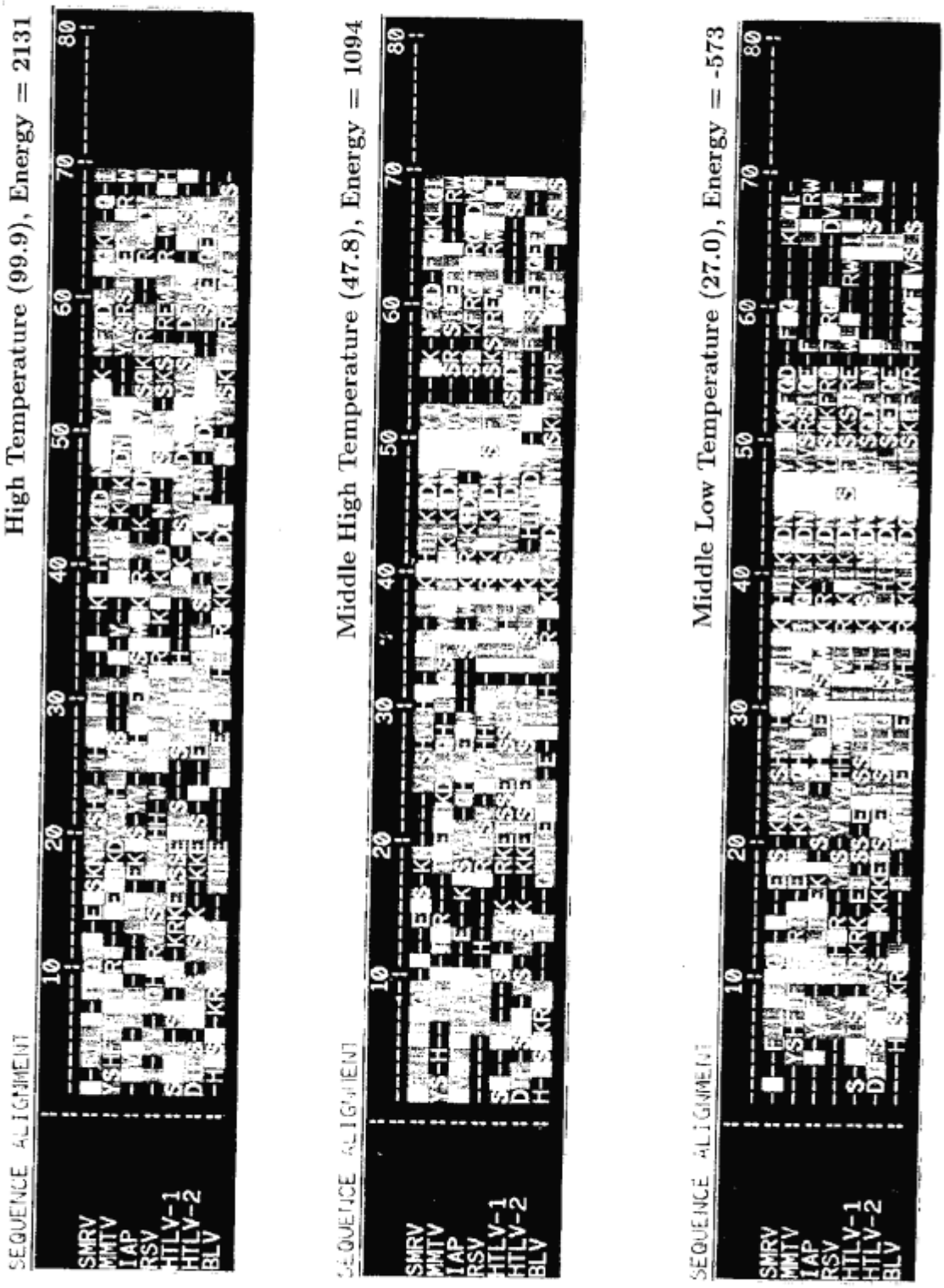


図 25: 高温時の解の状態

2.3.2 ツリーベース組合せ法との比較実験

図 20(a) を初期状態とした問題について、SA を実行した結果を図 26に示した。それぞれ、PE63 台の温度並列 SA を各 PE 当り 2 万回の微小変形を行ったときのエネルギー履歴 (a)、PE63 台の単純並列 SA を各 PE 当り 1 万回の微小変形を行ったときのエネルギー履歴 (b)、逐次 SA を 2 万回の微小変形を行ったときのエネルギー履歴 (c) である。乱数系列を変えて繰り返し試行しており、(a), (b) は 2 回の試行の平均、(c) は 63 回の試行の平均である。水平ライン (d) は、グループ間 DP を用いたツリーベース組合せ法によって得られた結果のエネルギー値である。このソフトウェアも KL1 で記述し、Multi-PSI 上のひとつの PE で実行させたものであり、実行には約 3 分を要した。

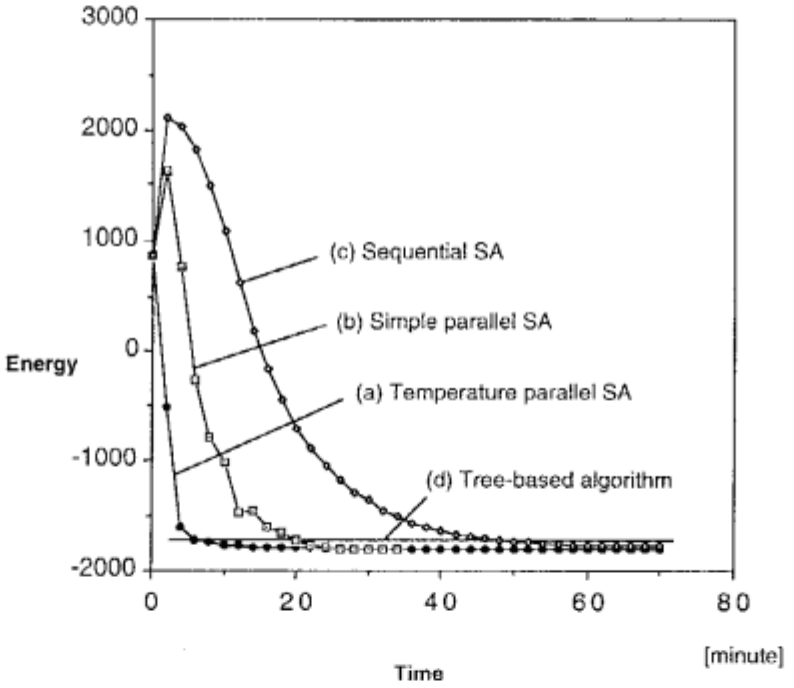


図 26: SA 処理の履歴比較

この実験結果から次のことがわかる。

- 温度並列 SA は、初期状態から単調に良い解を提示しているが、他の SA は、いったんエネルギー値が悪化してから、良い解へと収束していく。
- 温度並列 SA と単純並列 SA は、ともに 30 分以内に最良の解に達している。その後 30 分以上温度並列 SA を続けても、この解より良い解が得られていないところをみると、

この解（図27(a)）は、最適解である可能性が高い。

- ツリーベース組合せ法は比較的高速に解が得られるが、その解（図27(b)）のエネルギー値は、最適と思われるエネルギー値よりも悪い。

- 逐次SAは、1時間を越えてアニーリングしても、最適解と思われる状態に必ずしも至らない。局所最適解に捕われてしまったためだろう。しかし、最終のエネルギー値は、平均するとツリーベース組合せ法で得られるものよりは良い。

この実験では、温度並列SAと単純並列SAとのパフォーマンスの差異は、ほとんど見られない。これは単純並列SAの温度スケジュールの設計が、割合に良かったためであろう。もし、もっと短い時間の温度スケジュールの設計をしてしまうと悪い局所最適解に捕われてしまっただろう。このような場合、単純並列SAでは、再び始めから適当な、もっと長い温度スケジュールでやり直さねばならない。それに対し、温度並列SAは、つねに最低温度のPEをモニタしていれば、そこにより良い解が次々と現れるので、単に、もうしばらくの時間、待てばよい。こうした、適当な温度スケジュールがわからないときに、温度並列SAの長所がでてくる。

```
(a)
   SMRV  :------GFILATPQTGEASKNVISHV-IHCLATIGKPHTIKTDNGPGYTGKNFQDFCQKLQI-------
   MMTV  :----YSHFTFATARTGEATKDVLQHL-AQSFAYMGIPQKIKTDNAPAYVSRSIQEFLARW---------
   IAP   :------GVMFATTLTGEKASYVIQHC-LEAWSAWGKPR-IKTDNGPAYTSQKFRQFCRQMDVT------
   RSV   :----------IVVTQHGRVTSVAVQHHWATAIAVLGRPKAIKTDNGSCFTSKSTREWLARWGIAH-----
   HTLV-1:-----SGAISATQKRKETSSEAISSL-LQAIAHLGKPSYINTDNGPAYISQDFLNMCTSLA--------
   HTLV-2:--DTFSGAVSVSCKKKETSCETISAV-LQAISLLGKPLHINTDNGPAFLSQEFQEFCT-----------
   BLV   :----------HASAKRGLTTQTTIEGL-LEAIVHLGRPKKLNTDQGANYTSKTFVRFCQQFGVSLS----
   (energy = -1800)                                ^ ^    ^^

(b)
   SMRV  :----GFILATPQTGEASKNVISHVIH---CL---ATIGKPHTIKTDNGPGYTGKNFQDFCQKLQI-----
   MMTV  :--YSHFTFATARTGEATKDVLQHLAQ---SF---AYMGIPQKIKTDNAPAYVSRSIQEF---LARW----
   IAP   :----GVMFATTLTGEKASYVI-Q--H---CLEAWSAWGKPR-IKTDNGPAYTSQKFRQFCRQMDVT----
   RSV   :-------IVVTQHGRVTSVAVQH--HWATAI---AVLGRPKAIKTDNGSCFTSKSTREW---LARWGIAH
   HTLV-1:---SGAISATQKRKETSSEAISSLLQ---AI---AHLGKPSYINTDNGPAYISQDFLNMCTSLA------
   HTLV-2:DTFSGAVSVSCKKKETSCETISAVLQ---AI---SLLGKPLHINTDNGPAFLSQEFQEFCT---------
   BLV   :-------HASAKRGLTTQTTIEGLLE---AI---VHLGRPKKLNTDQGANYTSKTFVRFCQQFGVS-LS-
   (energy = -1725)                                  ^ ^    ^^
```

図27: 最終解の比較：(a) SA、(b) ツリーベース

図27(a)、(b)を比べると、SAによるアライメント結果の、ツリーベース組合せ法に対する優位性が良くわかる。SAでは小さな変形を繰り返すので、誤ったギャップの入り方でも修正されて、最終解へ至る。一方、ツリーベース組合せ法では、始めのペアワイズアライメントの段階で入ったギャップが最後まで残ってしまい、同じギャップコストを用いても、

SA の解よりギャップがたくさん入る。その結果、アライメントが長くなりがちで、類似性の低い部分がアライメントされない。

```
CBP :-AVYVVGGSGGW--TFNTE-----SWPKGKRFRAGDILLFNYNPSMHNVVVVNQGGFS
Sc  :-TVYTVGDSAGWKVPFFGDVDYDWKWASNKTFHIGDVLVFKYDRRFHNVDKVTQKNYQ
Pc  :IDVLLGADDGSL--AFV---------PSEFSISPGEKIVFKNNAGFPHNIVFDEDSIP

CBP :TCNTPAGAKV------YTSGRD-QIKLP-KGQSYFICNFPGHCQSGMKIAVNAL---
Sc  :SCNDTTPIAS------YNTGNN-RINLKTVGQKYYICGVPKHCDLGQKVHINVTVRS
Pc  :SGVDASKISMSEEDLLNAKGETFEVALSNKGEYSFYCSPHQGAGMVGKVTVN-----

(energy = -330)
```

図 28: 3本アライメントの最適解

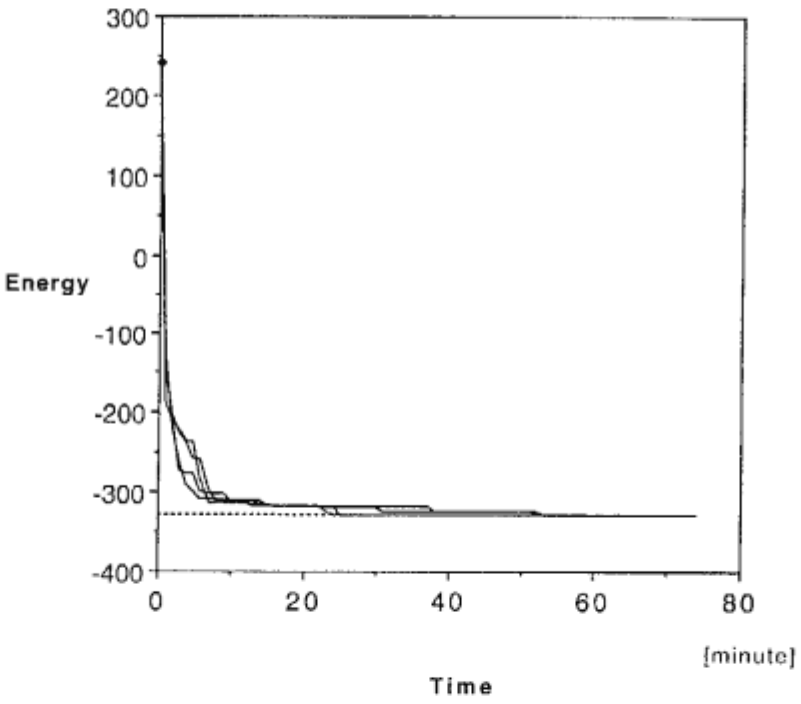


図 29: 3本アライメントの最適化履歴

2.3.3 3次元ダイナミックプログラミングとの比較実験

この実験では、最適解が判明している問題において、温度並列SA が首尾良く最適解を与えるかどうかを確かめる。図 28は、 Murata らが使った配列データ [23] を、 KL1 で組んだ3次元 DP のソフトウェア [50] でアライメントしたものである。 DP を用いているので、

この解は最適解であることが保証されている。図28は2段になっているが、一連の長いひとつのアライメントである。

図28の3本の配列を、内側にギャップを含まない初期状態から、温度並列SAを実行した。図29は、温度並列SAを5回行った結果のエネルギー履歴を示したものである。点線が、最適解のエネルギーレベルであるが、5回とも55分以内にそのレベルに達していることがわかる。アライメントの結果も図28と同一であった。これにより、温度並列SAが十分な時間があれば、最適解を与えることが明らかとなった。

3次元DPの実行は、Multi-PSI上のひとつのPEで約74分を要した。一見、温度並列SAは、3次元DPより高速に最適解を与えるように思えるが、3次元DPは、あらゆる解の可能性をチェックした結果として、最適解を保証しているのであるが、温度並列SAは、仮に最適解が求まっていても、それが最適解であることがわからない。ゆえに、これらの実行時間を安易に比較はできない。

2.4 考察

我々はマルチプルアライメントの課題を、シミュレーテッドアニーリング（SA）の手法で解決するシステムを、並列計算機上に構築した。本システムは、SAを温度並列で実装しており、3本のアライメントで最適解を与えること、小規模なアライメントで従来のツリーベース組合せ法の解より、良い結果を与えることが確認された。本節では、さらに、微小変形の定式化の重要性と、アライメントの問題にSAを応用する際の有効性とを検討する。最後に、今後の課題を述べる。

2.4.1 微小変形の重要性

研究の当初、微小変形は単純ギャップ移動モードしか実装してなく、システムはうまく動作していなかった。図30は、温度並列SAにブロック移動モードを導入した前後のエネルギー値履歴の違いを表しているが、ブロック移動モードの導入により、飛躍的に性能が向上しているのがわかる。SAは微小変形の組合せで問題が解けるといえども、長大な実行時間を少しでも削減するには、問題の性質を調べ、適切な規模の微小変形を導入すべきである。

SAでは、理論的な正当性を保存するため、そして、高温でのシャッフルを十分に行うために、対称性の前提がある。それは、ある微小変形があれば、その逆の微小変形を必ず設けることである。最近では実用性の観点から、その対称性の前提をくずした、特殊な微小変形を導入する傾向も見られる。どのみち実用的な時間内に最適解が得られそうもないなら。満

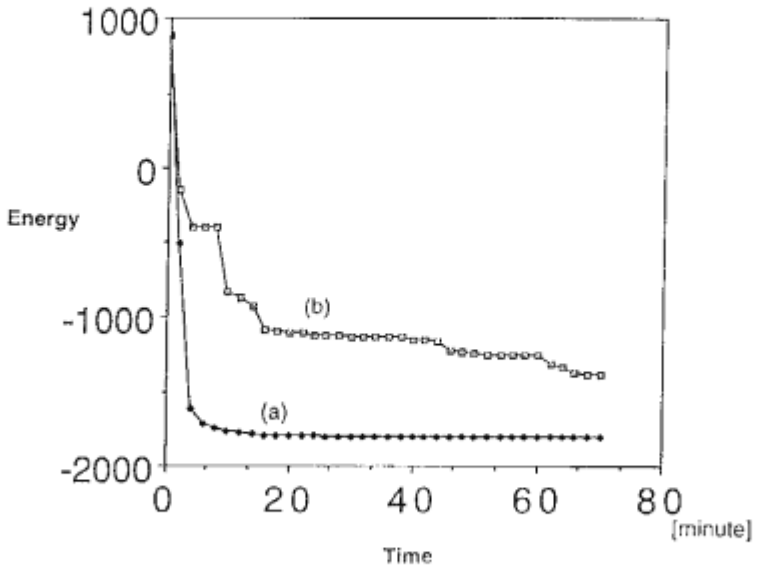


図 30: ブロック移動モードの効果： (a) ブロック移動あり、 (b) なし

足できるレベルの解が少しでも早く求まれば良いという動機からである。

対称性の前提をくずすと、もはやSA と呼ぶのは適当でなくなってしまうのかも知れないが、このアライメントへの応用でも、図31に示すような微小変形を導入により、解探索の効率が向上すると予想される。 (a) は、揃い始めた文字を強制的に揃えてみる微小変形、 (b) は、互い違いになっているギャップを縦に揃えてみる微小変形である。我々はこうした微小変形を推論ルールと捉えたアプローチも提案している [69]。

2.4.2 シミュレーテッドアニーリングの有効性

今回のSA の応用では、小規模なアライメント問題については有効性が実証できたが、実用的な規模の問題ではどうであろうか。少し荒っぽい計算をして、推定してみよう。図 27(a) の問題には、アミノ酸が 385 個あるが、実用規模のアライメント問題では、数千個のアミノ酸が存在する。アミノ酸の数にして、 10 倍である。するとアライメントに必要なギャップの数も 10 倍程度に増えよう。微小変形は基本的にギャップの移動であるから、移動するものと移動先がとが、それぞれ 10 倍ずつになると、微小変形の種類は、その積で 100 倍程度となる。そうなると、収束に必要な微小変形回数も大雑把ながら、少なくとも 100 倍程度は必要と想像できる。となると、今回の問題が 1 時間で解けていたとすると、実用規模の問題の実行時間は、 100 時間以上になると予想される。

実用規模の問題をSA で解くのに必要な、長大な実行時間を待てない場合、 SA を他の手

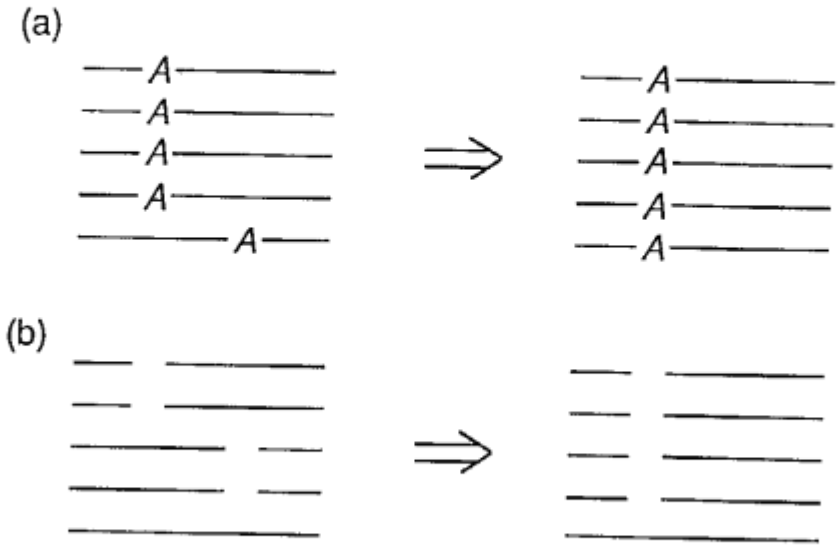


図 31: アライメントを向上させる微小変形の例

法で得られた解の修正に使うと良い。初期状態として近似解を入れ、中低温からアニーリングするのである。このアニーリングには、その近似解に合わせた温度スケジュールが必要であるが、こうした場合にも、温度並列SA は温度スケジュールの吟味が不要で、気軽に使用できる。実際、ツリーベース組合せ法で得られた図 27(b) の状態を初期状態として、温度並列SA を適用すると、約 14 分で図 27(a) の状態へ至った。

今回の実験で使った評価値は、比較的単純な体系であったが、SA をアライメントに応用する場合、複雑な体系を導入できる利点がある。すでにわかっているモチーフが揃ってきたら高得点を与えるとかの、生物学的な知見を評価値に反映させることもできる。第 6 章では、立体構造を考慮した評価値を使用し、単純な DP の組合せでは解けないアライメントの問題を、SA で解決した例を報告する。

2.4.3 今後の課題

本システムの効率化について、今後の課題を述べておこう。まず、並列性の面からの課題は、並列の規模があがった場合の効率的な PE の利用である。多くの PE を異なる温度に割り当てていくと、高温から低温に解が移動するのに時間がかかってくる。中高温で偶然良い解が生まれても、低温へ至る間に悪くなってしまうことも想像できる。ひとつの対処法は、低温領域では同じ温度を担当する PE を増やし、複数の解のアニーリングをすることである。高温領域では、同一 PE で広い解空間を探索しているのに対し、低温領域では、局所

的な探索をしているので、低温領域に多くのPEを割り当て、複数のPEでいろいろな場所を探索すると効率が上がるだろう。また、隣接する温度間だけでなく、間の温度を飛ばしたPE同士で解の交換を試みるのも、効果が期待できる。さらに、温度並列以外の並列性の検討も必要かも知れない。我々は、同一温度にある複数のPEが持つ解のエネルギー分布から、温度平衡を検定してSAを行う手法も試行した[70]。

微小変形にかかわる時間を極力減らすという工夫も重要である。とくに、評価時間の削減と、無駄な微小変形の低減は課題である。高温領域では、そもそも正確な評価は必要ではないので、簡単に概算する計算式を開発するとよい。低温では、ほとんどの微小変形は解を悪化させるので、無駄になる。微小変形の生成にあたって、低温では、小さな変形がより高い確率でなされるようにすると、無駄な微小変形を少なくできるだろう。

本システムを、アライメント以外の生物学的問題に応用するのも、興味深い課題である。我々は、タンパク質フォールディングの問題に応用してみたが、問題規模が大き過ぎて、あまりうまくいかなかった。ただ、立体構造のコアあたりの内外性指標が、目標構造とわずかに似たものとなった[71]。ほかに、Genome Assemblyの問題や系統樹描画の問題への応用が考えられよう。いずれにせよ、問題の規模を正しく推定し、適切な微小変形を設計することが成功の鍵である。

3 並列反復改善法によるアライメント

本章では、反復改善法を並列化し、実用規模のマルチプルアライメントの問題を解決可能としたシステム [72, 73] について、報告する。本システムは、反復改善法のアライメントアルゴリズムを並列計算機上に実装したうえに、大きな問題まで対応可能にする限定分割手法を導入したものである。実用規模のマルチプルアライメントの問題にも、従来のツリーベース組合せ法を上回る高品質の解を与える。

本章の構成は次の通り。まず、反復改善法のアライメントアルゴリズムを解説したのち、我々が開発した並列化手法と、その並列計算機上への実装について紹介する。次に、独自のアイデアである限定分割手法を説明し、実用規模のマルチプルアライメント問題への適用結果を述べる。最後に、考察を示す。

3.1 反復改善法によるアライメント

反復改善法のルーツは、Barton らの逐次組合せ法によるアライメントの論文 [31] に遡れる。彼らは、わずか9行のコメントであったが、DP を用いてアライメント結果を改善する着想を、論文に記していた。ギャップのないアライメントの初期状態から反復改善する、本来の反復改善法は、Berger ら [74] が提案した。その後、後藤 [32] が、ギャップコスト計算を忠実に行う方法で、再提案した。音声認識でよく使われている隠れマルコフモデルを、マルチプルアライメントに応用した手法 [75, 76] も、広い意味では、反復改善法の一種とみなすことができる [77]。

本節では、反復改善法について、そのアルゴリズムの解説をした後、具体的な問題に適用したときの性能と問題点を述べる。

3.1.1 反復改善法のアルゴリズム

反復改善法は、次に示す手順で、配列群間の DP を反復的に適用することによりマルチプルアライメントを行う（図 32）。

1. ギャップのない配列群を初期アライメントとする。
2. 初期アライメントを、ランダムに2つのサブアライメントに分割する。
3. サブアライメント同士のアライメントを、DP を適用して最適化する。
4. その結果を、次の改善サイクルの初期アライメントとする。

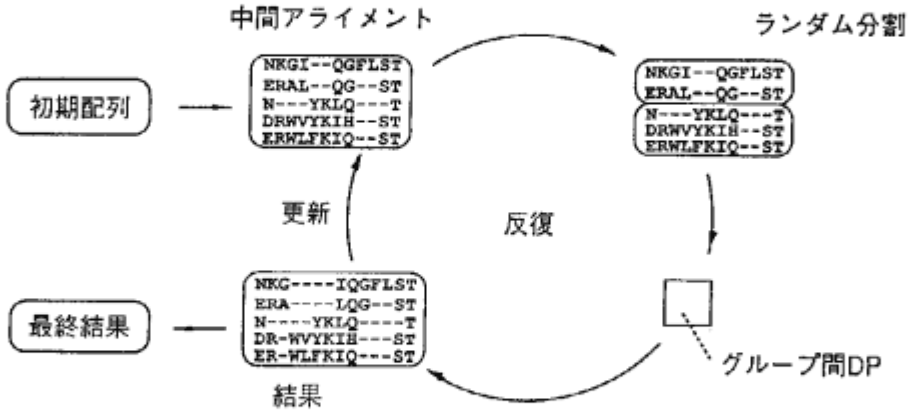


図 32: 反復改善法のアルゴリズム

収束条件に適当なサイクル数を決め、そのサイクル数にわたって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。

反復改善法の基本的アイデアは次のところにある。マルチプルアライメント全体の評価値が、それに含まれるすべての配列対の評価値の和（配列ごとに重みがついていてもよい）で与えられるという条件のもとでは、あるマルチプルアライメントを任意に2つの配列群に分け、それらの間で互いに2次元DPを行うと、得られるアライメントは、もとのマルチプルアライメントより、スコアが良い（少なくとも等しい）。つまり、配列群間のDPを繰返すと、マルチプルアライメントのスコアが初期状態から次第に良くなっていくのである。しかし、次第に良くなっていくといえども、どこまでも単調に良くなるわけではない。乱数の与え方によりアライメントが不適当な方向へ進むと、比較的悪い局所最適値に陥ってしまうこともある。

3.1.2 反復改善法の性能と問題点

反復改善法を用いると、得られるマルチプルアライメントの品質が大幅に良くなる。図33では、ツリーベース組合せ法で解いた解と、反復改善法で解いた解を比較している。配列群はキナーゼと呼ばれる一群の酵素[78]の、触媒部位の先頭部分である。キナーゼは細胞内の情報伝達機構で重要な役割を担っている代表的なタンパク質である。アライメントを見ると、ツリーベース組合せ法の解は、全体にギャップがたくさん入ってしまって、アライメントが長くなったうえに、中央にあるモチーフA.K（ATPを結合する部位の一部）については、完全にはアライメントできていない。これは、初期の組合せで起こったアライメント

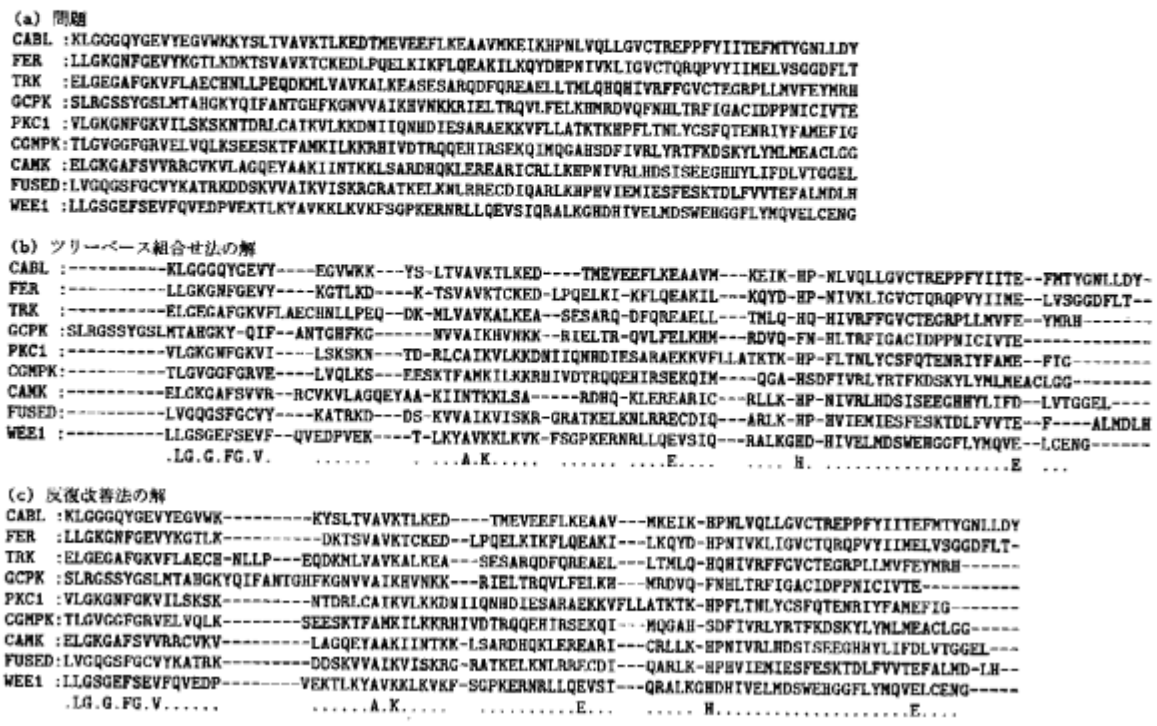


図 33: 反復改善法のアライメント例

のエラーが、後々まで波及して、最後に得られる解の精度が低下しているからである。それに対し、反復改善法の解は、ギャップが少なく、良くアライメントされている。

しかし、反復改善法は大きな問題点を持っており、このままでは実用規模のマルチプルアライメントに適用するのは難しい。大きな問題になると膨大な数の改善サイクルを要し、実行時間が反復改善法の障壁となる。n 本の配列からなる配列群の分割の仕方は $2^{n-1}-1$ 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、分割数は 50 万通り以上にのぼってしまう。とくに、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しないので、収束までの試行サイクル数も多大なものとなる。これが理由で、反復改善法はあまり注目されていなかった。

3.2 反復改善法の並列化

我々は、実用規模のマルチプルアライメント問題に対して、ツリーベース組合せ法以上の品質の解を与えることを目指し、反復改善法の並列化を試みた。そして、その実験システムを、並列計算機上に実装した。本節では、並列化の方法、その実装法、そして実験結果を、順に述べる。

3.2.1 並列化の方法

並列化には、最良優先探索とマルチ山登りの２つの方法を試みた。

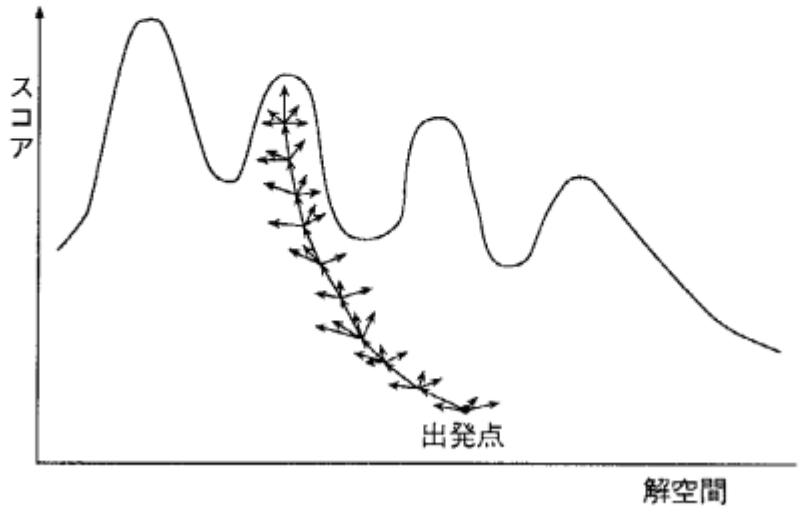


図 34: 最良優先探索の概念図

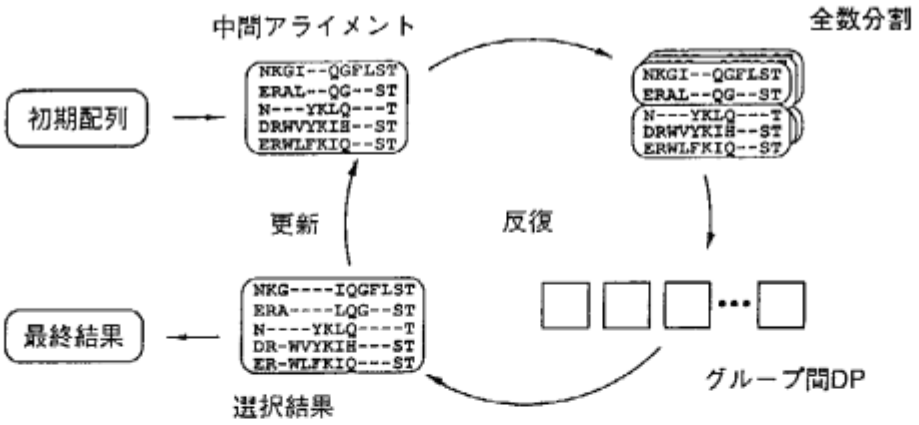


図 35: 最良優先探索の並列反復改善法

最良優先探索 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（図 34）である。最良優先探索の並列反復改善法は、近傍探索を並列化したもので、次に示す手順で改善を行う（図 35）。

1. ギャップのない配列群を初期アライメントとする。

2. 初期アライメントに対して、可能なサブアライメントへの分割 $2^{n-1}-1$ 通りを生成する。

3. サブアライメント同士の、DP によるアライメントの最適化を、各々のプロセッサで並列に行う。

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。

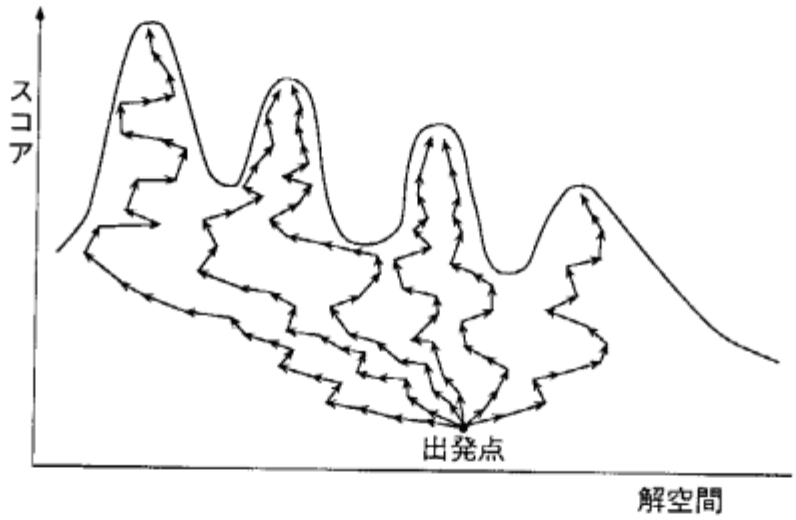


図 36: マルチ山登りの概念図

マルチ山登り　マルチ山登りの並列反復改善法では、各プロセッサで異なった乱数を用いて独立に反復改善法（図 36）を行い、その中で最もスコアの良い解を全体の解とする。前章で、シミュレーテッドアニーリングの単純並列を扱ったが、その並列化と同様な発想である。しかし、反復改善法は、解空間を「山登り」で探索しているので、その単純な並列化を、ここでは「マルチ山登り」と呼んでいる。

3.2.2　並列計算機への実装

並列反復改善法を、前章と同様に並列論理型言語 KL1[79] でプログラミングし、並列推論マシン PIM/m[80] 上に実装した。PIM/m は、前章の実装に使用した Multi-PSI の上位

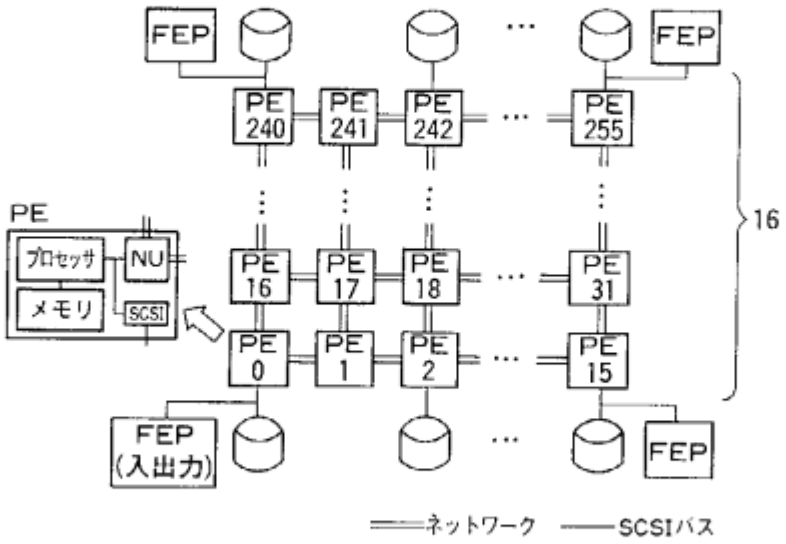


図 37: 並列推論マシン PIM/m の構成

機種であり、図37のように、256 台の要素プロセッサ（PE）が結合された分散メモリマシンである。それぞれの PE は、プロセッサの他に、80MB のメモリ、ネットワークユニット（NU）を有している。いくつかの PE は、入出力インターフェース（SCSI）を持っており、コンソール（FEP）や、ハードディスクが結合可能となっている。

各 PE で行うグループ間 DP の計算は、後藤の方法 [32] に準じて、以下の SP 体系のアライメントスコアを最適化させている。本章では、前章で扱った問題より、類似性の低い問題を扱うので、ギャップコストを上げている。

$$AlignmentScore = \sum_{i<j}^{seq.pair} \sum_{k}^{column} MatchScore(A_{ik}, A_{jk})$$

$$MatchScore(A_{ik}, A_{jk}) = \begin{cases} Dayhoff(A_{ik}, A_{jk}) & A \text{ がともにアミノ酸} \\ 0 & A \text{ がともにギャップ} \\ 0 & A \text{ の一方がアウトギャップ} \\ -8 & A \text{ がアミノ酸と opening gap} \\ -1 & A \text{ がアミノ酸と extending gap} \end{cases}$$

最良優先探索　並列計算機 PIM/m 上の最良優先探索の実装は、次のようにした。たとえば、図33のような9本のアライメント問題は、分割数が全部で255 通りあるので、マスタープロセッサ（PE0）が 255 台のスレーブプロセッサに、各分割問題を分配して、それぞれ並列

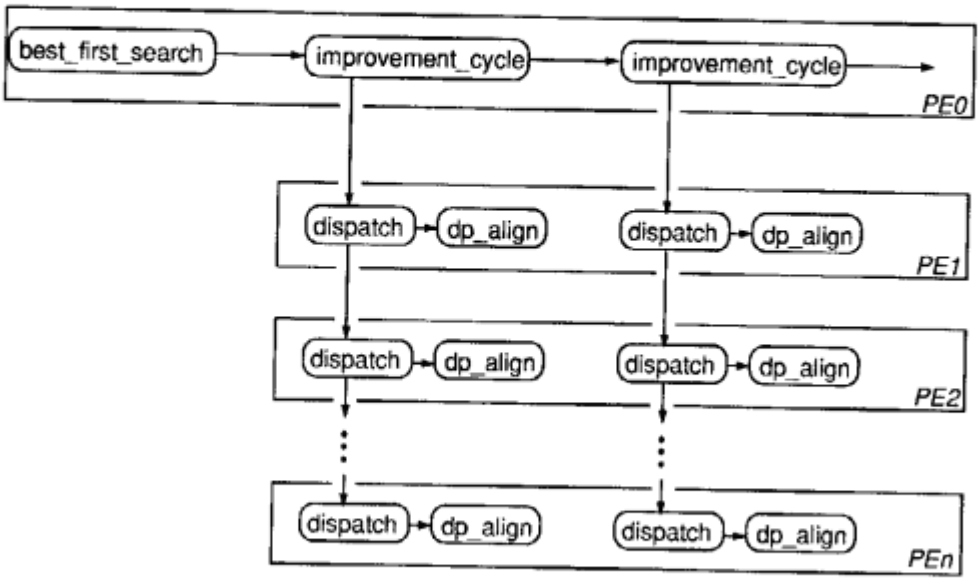


図 38: KL1 による最良優先探索並列の実装概念

に DP を適用して最適化させる。その後、マスタープロセッサは順に解を回収し、全てが集まったならば最良解を判定して、次の改善サイクルを開始する。KL1 によるこの実装は、図 38のようになされる。図中の矢印はプロセス生成の依存関係を表し、処理すべきデータは、この場合矢印方向にプロセスに与えられ、処理結果は、反対方向に回収される。

KL1 プログラムの概要は、図 39の通りである。可能な partition の数だけ、dispatch プロセスが生成され、並列実行される。それらは、@node に PE 番号が設定されることで、実行プロセッサが指定されている。解の回収は Result_pair のリストで行われるが、最良解以外は使用されないので、マスタープロセッサには、ポインターのみが渡されるだけで、最良解以外のアライメントの実データは転送されない。KL1 の処理では、常に必要なとき必要なだけのデータ転送が、PE 間でなされる。

KL1 のような論理型言語の実行では、メモリが一杯になると、データのゴミ集め処理（GC; gabage collection）が起動されることで、空きメモリが確保される。この処理は、どこからも参照されていない不要データを探す、時間のかかる作業である。並列実行を長時間行うとメモリの消費量も多くなり、必然的にゴミ集め処理の時間が大きな割合を占めてくる。実行時間の節約には、なるべくメモリを消費しないプログラムを組むこと、とくに、PE をまたがったデータ参照を極力しないことである。図 39で、中央部に copy という処理があるが、これは、たびたび変更を加えるアライメントのデータを完全にコピーし、PE をまたがったデータ参照を抑えるものである。コピーにより不用意にメモリを消費しているように見える

が、こうしておくと、不要になったあとは、PE ごとにゴミ集め可能なデータとなり、空きメモリ確保が速い。

```
best_first_search(Align,Parameters,Cycle_out,Align_out,Score_out) :-
        evaluate_score(Align,Parameters,Score),
        generate_partitions(Align,Partition_list),
        improvement_cycle(1,Align,Score,Partition_list,Parameters,Cycle_out,Align_out,
                                                                    Score_out).

improvement_cycle(Cycle,Align,Score,Partition_list,Parameters,Cycle_out,Align_out,
                                                                Score_out) :-
        dispatch(1,Align,Partition_list,Parameters,Result_pair_list)@node(1),
        search_maximum_score(Result_pair_list,Max_pair),
        Max_pair = (MaxAlign,MaxScore),
    ( MaxScore > Score  -> (
        NextCycle := Cycle + 1,
        improvement_cycle(NextCycle,MaxAlign,MaxScore,Partition_list,Parameters,Cycle_out,
                                                        Align_out,Score_out));
      MaxScore =< Score -> (
        Cycle_out = Cycle,
        Align_out = Align,
        Score_out = Score )).

dispatch(PE,Align,[Partition|Partitions],Parameters,Result_pair_list) :-
    Partitions \= [] |
        copy(Align,Align_copy1,Align_copy2)
        NextPE := PE + 1,
        dispatch(NextPE,Align_copy2,Partitions,Parameters,Result_pairs)@node(NextPE),
        dp_align(Partition,Align_copy1,Parameters,Result_pair),
        Result_pair_list = [Result_pair|Result_pairs].
dispatch(PE,Align,[Partition|Partitions],Parameters,Result_pair_list) :-
    Partitions = [] |
        dp_align(Partition,Align,Parameters,Result_pair),
        Result_pair_list = [Result_pair].

dp_align(Partition,Align,Parameters,Result_pair) :-
        partitioning(Partition,Align,SubAlign1,SubAlign2),
        dp:main(SubAlign1,SubAlign2,Parameters,NewAlign,Score),
        Result_pair = (NewAlign,Score).
```

図 39: 最良優先探索並列の KL1 プログラム

現在程度の PE 数では問題ないが、PE が数千個にもなると、解を回収する部分で、マスタープロセッサが通信ネックになる可能性が出てくる。そのような場合は、マスタープロセッサを複数にし、それらを統括するマスタープロセッサのマスターを設け、通信の問題を回避する必要がある。

マルチ山登り マルチ山登り並列反復改善法は、反復改善法の逐次処理を、複数のPEで個々独立に行ったものから、各時点での最も良い解を選び、全体としての解とするものである。使用プロセッサ数は任意であるが、最良優先探索並列と比較するときは、配列9本のアライメントに対してスレーブプロセッサ数を255台としている。マスタープロセッサは必ずしも必要ないが、我々の実装では、定期的（80秒ごと）に解を回収しその時点の最良解をモニタするために、マスタープロセッサを1台（PE0）割り当てている。通信量は極めて少なく、通信のオーバーヘッドの問題はない。

3.2.3 並列効果の比較実験

図33のアライメント問題を、逐次反復改善法と、2つの並列反復改善法とで解いた場合の性能比較を、図40に示した。また、従来のツリーベース組合せ法で得られるスコアのレベルも合わせて示した。図40の逐次法の曲線は、マルチ山登り並列の平均をプロットしたもので、逐次法を255回、異なった乱数で行ったものの平均に相当している。図を見ると、いずれの反復改善法においても、ツリーベース組合せ法で得られるレベルを越えてアライメントが改善されること、逐次反復改善法が並列化手法で高速化されていることがわかる。

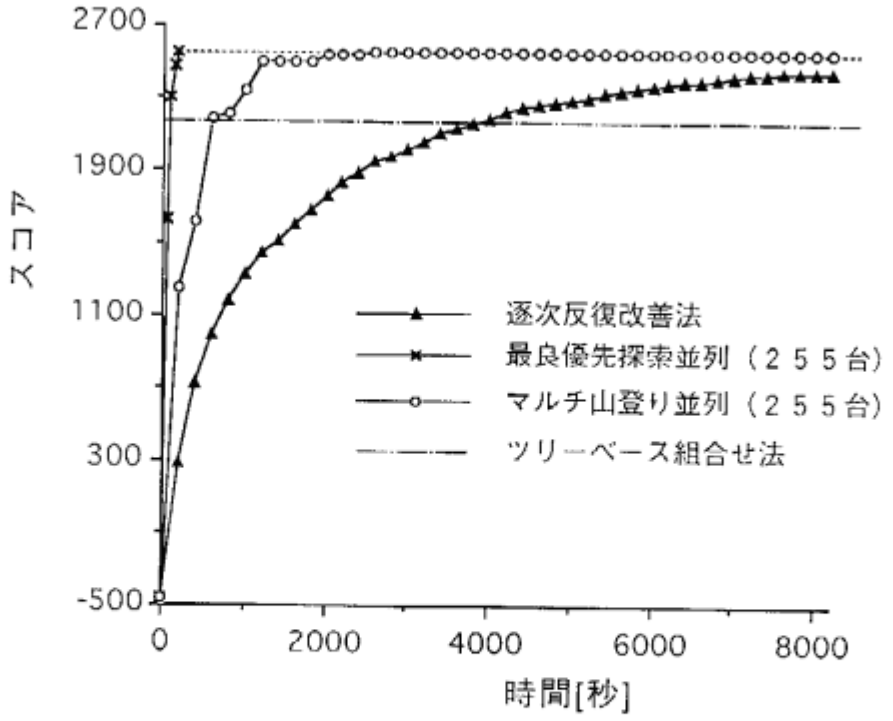


図 40: 逐次／並列反復改善法の改善履歴比較

逐次反復改善法では、実行時間8000秒（反復改善サイクルにして約630回に相当）の解の平均スコアは2447であったが、その値に最良優先探索並列では124秒（総反復改善サイクルにして2550回）で、マルチ山登り並列では1063秒（総反復改善サイクルにして約2.1万回）で至っている。実行速度として比較すれば、それぞれ、64.5倍、7.5倍となったことに相当するが、並列化手法では、総反復改善サイクル数が大幅に増えている。これは、膨大な無駄計算に支えられた結果として、計算時間が低減したことを意味している。

最良優先探索並列を、マルチ山登り並列と比較すると、前者はサイクルごとにPEの待ち時間が発生するものの、無駄計算が少なく、早めに良い解が得られる。図40の実験では、最良優先探索並列は、182秒で2544のスコアの最終解を生成したが、マルチ山登り並列の解は、2544に達するのに2702秒を要した。しかし、さらに時間をかけると、その解は2544を越えて良くなっていった。マルチ山登り並列は、時間をかければ最良優先探索の解のスコアを上回るようである。最良優先探索の解は局所最適解に陥っていると予想されるので、マルチ山登りの解の方が最終的には良くなる可能性が高い。

それを確めるために、次の実験を行った。配列9本のアライメント問題の場合、分割数はわずか255通りなので、全通りの分割についていずれも改善がないことを確認したうえで、収束を判断できる。そのような収束条件を各プロセッサに設定したうえで、図33と同じ規模の30種のアライメント問題について、255台並列のマルチ山登りを適用してみた。その結果、マルチ山登り並列は、全ての問題において最良優先探索の解のスコアを上回った。また、収束時に各プロセッサが持っていた解のスコアをそれぞれ調べたところ、そのうちの53.4％は、最良優先探索の解のスコアに満たなかった。最良優先探索並列は、逐次反復改善法の平均的な解に、高速に至っていると判断できる。

3.3 限定分割法の導入

配列20本以上の実用規模のアライメント問題は、並列化だけでは対処できない。なぜなら、反復改善法の探索空間は、配列の本数の増加に対して、指数的に広がるからである。そこで、探索空間を効率良く制限するヒューリスティクス（経験的手法）である限定分割法を３種類開発し、並列反復改善法に導入した。

3.3.1 限定分割法とその妥当性

我々は反復改善法の実験を繰り返すうちに、配列数が不均等に分割されたサイクルのスコア改善が比較的大きいことに気づいた。つまり、１本と残り（１本抜き）とか、２本と残り

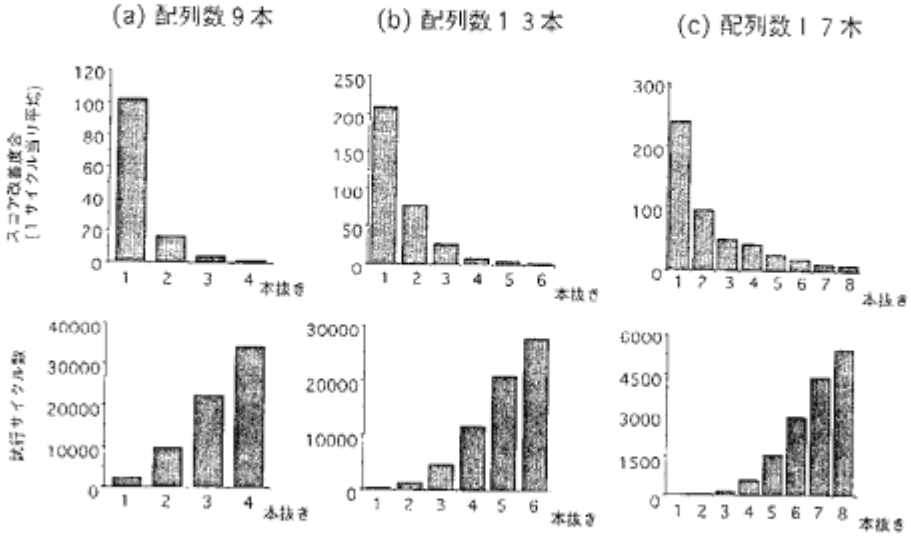


図 41: 改善度合の各分割による比較

（２本抜き）のように配列が分割されたサイクルは、配列の本数が均等に分割されたサイクルよりも、スコア改善度合が平均して大きいのである。さらに、各分割を等確率に選ぶ、反復改善法の本来の選択方法では、数が少ない不均等の分割が選ばれにくく、スコア改善が遅いことがわかった。

図41は、その傾向を示すグラフである。上段にはサイクル当りのスコア改善度合が、下段には試行されたサイクル数が、それぞれ、分割が何本抜きであったかによって区別されて示されている。図を見ると、１本抜きや２本抜きの改善度合が大きいのに、それらが試される回数は相対的に小さいことがわかる。試行回数のグラフは二項分布になっているので、配列本数が増えると、ますますその傾向が大きくなってくる。

１本抜き／1,2 本抜き限定分割　こうした特徴を捉え、１本抜き限定分割法と 1,2 本抜き限定分割法を開発した。１本抜き限定分割法では、配列群を２つに分割するときに、小さい配列群の配列数を必ず１本とする。配列 n 本のアライメント問題の場合、分割数は n 通りである。1,2 本抜き限定分割法では、小さい配列群の配列数を必ず１本か２本とし、配列 n 本に対する分割数は、$n(n+1)/2$ 通りになる。従来の反復改善法の分割数が、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。それでも、解のスコアはそれほど低下しない。もちろん、1,2 本抜き限定分割法は、１本抜き限定分割法よりも、平均して少し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、次の理由からだろう。アライメント途中の配列群は、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て、配列数が少ないとカラムの特徴が比較的よく表れる。カラムの特徴がはっきりしていると、配列群間のアライメントの最適化が効率良くなされ、スコアの向上が大きい。

ツリー依存限定分割 次に、なんとか1乗オーダーの分割数で、1,2本抜き限定分割法以上の効果をあげられないかと考え、第3の限定分割法を模索した。1,2本抜き限定の、1本抜き限定に対する優位性は、アライメントの過程で良く揃った2本が、一緒に最適化されることに由来すると思われた。それならば、反復改善の過程で良く揃った配列群を、改善サイクルの都度に抽出し、それらと残りの配列間で最適化をすれば効果的と推測された。こうした発想から、ツリー依存の限定分割法が開発できた。

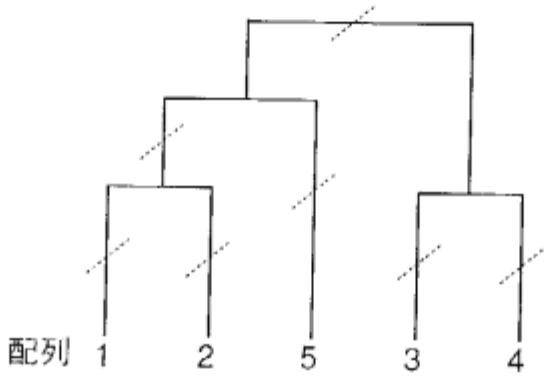


図 42: ツリー依存限定分割

ツリー依存限定分割法では、各改善サイクルの始めにその時点でのアライメントの状態を調べ、全配列対のアライメントスコアを三角表にする。それに基づいてUPGMA法でツリーを描画（図16参照）し、ツリー上で近い配列は、なるべく同じ配列群に含まれるように、分割を決めるのである。具体的には、図16で得られたツリーに対して、図42のように、ツリーの任意の1か所を切断して分けられる2つの配列群を、最適化の対象とする分割に選ぶ。この分割数はツリーの形状によらず、配列n本に対して$2n-3$通り[81]である。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。

3.3.2 限定分割導入時の並列効果比較

図33と同様規模の配列9本のアライメント30問について実行し、限定分割法による並列効果を検討した。最良優先探索の並列反復改善法に、各種限定分割法を導入した場合の性能

	全数分割	限定分割		
		1本抜き	1,2本抜き	ツリー依存
実行プロセッサ	255台	9台	45台	15台
解の平均スコア	2129	2064	2091	2116
平均実行時間	246秒	177秒	197秒	182秒
平均改善サイクル数	16.9回	15.6回	16.1回	14.7回
サイクル当り実行時間	14.6秒	11.3秒	12.2秒	12.4秒
プロセッサの稼働率	67%	78%	74%	68%

図 43: 最良優先探索並列反復改善法の性能比較

比較を、図43に示す。この実験結果から次のことがわかる。

- 限定分割法の導入により、実行プロセッサ数（マスタープロセッサを数に含めていない）が大幅に減少しているにもかかわらず、解のスコアと実行時間は、全数分割（限定分割を導入してない並列実行）と大きな差はない。
- 全数分割は、限定分割より解の平均スコアが少し良いが、実行時間が多くかかっているのを考慮すれば、大きな優位性を見い出せない（図44も参照されたい）。
- 全数分割は、稼働率が低く、サイクル当り実行時間も大きくなる傾向がある。その理由は、DP を行うときに、比べられる配列群が４本と５本のように均等に近くなっていると、１本と８本などに比べ最適化計算に時間がかかる。全数分割では、均等に近い分割を処理しているプロセッサの数が多く、その計算をしているプロセッサのうちのひとつが、サイクルの時間的ネックとなりやすい。
- 限定分割同士を比較すると、ツリー依存分割のスコアが最も良い。ツリー依存分割は、全数分割と同じ理由で稼働率がやや低いが、平均改善サイクル数がやや少ない。またツリー依存分割は、1,2 本抜きよりプロセッサ数が少ないのにもかかわらず、良い解を早めに生成できる傾向がある、

図 44には、マルチ山登り並列の反復改善法に各種限定分割法を導入した場合の、平均スコアの向上履歴が、グラフで表示されている。比較のため、図 43に示した最良優先探索並列の４手法による結果と、従来のツリーベース組合せ法による結果も合わせて示してある。この図から次のことがわかる。

- マルチ山登り並列の全数分割は、この実験時間内では、収束には遠く及ばない。それに対し、限定分割を導入した各手法は、いずれも収束性が早まり、解が高速に得られている。

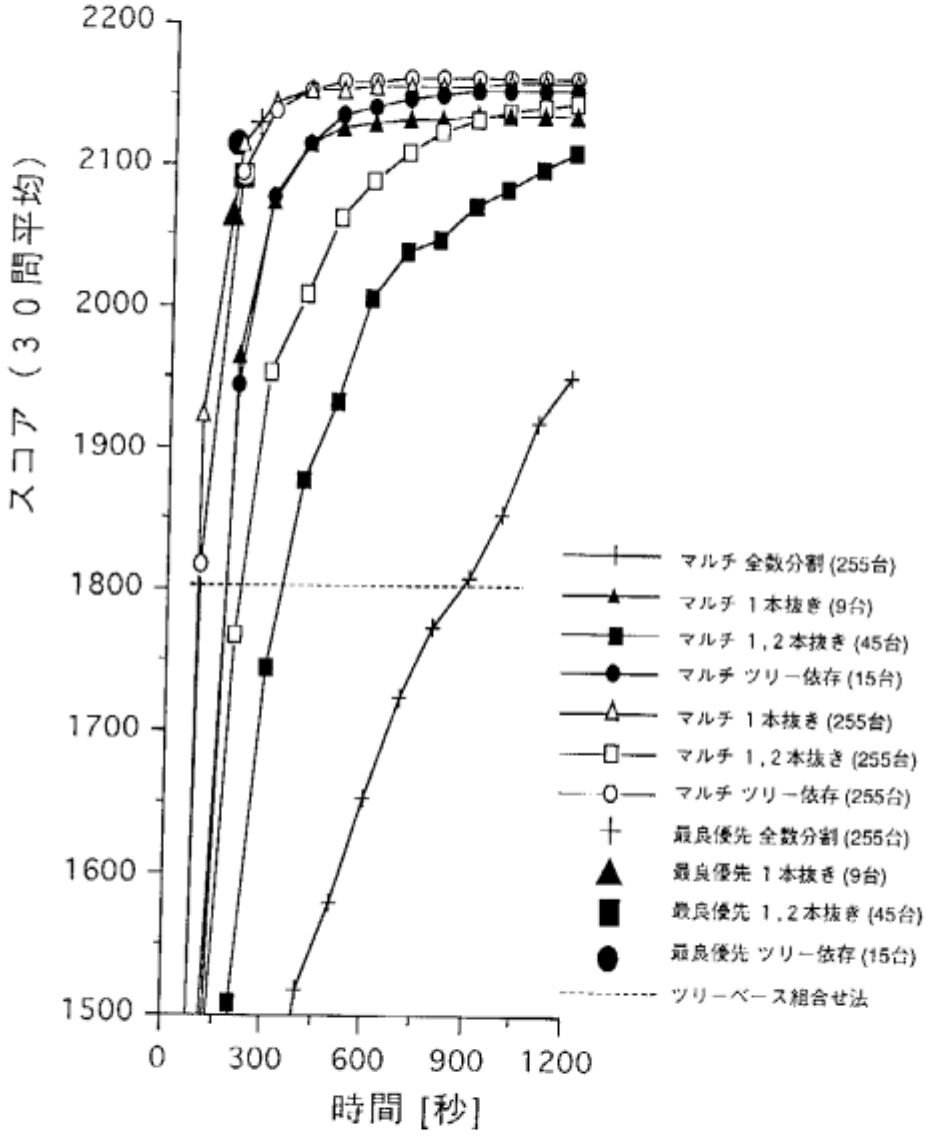


図 44: 限定分割による改善履歴比較（配列 9 本）

- マルチ山登り並列の限定分割を導入した各手法と、最良優先探索並列は、点線で示したツリーベース組合せ法の解（平均生成時間 69 秒）のレベルを、ツリーベース組合せ法の数倍の時間内に上回っている。

- マルチ山登り限定分割法を、分割数と同じ台数（9/45/15）で並列にした試行の性能を、最良優先探索並列の各分割法の性能と比較すると、いずれもマルチ山登り並列が、時間はかかるものの、最良優先探索並列の解のスコアを上回っている。（マルチ山登りの1,2本抜き限定分割は、無駄な分割の試行が多くなり、45台程度では、このグラフの時間内では収束傾向に至っていない。それでも、最良優先探索並列の1,2本抜き限定分割の解を平均して上回っている。）

- マルチ山登り限定分割法を、255台並列に拡張したものは、収束性が高まっている。とくに、ツリー依存限定分割、1本抜き限定分割の実行速度は、最良優先探索並列の各解の生成時間と比べても遜色ない。最終収束解のスコアは、同じ255台並列でもツリー依存限定分割が最も良い。

3.3.3 実用規模の問題解決

図33と同様の配列を22本に増やした実用規模のアライメント問題を30問作成し、限定分割法を導入した並列実行を行った（全数分割は分割数が200万を越えるため行えない）。図45では、それらの平均スコアを図44と同様なかたちで比較している。その結果、次の点で図44と同様な傾向が見られた。

- 限定分割を導入した並列反復改善法の各手法が、点線で示したツリーベース組合せ法の解（平均生成時間388秒）のレベルを、早期に上回っている。

- 同じPE台数のマルチ山登り並列と最良優先探索並列を比べると、最良優先探索の方が、ある程度良い値を早期に与える。

- 時間をかければ、一般に、マルチ山登り並列が最良優先探索並列の解のスコアを上回る。

- 同じPE台数のマルチ山登り並列同士を比べると、ツリー依存限定分割が平均して最も良い解を与える。

問題の規模が上がったことにより、図44と異なる傾向も見られた。

- マルチ山登り並列の1,2本抜きは、試行する分割数が増大したため、PEを253台用いても、収束性が十分でない。

- 探索空間が広がったために、比較的良い解が早く得られる最良優先探索の特徴が際だった。

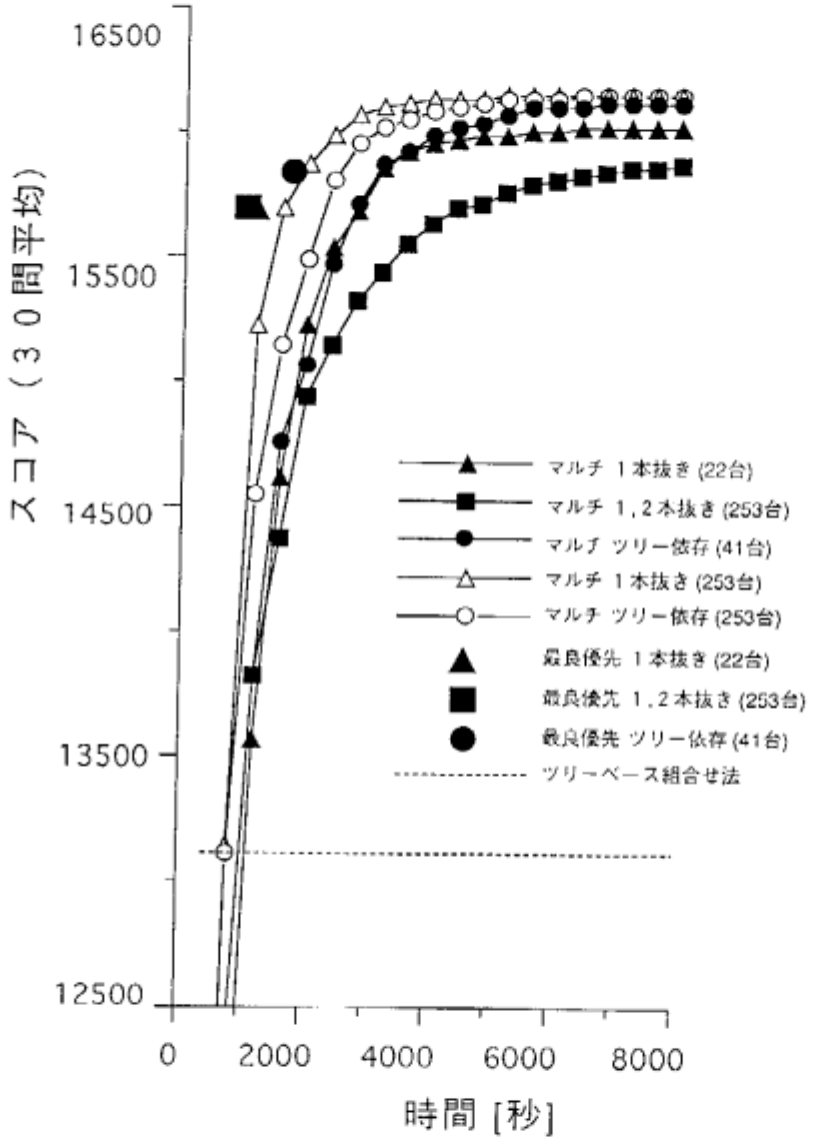


図 45: 限定分割による改善履歴比較（配列 22 本）

- 最良優先探索のうち、ツリー依存分割は、処理する配列の本数が増えて要素プロセッサの稼働率が低下（50 %になっていた）し、実行時間が相対的に上がってしまった。

この実験結果からすると、この程度の実用規模の問題を解く場合は、次のように解決手法を選ぶとよい。解決にそれほど時間をかけたくないときは、最良優先探索並列の 1,2 本抜き

／1本抜き限定分割が良い。解決にある程度時間をかけられるときは、マルチ山登り並列のツリー依存／1本抜き限定分割が良い。

今回使用した問題のうちの1問を、最良優先探索並列の1,2本抜き限定分割で実行した結果を、我々が開発した表示ツールに出力したところを、図46に示した。各文字をアミノ酸の性質によっておおよそ色分けをしており、アライメントされた各カラムの特徴が一目でわかる。また、中段の棒グラフは、各カラムの類似性の評価スコアを示している。明るい棒は比較的スコアの高いカラムを示している。なお、この画面では、アライメント結果の右側が少し切れているが、それは画面をスクロールして見るように設計されている。

図46の問題解決を実行している途中の、各プロセッサの稼働状況が、図47に示されている。縦軸は256台のプロセッサ（一番上がマスタープロセッサ）、横軸は時間で、2秒ごとの稼働率が色で示されている。時間方向に見られる周期的なパターンが、反復改善法の1サイクルに相当する。最良優先探索並列では、改善サイクルごとにスレーブプロセッサの同期をとっているため、各サイクルの終りに、待ち時間（青色の部分）が発生する。実行時間全体でスレーブプロセッサの稼働率は67％であった。周期が徐々に長くなっているのは、配列にギャップが挿入されてグループ間DPの所要時間が長くなるからである。この現象が起きるのは、実行の始めの段階で、実行の後半では、ギャップの数が安定して、周期がおおよそ一定になる。

マルチ山登り並列を実行中のスレーブプロセッサは、ほとんどつねにフル稼働（全面赤）である。各スレーブプロセッサは改善サイクルが終り次第、システムタイマーを調べ、80秒を越えるごとに、現在のアライメントの状態をマスタープロセッサへ送っている。スレーブプロセッサによっては、そのモニタ解の通信が競合して待たされることもあるが、それでも、待ち時間はごくわずかで、全体のスレーブプロセッサの稼働率は99％以上であった。

3.4 考察

我々は、これまで実行時間がかかり過ぎ、実用的でなかった反復改善法を、並列化の手法を適用することで、実行性能の実用レベルまでの向上を図った。

はじめに、配列9本程度の小規模な問題で、逐次反復改善法と2種の並列反復改善法とを比較した結果、255台規模の並列実行で、最良優先探索並列で数十倍程度、マルチ山登り並列で数倍程度の高速化が得られた。そのうえ、得られる解の平均スコアも向上した。また、マルチ山登り並列は、十分な実行時間をかけると、最良優先探索並列の解のスコアを平均して上回ることがわかった。

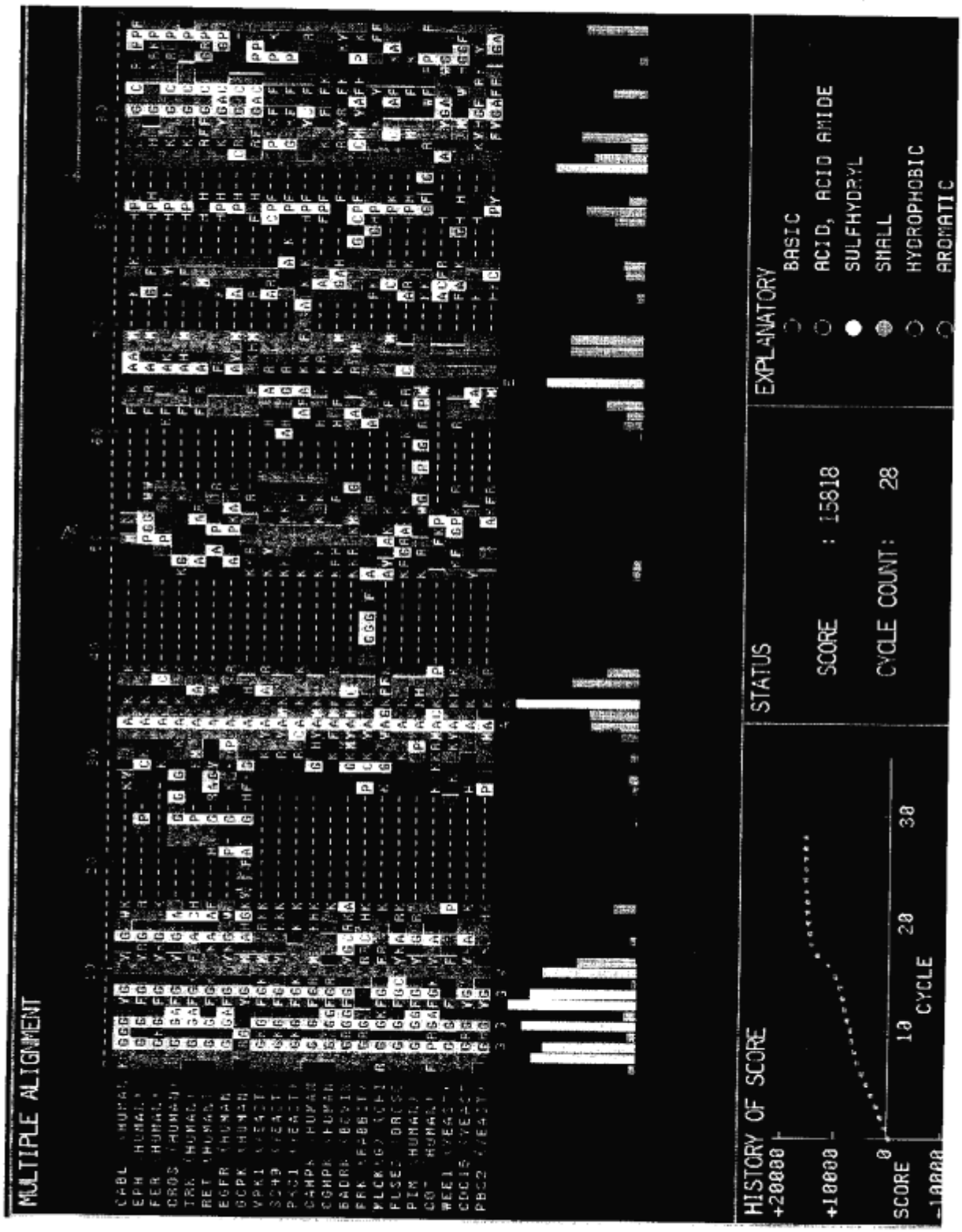


図 46: 並列反復改善法による実用規模のアライメント例

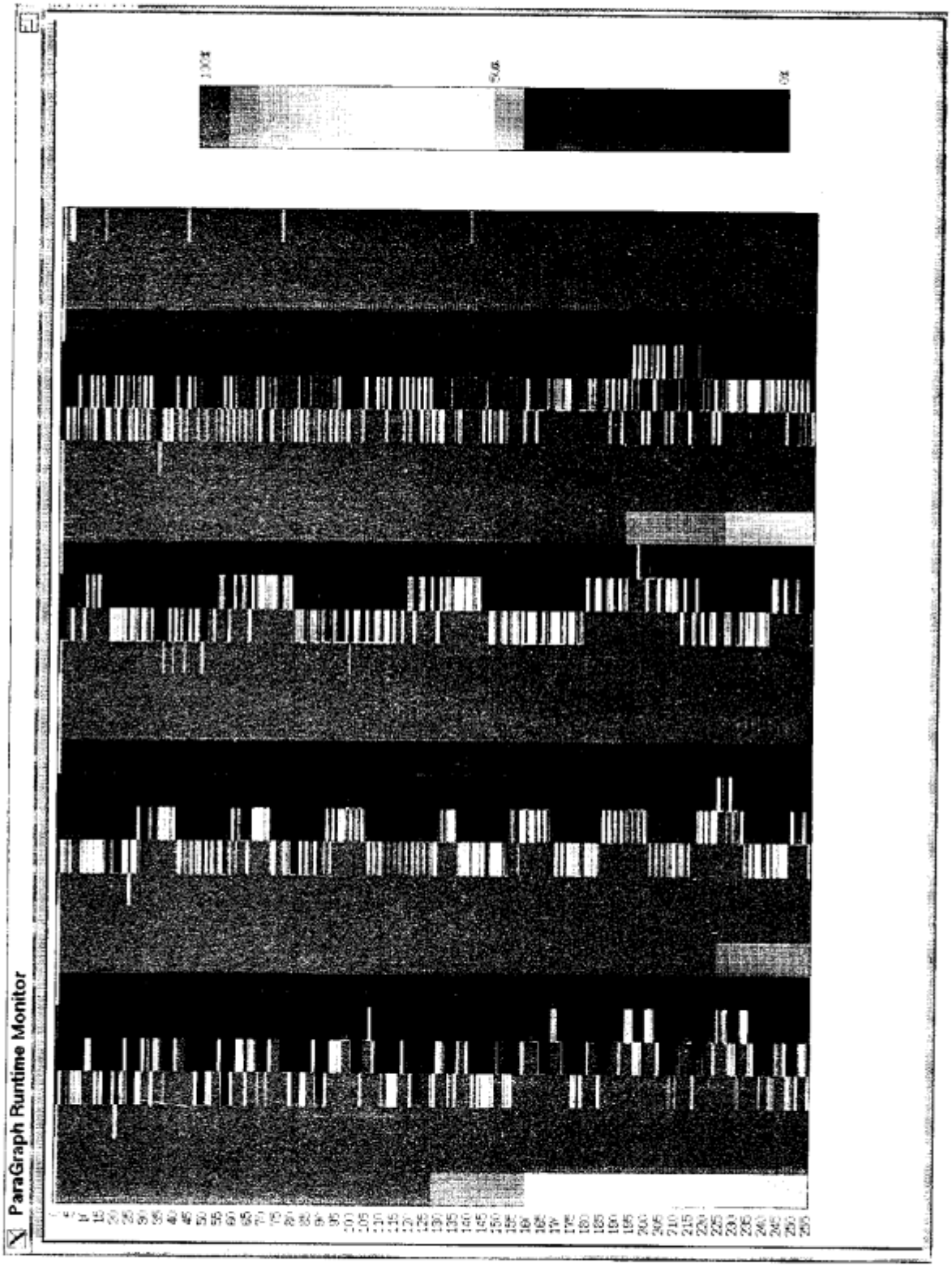


図 47: 要素プロセッサ稼働状況（最良優先探索並列）

さらに、独自の限定分割法を導入することにより、解の品質の大きな低下を招くことなく、使用プロセッサ数、収束までにかかる時間を大きく削減できた。その結果、配列22本程度の実用規模の問題も、反復改善法で実用的な時間内に解決可能となり、従来用いられていたツリーベース組合せ法に比べ、安定して高品質の解を生成できるようになった。

本節では、前節の実験結果を踏まえ、さらに大規模な問題を解くことを想定し、限定分割法の比較、最良優先探索並列とマルチ山登り並列との比較を行う。最後に、今後の課題を述べる。

3.4.1 限定分割法の比較

今回の実験によると、1,2本抜き限定分割の1本抜き限定分割に対する優位性は、十分には出てはいなかった。マルチ山登りでは、1,2本抜き限定分割の収束性が低くく、最終解に至るまで実験を続けられていなかった。最良優先探索では、1,2本抜き限定分割は1本抜き限定分割よりも良い解を与えていたが、その差はわずかであった。ここで、配列が50～100本ある大きな問題を解くことを考えてみると、1,2本抜き限定分割では、1000台から5000台のPEが（あるいはそれに見合う時間が）必要となり、ますます優位性がなくなってくる。

また、実験では、ツリー依存限定分割は、1本抜き限定分割より若干良いスコアの解を与えていた。この優位性は、配列が50～100本の問題になると、若干大きくなると予想される。なぜなら、配列の本数が増えると、その中に比較的類似した2～3本の配列群が存在しやすい。ツリー依存限定分割は、配列の類似性のアンバランスに対応できるが、1本抜き限定分割は、類似した配列群だけを一緒に動かす最適化ができないので、アンバランスに対応しにくい。

しかし、ツリー依存限定分割は、1本抜き限定分割より2倍のPE数が必要な点も無視できない。類似した配列群は1本の配列とみなして、1本抜き限定分割を行うのも良いアイデアである。第5章では、ユーザーからのインタラクティブな指定で、配列をグループ化して1本抜き限定分割を行うシステムを紹介する。

3.4.2 並列化手法の比較

今回の実験によると、マルチ山登り並列が、実行時間はかかるものの、最終解のスコアでは、最良優先探索並列を上回っていた。しかし、配列が50～100本の問題になると探索空間が広まり、マルチ山登り並列の収束性も落ちてくる。そうなると、マルチ山登り並列は実行に膨大な時間を要し、とても使えるものではなくなってしまう。すなわち、大規模な問題

では、収束性の良い最良優先探索並列が、相対的に有用となる。だが、最良優先探索は、比較的スコアの悪い局所最適解に陥りやすい点も考慮に入れておくべきである。

最良優先探索とマルチ山登りとの折衷案を考えることで、大規模な問題に効果的に対処できる可能性がある。ひとつの案は、つねに最良から数個の解候補を保存しながら探索する、いわゆるビームサーチを行うことである。すると、最良優先探索にマルチ山登り的要素を入れたこととなり、高い収束性を保ったまま、局所最適解の回避をある程度行え、解のスコアを向上させることができる。もうひとつの案は、逆に、マルチ山登りに最良優先探索的要素を入れたもので、山登りをする途中のたくさんの解を比較し、相対的に悪い解を捨て、良い解の近傍を複数で探索する手法である。これは、いわゆる遺伝的アルゴリズムの枠組に含めることができる（次章）。

3.4.3 今後の課題

本システムの効率化について、その他の今後の課題を述べておこう。まず、初期状態のとり方に改善の余地がある。今回は、ギャップなしのアライメント状態から始めたが、マルチ山登りは、解空間のいろいろな点から探索を始めたほうが、一般に効果的である。前章で議論した、シミュレーテッドアニーリングの高温処理のようなモジュールを設けて、いろいろな状態を生成し、それらを初期状態とするマルチ山登りを行うと良いだろう。最良優先探索も、違った初期状態から何回かやってみると、最良解のスコアが向上するであろう。

反復改善法とツリーベース組合せ法の融合も興味深い。ツリーに従って配列を組み合わせたたびに、その状態を反復改善法で修正する方法が提案されていた[82]。しかし、この方法も分割数が膨大で、反復改善に大きな時間がかかるため、実用規模の問題に適応できなかった。ところが、我々の限定分割法を導入すれば実用規模の問題に対処可能となるうえ、並列化手法による実行時間の低減も大きく、一転して有望な手法となっている。我々は、こうした一連のアルゴリズムの、逐次計算機上での性能比較も行った[83]。

最後に評価スコアについて考察する。山登りの収束解が非常に多様であることからわかるが、解空間はかなりマルチピークである。今回は、評価スコアはSP体系を用いることが前提であったが、今後は、スコアに局所的な平滑処理を加えることも検討が必要であろう。そうすると、解空間がなだらかになってピークが減り、大局的な最適解へ至り易くなると考えられる。ただ、それを行う際には、スコア計算にかかる演算の増大と、変更されたスコアの生物学的な妥当性の検討も、合わせて行う必要がある。

4 並列遺伝的アルゴリズムによるアライメント

本章では、マルチプルアライメントの問題を組合せ最適化問題と捉え、遺伝的アルゴリズム (genetic algorithm) の手法を応用した並列システム [84] について、報告する。本システムは、遺伝的アルゴリズムをマルチ個体群の方式で並列計算機上に実装したものであり、実用規模のアライメント問題でも、準最適な解を比較的高速に求めることができる。本システムは、前章で述べたマルチ山登り並列の反復改善法より、早期に収束に至り、かつ、最良優先探索並列のような局所最適解に陥ることが少ない。

本章の構成は次の通り。まず、遺伝的アルゴリズムの手法を紹介したのち、我々が開発した、マルチプルアライメントへの適用法について解説する。次に、遺伝的アルゴリズムの並列実装法を説明し、続いて、その実験結果と考察を示す。

4.1 遺伝的アルゴリズム

遺伝的アルゴリズム（以下 GA と略す）は、遺伝子の生物学的進化の過程 [6] にヒントを得た、計算機上の組合せ最適化問題の解法である [85, 86]。GA の解空間探索は、複数の探索点が互いに協調あるいは競合をすることで、高い収束性と局所解の回避を同時に実現する特徴を有している。最近になって、生物学の問題を組合せ最適化問題と捉え、GA を応用する研究が出てきている [87, 88, 89, 90]。生物界の現象にヒントを得た計算機手法が、生物学分野の問題に応用されているのは、面白い現象である。

本節では、GA について、その基本方式と、マルチプルアライメント問題への適用に際して、我々のアイデアの基盤となったマルチ個体群方式とに、分けて説明する。

4.1.1 遺伝的アルゴリズムの基本方式

GA は、生物個体群が、世代交代を繰り返して、環境へ適応していく過程を模擬している。生物個体群は各世代で、遺伝情報に発生する突然変異と、それに伴う自然淘汰、繁殖、そして、雌雄の遺伝情報の交配によって、環境により良く適応する個体群が形成されていく。この過程を解空間内の探索にあてはめたのが、GA である（図 48）。具体的には、各個体が解空間のあるひとつの状態を保有し、その各々に状態を変更する操作（変異）を施した後、スコア（適応率）の悪い状態のいくつかを捨てる（淘汰）。その失われた状態数に見合う数だけ、生き残った個体がコピーされる（繁殖）。そして、さらに適当に 2 個体を選び、部分解同士を組合せた状態に交換する（交配）。この一連の操作を 1 世代として、世代交代を繰

り返すと、各個体が持つ解がスコアの良い状態に一様化して行き、解空間の（準）最適解が得られる。通常は、個体群内の最大スコアの解を、最終解とする。

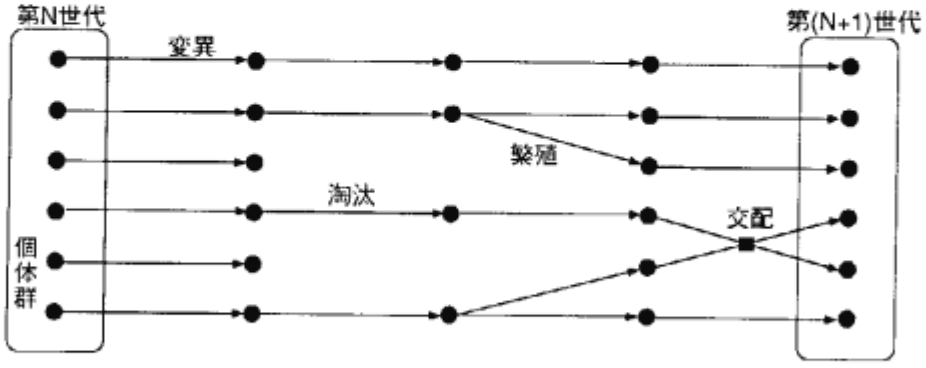


図 48: 遺伝的アルゴリズム（GA）の基本戦略

初期の頃のGAは、遺伝情報を担うDNAとのアナロジーで、解の状態表現を1次元のビットストリング（1/0の文字列）としていた。しかし、それでは大きな問題を解くには著しく効率が悪いことが多く、最近では、問題に応じた適当な状態表現を、GAの開発者が自由に設計する傾向にある。変異、淘汰、繁殖、交配、そして個体数は、解空間探索において、以下のような役割をする。

変異： 基本的にはシミュレーデッドアニーリングにおける微小変形に相当するものであり、解空間内の探索点の近傍にランダムに移動する操作である。変形の度合を大きくすると解空間の広い探索、小さくすると狭い探索になる。伝統的なGAとは異なるが、変形のタイプをスコアを向上させるもの（もしくは向上させる可能性の高いもの）に限ることもできる。すると、探索は山登り的になり、収束は早まるが、局所解に捕われ易くなる。

淘汰： 悪い解の破棄であるから、この度合が大きいと解が一様化（各個体が同じ解を持つ）傾向になり、局所解に捕われ易くなる。逆に小さいと、解の多様性を保持し、収束は遅くなる。

繁殖： 良い解のコピーであるから、良い解の周辺の空間を密に探索することに該当する。繁殖の親を最も高いスコアの解に限定する（エリート戦略）と、収束は早まるが、解が一様化傾向になり、局所解に捕われ易くなる。

交配： 2つの解の部分解を組合せ、新たな解を作り交換するのだから、解空間の遠い位置へのジャンプに相当する。個体群内の解が一様化してくると、意味をなさなくなる。解表現のモジュール性が高い（良い部分解と良い部分解とを組合せると良い解ができる）場合は、交配が探索の効率に大きく寄与する。

個体数： 淘汰数と繁殖数を違えると、個体数が変化する。個体数が多いほど解空間内の広い探索となり有利である。しかし、普通は計算資源が限られているので、個体数を増やすと実行時間が増大する。一般には、計算資源と問題規模に見合った、適当な一定の個体数とする。個体数を制御して、探索の範囲を動的に決めることも可能である。

こうした操作をどのように設計するかで、かなり特徴の違った探索を実現できるので、GA は非常に柔軟な枠組と言える。逆に、GA を効率良く動作させるのもさせないのも、これらの設計の如何にかかっているとも言える。

4.1.2 遺伝的アルゴリズムのマルチ個体群方式

GA では、変異や交配などの設定の仕方で、探索の特徴が大きく変わる。一般には、ある方針に決めたならば、それに従って最初から最後まで GA を実行する。しかし、問題によっては、実行の経過に従って設定を変更した方が良い場合がある。たとえば、実行の初期段階では広い解空間を探索し、終了段階では狭い空間を集中的に探索する戦略が、多くの問題で有効なことは容易に予想がつく。GA には、こうした実行の経過に沿った設定の変更が導入できる。まさに、シミュレーテッドアニーリングにおける、温度スケジュールに相当する。

GA では、実行の経過に沿った設定の変更を、マルチ個体群方式を用いて合理的に行える[91, 92]。設定の異なるいくつかの個体群を準備しておき、各々の個体群の個体数を増減させることで、実行の経過に沿った設定の動的制御が実現できる。たとえば、Muehlenbein らは、突然変異率の大きく異なる個体群をいくつか設け、それらの間で移民を行う方法を提案している。

図 49に示す彼らの方法では、数世代ごとに、成績（群中の最大スコアの解で決まる）の良い個体群が、他の個体群から移民を受け付ける。ほかに少しの無作為な移民を認めることで、実行の初期には、探索の広い（突然変異率の高い）個体群が成績が良く個体数を増やし、実行の終盤では、探索の狭い（突然変異率の低い）個体群が優勢となる。全体として、結果的に、実行の初期段階では広い解空間を探索し、終了段階では狭い空間を集中的に探索する戦略が実現できている。

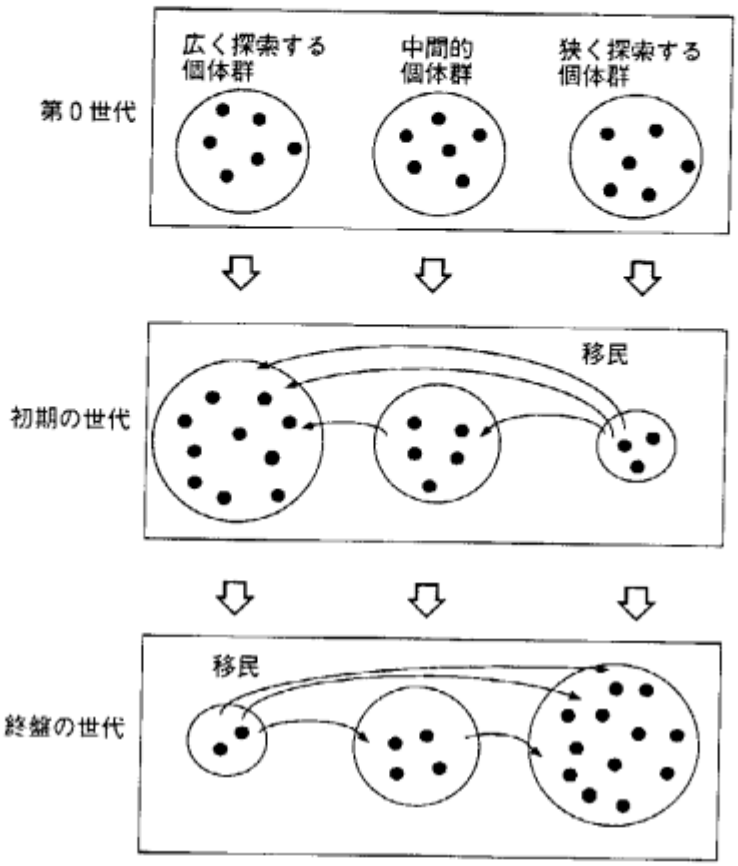


図 49: マルチ個体群方式の例

マルチ個体群方式は、自然界で近隣の生物種が、棲み分けしながらも競合し、環境の変化に柔軟に対応していくモデルに似ている。その意味で、生態学的シミュレーションの観点からも興味深い。

4.2 マルチプルアライメントへの適用

GA では、多くの「個体」からなる集団が、世代交代を繰り返すことで、「個体」の平均「適応率」をあげていく。「個体」を組合せ問題の解に、「適応率」を解の評価に対応させれば、組合せ最適化問題が解ける。本節では、マルチプルアライメントの問題を解くために、我々が行った定式化について、基本操作と、マルチ個体群とに分けて説明する。

4.2.1 基本操作の定式化

マルチプルアライメントを解く場合には、「個体」にマルチプルアライメントの解をもたせ、「適応率」にそのアライメントのスコアを対応させる。（つまり、解の表現にビットストリングなどは用いない。）

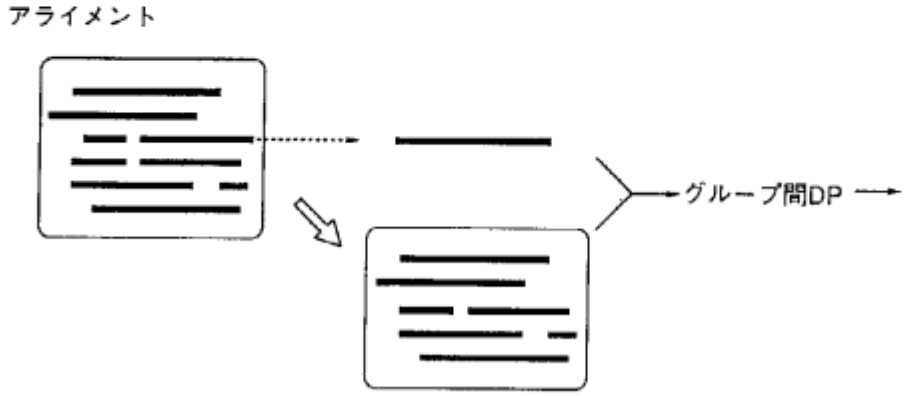


図 50: 突然変異の適用法

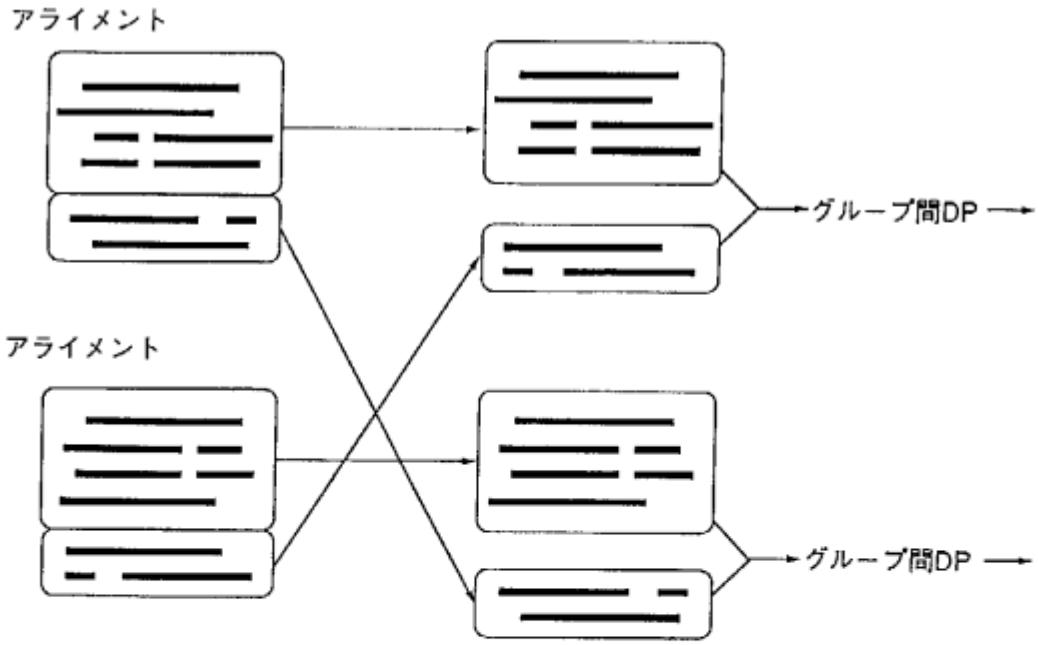


図 51: 交配の適用法

そして、変異、淘汰、繁殖、交配、個体数を、アライメント問題に即し、次のように適用する。

変異： 変異は、各個体に独立に行われる。個体がもつマルチプルアライメントから、ランダムに1本を抜きだし、残りとグループ間DPする（図50）のがひとつの変異である。ひとつの変異は、前章で議論した、1本限定分割の反復改善法の1サイクルに相当する。伝統的なGAで突然変異というと、スコアが良くなる場合も悪くなる場合もあるが、この適用では、スコアは悪くなることはない。

淘汰： 淘汰は、個体群全体を調べ、低いスコアの個体（アライメント解）を捨てる操作である。

繁殖： 繁殖は、淘汰によって残った個体群から、ランダムに選んだ個体をコピーする。

交配： 交配は、個体群から2つの個体をランダムに選び出し、部分解を交換する。選ばれた個体のアライメントされた配列群を、ランダムに2つの配列グループに分け、それらの片側の配列グループを交換して、グループ間DPで融合する（図51）。これにより、アライメントのなかに、比較的良く揃っている部分があれば、他のアライメントにある、別な良く揃っている部分と、互いに融合され、ひときわ高い評価のマルチプルアライメントができる可能性が生まれる。

個体数： 淘汰によって失われた個体は、繁殖によって生まれた個体で補われるため、通常は個体群の個体数は一定である。しかし、次で述べるマルチ個体群の方式では、移民などの操作により、各個体群の個体数は増減する。

このようにマルチプルアライメントの解法を定式化すると、淘汰率の設定により、解探索の戦略が調整できる。今、交配は行わないとし、淘汰率を0%に設定すると、前章で議論した、1本抜き限定分割のマルチ山登り反復改善法と等価である。淘汰率を上げていくに従って、探索は、最良優先探索の色彩が強くなってくる。なぜなら、淘汰率が高いと、スコアの高い解のみが残って、かつ、それらのコピーが次々に生まれ、その時点での最良解に近い解空間が集中的に探索される。

4.2.2 マルチ個体群の形成

前章で議論した、並列反復改善法を振り返ってみる。マルチ山登り法であると、最終的に良い解に至る可能性が高いのだが、収束にたいへん時間がかかった。最良優先探索法であると、収束は早いのだが、局所解に捕われがちであった。我々は、GAのマルチ個体群方式を利用して、マルチ山登りと最良優先探索の両者の利点を生かし、局所解に捕われにくく、

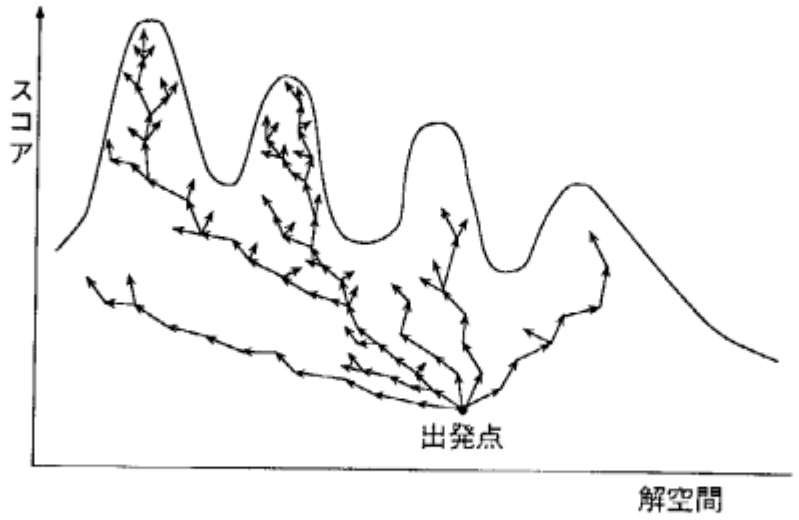


図 52: マルチ個体群による探索の概念図

かつ、収束も早いシステムの構築を実現した。具体的には、図52に示すように、解空間を全体的に多点探索する一方、有望な良い解の周辺は、局所的に集中探索する。

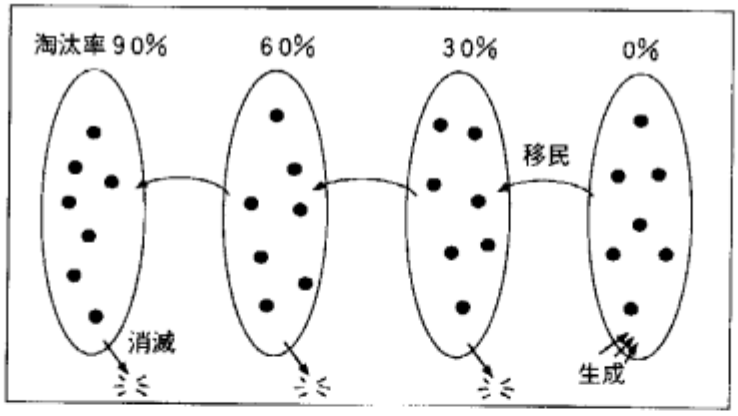


図 53: アライメント問題を解くマルチ個体群

我々のマルチ個体群方式の適用では、図53に示すように、淘汰率を変えた個体群を４つ用意している。基本的な発想は、マルチ山登り法に近い探索をしている（淘汰率が低い）個体群で見つかったスコアの良い解は、最良優先探索に近い（淘汰率が高い）個体群へ移し、集中的な探索をする。一方、局所解に陥って来た個体群は個体数を減らし、代わりに、解の多様性が残っている淘汰率の低い個体群の個体数を増やす。移民、消滅、生成の操作は以下のようにする。

移民： 淘汰率が隣接している２つの個体群を比較し、淘汰率の高い個体群の最大スコアの

解以上の良いスコアの個体が、淘汰率の低い個体群にあったら、それらを全て淘汰率の高い方の個体群へ移動させる。

消滅： 最もスコアの良い解と同じスコアの解が個体群の半分を越えたら、そのスコアの解の半分を捨てる。（同じスコアでも異なった解の場合があるが、アライメントの同一性を調べるのは手間がかかるのでスコアで代用する。）

生成： 解の多様性が、最も保存されていると考えられる淘汰率0%の個体群で、繁殖を行い、消滅によって減った個体数相当の個体を新たに生み出す。

4.3 実験と結果

前節で述べたGAのマルチ個体群方式を利用したシステムを、並列推論マシンPIM/mの上に実装した。まず、交配なしの性能を、前章で議論した並列反復改善法と比較する実験を行った。次に、交配を導入して、性能の向上を図った。

4.3.1 並列計算機への実装

前章の並列反復改善法と比較するため、実装形態は、前章のシステムと極力同一とした。PIM/mの多数のPEのうち、PE0はマスタープロセッサとし、全体の包括的処理にあてる。他のPEには、各個体（アライメント状態解）を割り当て、スレーブプロセッサとする。マスタープロセッサは、群間の移民、消滅、生成処理と、群内の淘汰、繁殖処理、そして交配の割り当てを行う。一方、各スレーブプロセッサは、群内の変異処理、割り当てられた交配の具体的な実行を行う。

ひとつの世代は、いくつかの変異を一定時間（現在80秒にしている）行うことで始まる。各スレーブプロセッサは変異を1回終えるたびにシステムタイマーを見て、80秒が経過していたならば、その時点の解の状態をPE0へ送る。PE0は、群間処理（毎世代行う）を行い、続けて群内処理を行う。その結果、スレーブプロセッサによっては、新たなアライメント状態が割り当てられる。この時点でシステムタイマーはリセットされる。割り当てが行われたスレーブプロセッサは、保持している解を捨て、新しい解について変異処理を始める。つまり、スレーブプロセッサは、交配が割り当てられない限り、常に変異処理を行っているのである。

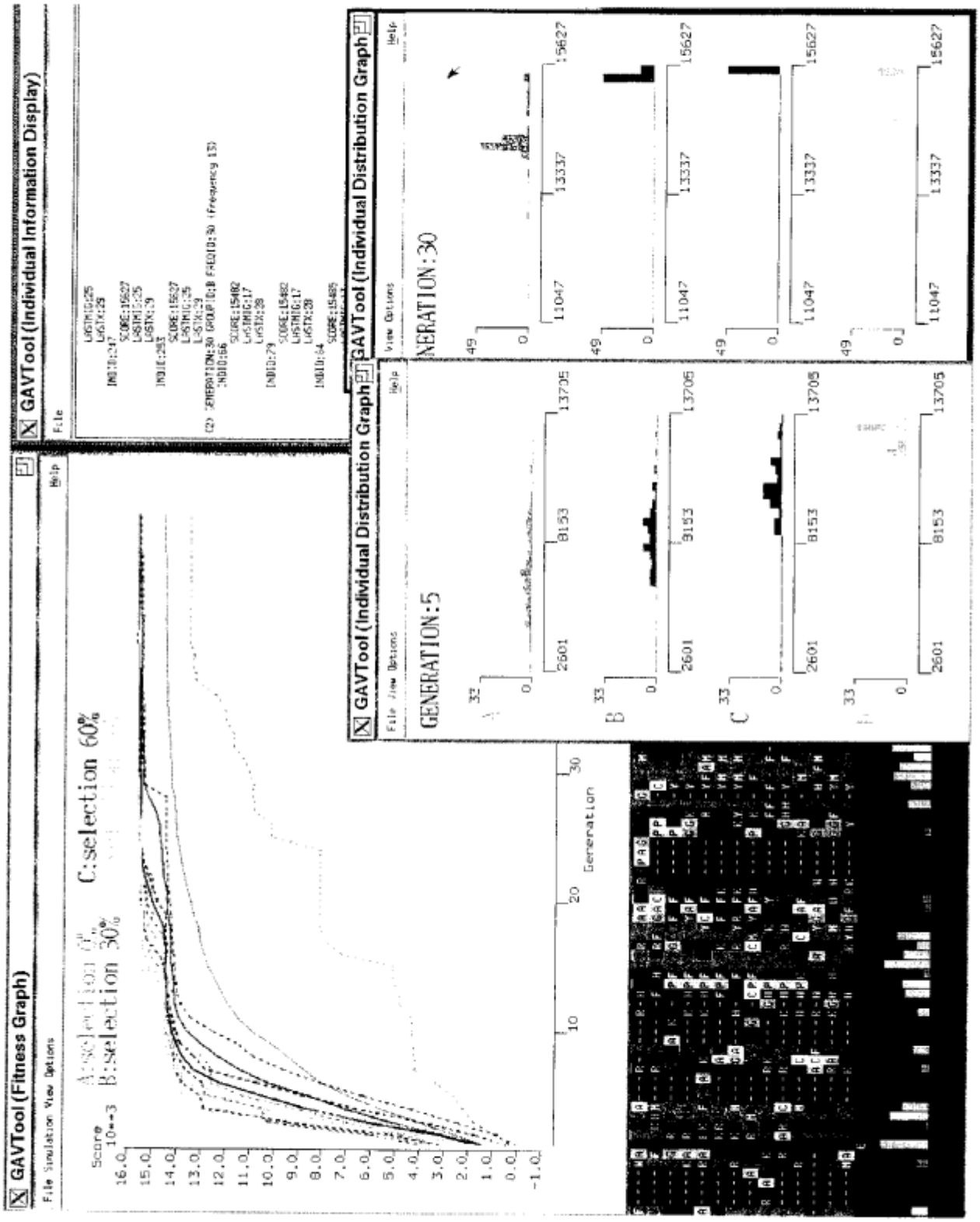


図 54: GA 実行の性能モニタ（交配なし）

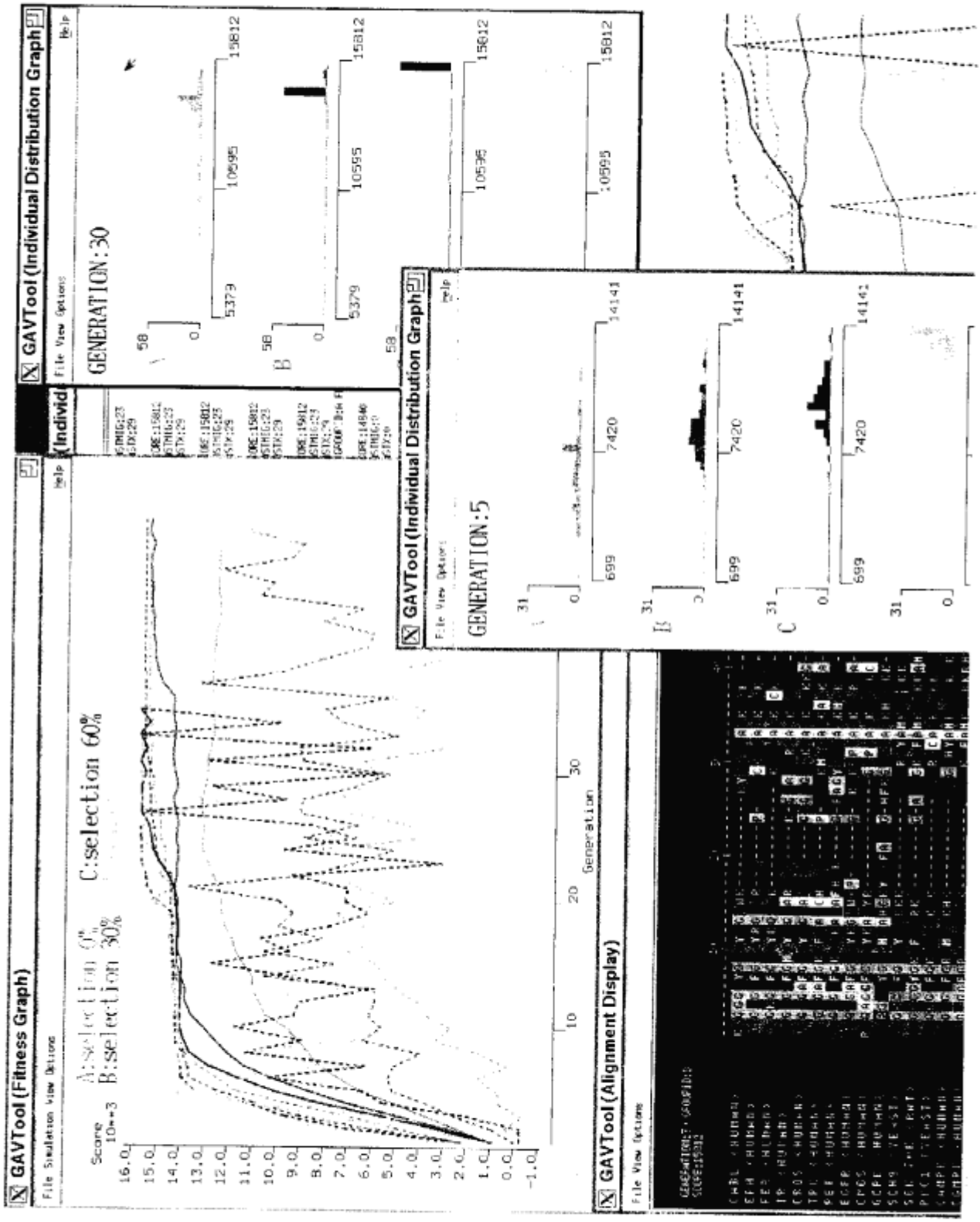


図 55: GA 実行の性能モニタ（交配あり）

交配が割り当てられたスレーブプロセッサは、変異処理に入る前に交配の実行（２つの部分アライメントのDPによる最適化）を行わねばならない。交配の実行は変異１回（１本配列と残りのアライメントとのDPによる最適化）より少し時間がかかる。80秒の世代時間に、変異は数回行えるが、最初に交配を実行してから変異処理に入ると、変異回数は１、２回少なくなる。

個体群は、各スレーブプロセッサの解に付されたラベルによって区別されており、個体群ごとに、解がまとまって管理されているわけではない。だから、移民がなされても、個体の属する個体群のラベルが変わるのみで、スレーブプロセッサ間を解のデータが移動することはない。よって、マスタープロセッサが集中的な処理をしても、PE間のデータ転送はわずかである。PEの稼働率は、実行全体で98％以上であった。

我々は、GAの実行中の性能をオンラインでモニタするツールも開発した。図54，55は、それぞれ、２時間程度の交配なしの実行、そして交配10％を加えた実行をモニタした画面である。このツールにより、各個体群の平均（実線）、最大（点線）、最小（点線）スコアが折れ線グラフ表示できるほか、世代ごとに、各個体群のスコア分布や詳細情報が表示できる。もちろんアライメントの内容も表示でき、GAの効率を向上させるための検討に、有効なツールとなっている。

4.3.2 並列反復改善法との比較

前に述べたように、我々のマルチ個体群の定式化は、１本抜き限定分割の反復改善法の改良方式に相当する。そこで、図45の結果と比較するため、まず最初に、総個体数を253とし、交配なしで実験した。群内個体数の初期状態は、淘汰率90,60,30,0％の群が、それぞれ63,63,63,64個体とした。テストデータにも前章と同一の30問を使用した。

図56は、図45のうち、１本抜き限定分割だけを抜き出して、今回の実験結果を加えたものである。折れ線（●）が、交配なしのマルチ個体群方式の結果である。その線は、最良優先探索（▲）と同様な早さで、同程度のスコアの解に達しているうえに、マルチ山登り（△）と同様な高いスコアに早期に収束している。この結果は、マルチ個体群方式により、前章の並列反復改善法にまさる、性能の良い並列探索が実現できたことを意味する。

図54を詳細に見ていくと、マルチ個体群の実行状況がよくわかる。折れ線グラフの各個体群の平均スコア（実線）を比べると、やはり、淘汰率の高い個体群の方が早期に収束傾向を見せている。最大スコア（上側の点線）の変化をみると、淘汰率の低い個体群の最大スコアが、移民によって淘汰率の高い個体群へと移動している。棒グラフで、各個体群のスコア

分布を比べると、第5世代では淘汰と移民によって、淘汰率の高い個体群がよりスコアの高いところに分布する傾向が出ている。第30世代になると淘汰率の高い個体群の解が局所解になり、消滅・生成の作用で淘汰率の高い個体群の個体数が減り、代わりに、淘汰率の低い個体群の個体数が増えている。画面には明示されていないが、第50世代を越えるあたりから、淘汰率の高い個体群の個体数は数個となって変化がなくなり、淘汰率の低い個体群から良いスコアの解が移民してくるのを、単に待つだけの状態となる。

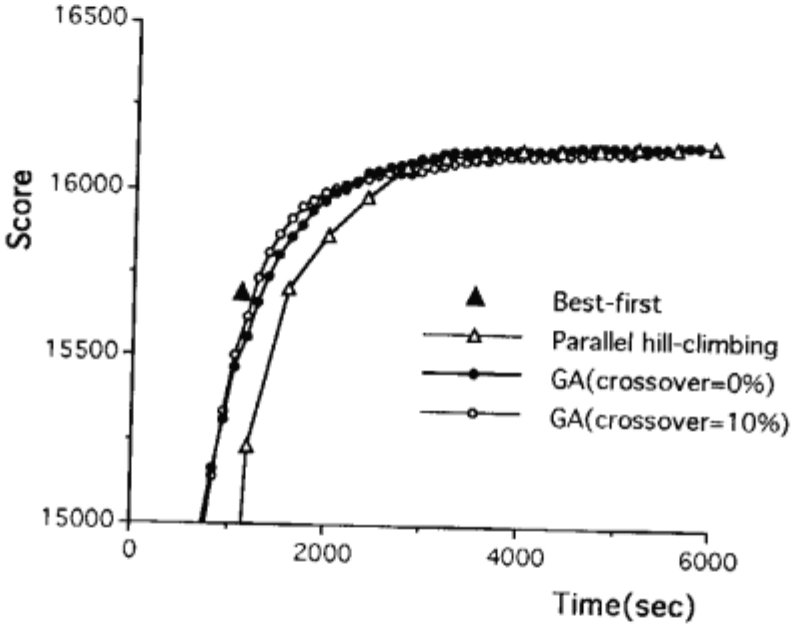


図 56: GA による改善履歴比較

4.3.3 交配の導入

次に、各個体群に10 %の交配率を導入して、同一のテストデータについて実験を行った。10 %の交配率とは、個体群の個体数の10 %を越えない最大の偶数だけ、ランダムに個体を選んでペアを作り、交配を適用する操作である。図56の折れ線（○）は、交配を導入した実験の結果を示している。

交配ありの遷移を見ると、交配なしより若干早い立ち上がりではあるが、収束は少し遅いことがわかる。この結果は、交配に関して次の効果を反映したと推測できる。実行の初期段階では、交配の操作により、良い部分解を組み合わせる現象が起き、より良い解が早期に生まれた。その反面、収束段階では、交配が解をシャッフルする作用を及ぼし、収束を遅らせた。収束が遅い分、解の多様性を保持するのであるから、理論的には、収束解のスコアが良

くなる可能性があるのだが、今回、収束解に際立った差異は得られなかった。この結果は、交配を導入しなくとも、収束解は十分良かったか、あるいは、交配を導入した試行の実行時間が十分でなかったことが原因に考えられる。

図55を詳細に見ると交配の影響がよくわかる。図54との際だった違いは、折れ線グラフの各個体群の最小スコア（下側の点線）が世代ごとに大きく変動していることである。とくに、淘汰率の最も低い個体群の最小スコアは、世代が進むにつれて、一旦向上してから徐々に悪化する傾向を見せている。これは、まさに交配の操作により、実行の初期段階では良い部分解をうまく組み合わせる現象が起きやすいのに対し、後の段階では異なる収束解の組み合わせがうまくいかず、悪いスコアの解ができやすいことを示している。棒グラフの方で、各個体群のスコア分布を比べると、ほぼ図54と同様の傾向が見られる。ただ、横軸に注目すると、交配の効果でスコアの分布が広くなっているのがわかる。

4.4 考察

我々は、遺伝的アルゴリズムをマルチ個体群の方式で並列計算機上に実装し、実用規模のアライメント問題でも、準最適な解を比較的高速に求めることを可能とした。本システムは、前章で述べたマルチ山登り並列の反復改善法より、早期に収束に至り、かつ、最良優先探索並列のような局所最適解に陥りにくいことが、実験の結果から明らかとなった。また、交配の導入により、初期段階のスコアの向上が早まった。本節では、マルチ個体群と交配の改良法をさらに検討し、次に、その他の今後の課題を述べる。

4.4.1 マルチ個体群方式の改良

本システムでは、次のように、マルチ個体群の定式化を行った。すなわち、淘汰率の低い個体群で、解空間の広い範囲を探索し、淘汰率の高い個体群で、スコアの比較的高い、ピークの周辺を集中的に探索しようと意図した。今回、この意図に沿った結果が得られたが、依然として、局所最適解に陥る問題点もわずかながら残されている。この問題点の根本的解決は困難であろうが、改良の方針をもう少し検討してみよう。

淘汰率の高い個体群での探索は、初期段階では複数のピークに渡っている。しかし、淘汰の作用で、その時点でスコアが高い探索点を含むピークが選ばれていく。このとき、探索点のスコアがまだ低かったために、実は、結果的に高い方のピークが探索範囲から捨てられている可能性がある。せっかく探索点は高い方のピークにあったのに捨てられるのは惜しい。なんとか利用する方法はないだろうか。

ひとつの案は、個体群のなかの複数のピークに渡った探索点をピークごとに分類し、各々異なった個体群に分割することである。そうすれば、別々の個体群において、それぞれピークが探索される。この分割を行うには、探索点を異なったピークに属するものとして区別する、問題に応じた基準が必要となる。

本システムの場合、探索点はアライメントの状態であるし、変異はDPの適用であった。そのため、探索点の属するピークを、アライメントの状態から区別するのは難しい課題となる。しかし、アライメントから系統樹を書かせるとある程度の指標が得られるであろう。系統樹のトポロジーが異なるアライメントは、異なるピークを探索しているとみなすと、効果的な結果が得られるかも知れない。

また、こうした個体群の分割を行うと、個体群の数が計算資源を越えて膨大になる可能性がある。そこで、ピークの順序づけの基準を設け、有望な（高そうな）ピークから優先的に探索する必要が出てくる。その時点でスコアが高い探索点を含むピークを優先するのが、実装として容易であるが、問題に応じた良い基準が得られれば、探索効率の向上が期待できる。アライメントの問題では、やはり系統樹から基準を得るのが得策だろうか。今後の検討が必要だが、いわゆる簡潔な系統樹 [81] ほど、有望なピークを示しているという仮説が有効そうである。

4.4.2 交配操作の改良

前に述べたように、交配の操作には次の２つの機能がある。ひとつは、解の多様性を維持するために、解をシャッフルする機能、もうひとつは、良い部分解を組み合わせることで、効率良く解のスコアを向上させる機能である。

本システムの交配では、アライメントをランダムに２つの部分アライメントに分けている。そのため、解のシャッフルは盛んになされているものの、良い部分解の組合せは偶然に任されている。良い部分解を積極的に推定して交配を行うと、最適化の性能が飛躍的に高まる可能性がある。

良い部分解の推定にも、系統樹の分析が有効である。交配される片方のアライメント状態に対応する系統樹を描き、前章で述べたツリー依存分割（ただし１本抜き分割は交配として意味をなさないので除く）を行う。そうすると、交配される部分アライメントが、その時点で整っている配列群に限定され、最適化性能の向上が期待できる。ただ、解のシャッフル機能はどうしても低下するので、注意を要する。

これまで、交配の操作には配列方向（アライメントに対して横方向）の分割のみを考えて

きたが、カラム方向（縦方向）の分割も検討すべきである。交配が可能なようにカラム方向に分割するには、工夫が必要である。

交配される２つのアライメントにおいて、同一のアライメントがなされているカラムを見つけ、そこで分割するのが最も良い[93]。しかし、配列の本数が多いアライメントでは、そうしたカラムがないことも考えられる。片方のアライメントをランダムに選んだカラムにおいて、縦にまっすぐ分割し、他方を対応する部分でギザギザに切ったあと、切口にギャップを埋めてまっすぐにするという手段で、とりあえずの交配実験を行うと良いだろう。

4.4.3 今後の課題

GAは、変異、交配、淘汰を工夫でき、柔軟で使いやすい枠組である。さらに、個体間の情報交換がわずかで済み、並列計算機上への実装も容易である。ここでは、並列実装に関連して、今後の課題を述べよう。

今回の実装法では1 PE当り1個体を割り当てたが、1 PEに複数個体を割り当てて、比較実験をしてみるのも興味深い。計算資源が同一なので、大きな差はないようにも思うが、多点探索の傾向が助長されるので、問題によっては性能向上が見られるかも知れない。

マルチ個体群の実装では、今回は淘汰率を変えて行ったが、今後は、個体群によって交配率も変える枠組も検討すると良いだろう。実行が進むに従って交配の効果が変わってくるという、今回得られた実験結果は、効率の良い探索には交配率も変化させることが有効であると暗示している。

また、個体間の情報交換がわずかといっても、今回のように、マスタープロセッサで集中管理していると、将来、PE数が数千にもなった場合には、マスタープロセッサに通信負荷の問題が発生する。マスタープロセッサを複数化すれば、ある程度解決できるが、GAの枠組を利用した解決法も考えられる。

解決法のひとつは、個体群の個体群といった、個体群の階層性を導入することである。人間の社会では、しばしば、こうした階層性でコミュニケーションの負荷を軽減している。社会の模倣という観点からも面白い研究になるかも知れない。

もうひとつの解決法は、マスタープロセッサを廃止して、物理的に近いPE同士のみが、局所的にデータを交換（交配、淘汰＋繁殖）する方法[87]の導入である。この方法を用いると、世代が明確ではなくなってしまうが、GAの最適化のメカニズムは維持できる。自然界の生態系のシミュレーションという意味からは、この方法の方が適しているだろう。ただ、最適化過程の制御は、はるかに難しくなる。

5 並列アライメントのワークベンチ

本章では、ユーザーのインタラクティブな操作で、並列計算機を使いながらマルチプルアライメントの問題を解くワークベンチ [94] について、報告する。本ワークベンチは、マルチプルアライメントを行っている途中の部分配列を指定し、指定部分のみのアライメントを並列計算機で高速に行う機能を有する。また、ユーザー指定のアミノ酸を揃えるという制約を課したうえでの、アライメント実行も可能である。

本章の構成は次の通り。まず、これまでの類似ツールが持つ問題点を述べたのち、我々が開発したワークベンチについて、その基本概念を解説する。次に、ワークベンチの使用法を説明し、続いて、考察を示す。

5.1 従来のアライメントワークベンチ

本節では、生物分野におけるインタラクティブなツールの動向を概観したうえで、従来のアライメントワークベンチの問題点を指摘する。

5.1.1 生物分野におけるツールの動向

生物分野では、安価に導入でき、しかも操作性もよいことから、マッキントッシュが非常に多く使われている。最近では、マッキントッシュ上で動く手軽な（そして多くは無償公開の）ソフトウェアで、ゲノム地図作成、配列パターン検索など、さまざまの解析ができるようになっている。生物関係の雑誌では、こうしたマッキントッシュ上のソフトウェアを紹介する連載を次々と行っている（たとえば、細胞工学 Vol.11、実験医学 Vol.10）。

しかし、研究者が計算機とインタラクティブに課題解決していく場合には、応答性も保持しながら多くの計算をこなす必要があり、ワークステーションレベルの計算機が使われる。たとえば、薬の設計がその代表的な課題であり、Sparc Station などのワークステーションが多く使用されている [95, 96]。薬品設計では、過去の膨大な情報を参照しつつ、専門的なノウハウを盛り込める操作環境と、シミュレーションなどの科学計算を行う機能とを必要とする。タンパク質などの複雑な分子の立体構造を、いろいろな角度から表示するニーズも多く、それにはグラフィクスに強いワークステーションであるシリコングラフィクス社の IRIS が、もっぱら使われている [97]。

アライメントのワークベンチは、自動アライメントの機能、研究者が手直しする機能、アライメントから情報を引き出す機能などが必要で、その計算量を考えるとワークステーションレベルの計算機が必須となる。事実、アライメントのワークベンチを搭載しているツー

ル群は、ワークステーションや、VAXなどの中型計算機上で動くものである。代表的なものに、ウイスコンシン大学で開発されたGCG、テキサスシステム大学で集められたGen-Tools (アライメントにはSEA[98]を搭載)、Steven Smithらが開発したGDE (Genetic Data Environment) がある。

5.1.2 従来のワークベンチの問題点

アライメントのワークベンチは古くから必要とされてきたが、それは、生物学的な意味で完璧な自動アライメント法が存在しないからである[99, 100]。自動的に完璧なアライメントが生成できないとすると、必然的に、自動アライメントの結果を研究者の観点から修正するなど、インタラクティブな環境が必要となってくる。最近になって、計算機の処理速度向上、ユーザーインターフェース構築ツールの完備などが進み、アライメントのワークベンチが次々に作られるようになった。しかし、それらは依然として、次の問題点を持っている。

- 自動アライメントの品質が低い。どのワークベンチも、自動アライメントにツリーベース組合せ法 (GCGはProgressive Alignment[35]を、GDEはCLUSTAL[36]を使用) や、それより信頼性の低い方法を採用しており、研究者の手作業での修正が多くなっている。

- 研究者の観点が盛り込みにくい。研究者はふつう、配列のうち、この部分とこの部分はアライメントされやすいなどの、事前知識を持っている。しかし、どのワークベンチも、そうした情報を活用できる機能を持っていない。

5.2 本ワークベンチの基本概念

我々の開発したワークベンチは、上に述べた従来の類似ツールがもつ問題点を克服し、アライメントを行う研究者に快適な環境を提供するものである。本節では、本ワークベンチの基本概念を、一番目の問題点を解決する並列反復改善法による部分アライメントと、二番目の問題点を解決する制約つきアライメントとに、分けて述べる。

5.2.1 並列反復改善法による部分アライメント

本ワークベンチでは、最初にアライメントすべき全体の配列を暫定的にアライメントし、その結果を部分的に見ていき、生物学的な観点を盛り込んだ最終的なアライメントをインタ

ラクティブに構成していく。暫定的なアライメントには、信頼性は十分ではないが、おおよその配列の対応関係が高速に求まる、ツリーベース組合せ法を用いる。

詳細な部分アライメントを行う時は、指定した部分配列群を初期状態とした、並列反復改善法を使用する。並列反復改善法にはいろいろな方式がある（第3章）が、そのなかでも1本抜き限定分割の最良優先探索を採用している。その理由は、インタラクティブに使用するため、ある程度の応答性が必要なことと、同じ問題を実行するたびに解が異なるのはユーザーの混乱を招くので、乱数を使う方式を避けることを考慮したためである。

指定した部分配列群の並列反復改善法は、部分ごとにユーザーが指定するギャップコストで行え、その結果は、第3章で示したように、ツリーベース組合せ法を上回る品質を持つ。また実行時に、次に述べる制約を考慮することが可能である。

5.2.2 制約つきアライメント

本ワークベンチでは、研究者が持つ事前知識を、制約のかたちでアライメントに盛り込むことができる。縦に揃えたい文字を連続的にクリックする（画面にカーソルを合わせてマウスのボタンを押す）ことで、制約の指定ができる。並列反復改善法や、ツリーベース組合せ法は、自動アライメントの際に複数の制約を考慮しながら実行でき、結果のアライメントでは、制約のかかった文字は必ず縦に揃っている。

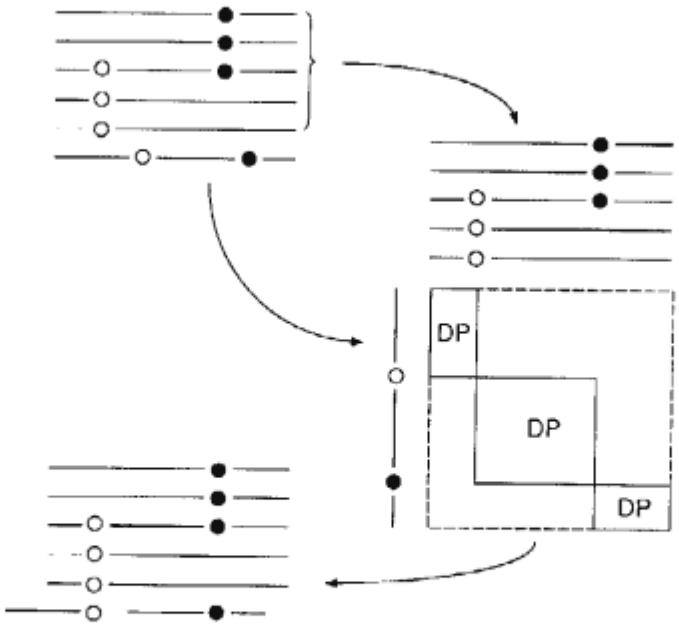


図 57: 制約つきアライメントの改善処理

たとえば、並列反復改善法のなかの1つの最適化では、図57に示す処理で制約部分が強制的にアライメントされる。左上の状態のアライメントが、6本の配列群から最後の1本を抜き出し、制約つきの配列間最適化処理で、左下のアライメントとなっている。○と●が、それぞれ一連の制約を表しており、制約つきの最適化処理で縦に揃う。最適化処理をみると、抜き出された1本の配列が、2つの制約によって3つの部分に分けられ、3つの独立して実行できるDP処理となっている。

並列反復改善法では、同一カラム位置に制約が課されている複数の配列については、それらの配列間のアライメントは動かさない機能も装備されている。ユーザーは、アライメントを動かしたくない配列群には、同一カラムに制約をかけて、ブロックモードを指定したうえで、自動アライメントを起動する。ブロックモードを指定すると、1本抜き分割を行う時に、同一カラムに制約が課されている複数の配列は、1本と見なされて分割が生成される。ブロックとなった配列群が類似した配列で、かつ、それらのアライメントがすでに整っている場合、ツリー依存分割のような効果が出て、アライメントの品質が上がり、ときには実行時間も小さくなる。

制約はマニュアルモード（手作業でギャップの挿入／削除を行う）のときも指定でき、指定された文字は、ギャップの挿入／削除を行っても、必ず縦に揃ったまま保存される。また、アライメント処理はせずに、配列を単に横移動して、制約のかかった文字を縦に揃える機能も備えてある。これらの制約に、ユーザーが持つ事前知識が反映できる。

5.3 本ワークベンチの使用法

本節では、本ワークベンチの使用法を、一部具体例も交えて説明する。

5.3.1 計算機環境

本ワークベンチは、OSF/Motifが実装されたUNIXワークステーション上で動作する。これまで、SUN、IRIS、DECで動作実績がある。ソースコードはC言語で書かれており、約2.8万行ある。コンパイルされたオブジェクトコードは約1.7メガバイトである。

アライメントの処理は、ネットワークで結合された並列計算機に問題が送られ、処理が終り次第、結果が戻って来る。アライメントのプログラムは、第3章と同様に並列論理型言語KL1（約2千行）で書かれ、並列推論マシンPIM/m（PE256台構成）上で動く。1本抜き限定分割を用いているので、256本のアライメント問題まで並列効果がコンスタントに出る。制約がかかっているアライメントの処理は、1分割の内部まで並列性があるので、

PE が余っている場合はその処理も並列に割り当てられる。

KL1 に類似の KLIC[101] という言語で記述しておくと、プログラムをC言語に翻訳でき、汎用の並列計算機で動かすことができる。現在、アライメントの処理には、並列部分を KLIC で、DP の演算部分はC言語で直接記述したものも用意している。このソフトウェアは、Sparc Center （PE20 台構成）上で動作実績がある。Sparc Center は新型のチップを搭載しているため、DP の演算部分が PIM/m 1PE の約 20 倍の速度で動く。PE の台数は少なくとも、全体性能はそこそこ良い。また、CM5 にも移植を進めている。

なお、ツリーベース組合せ法などの、簡便な逐次アルゴリズムのアライメントプログラムは、ワークベンチに組み込んであるので、並列計算機を利用できない環境のユーザーにも、一応使えるようになっている。

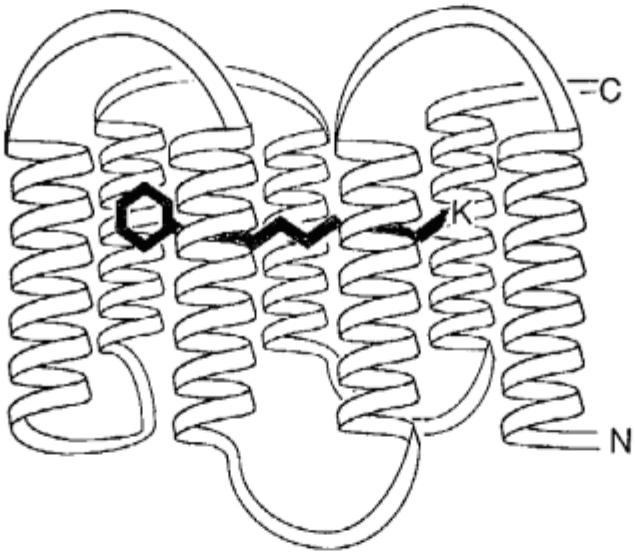


図 58: バクテリオロドプシンの立体構造

5.3.2 部分配列群のアライメント

ワークベンチで部分配列群のアライメントをどのように利用するかを、ロドプシンスーパーファミリーの配列を例にとって述べよう。ロドプシン (rhodopsin) は網膜視細胞の膜上にある視覚色素であり、内部にレチナール (retinal) 分子を 1 つ把持している。一方、高度好塩菌の膜表面にあり、内側の塩分を外側へ排出する光プロトンポンプの役目を果たすタンパク質も、ロドプシンの仲間とされている。このタンパク質は、バクテリオロドプシンと呼ばれ、ロドプシンと同様に、内部にレチナール分子を持つ（図 58）。

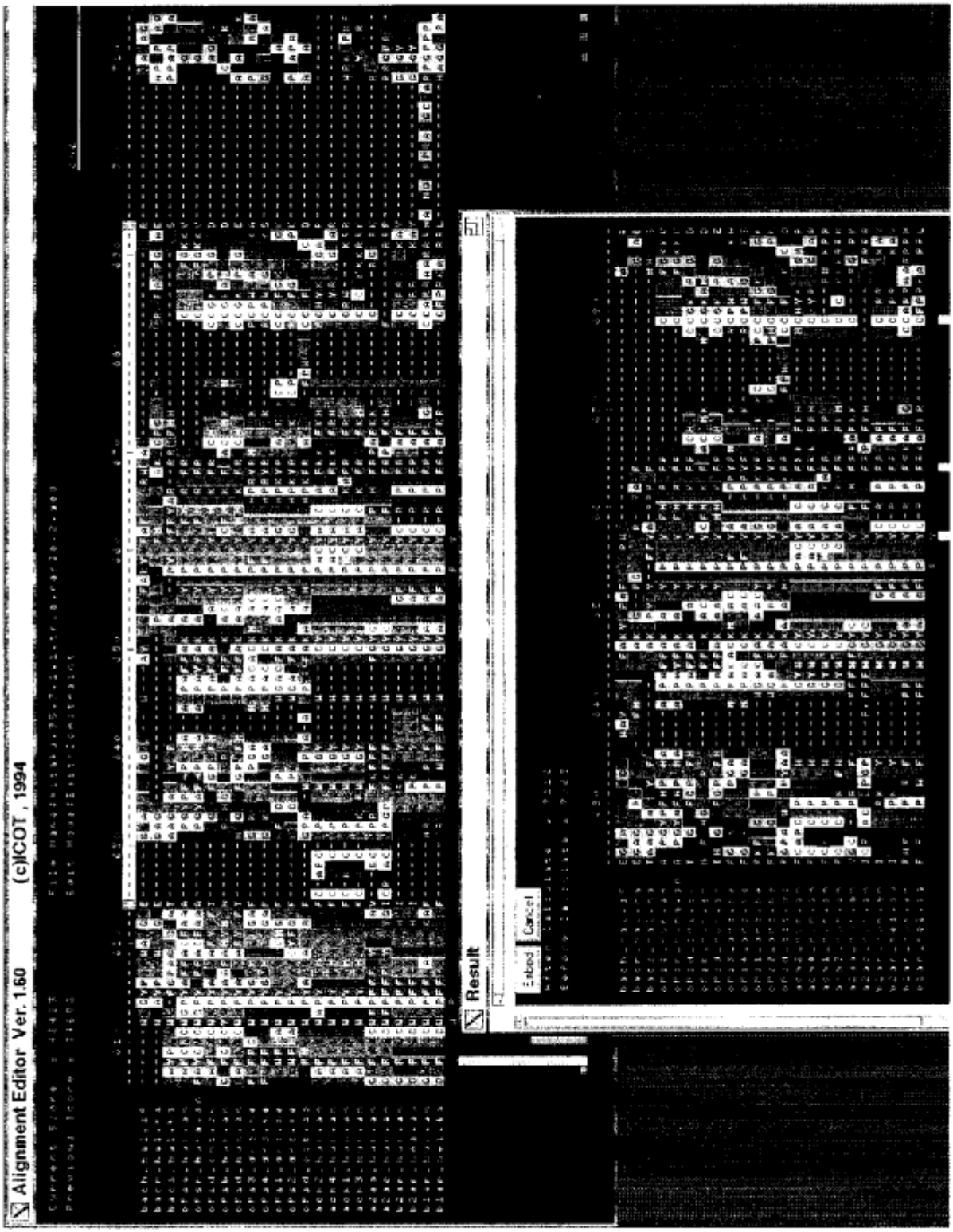


図 59: 制約アライメントの適用例

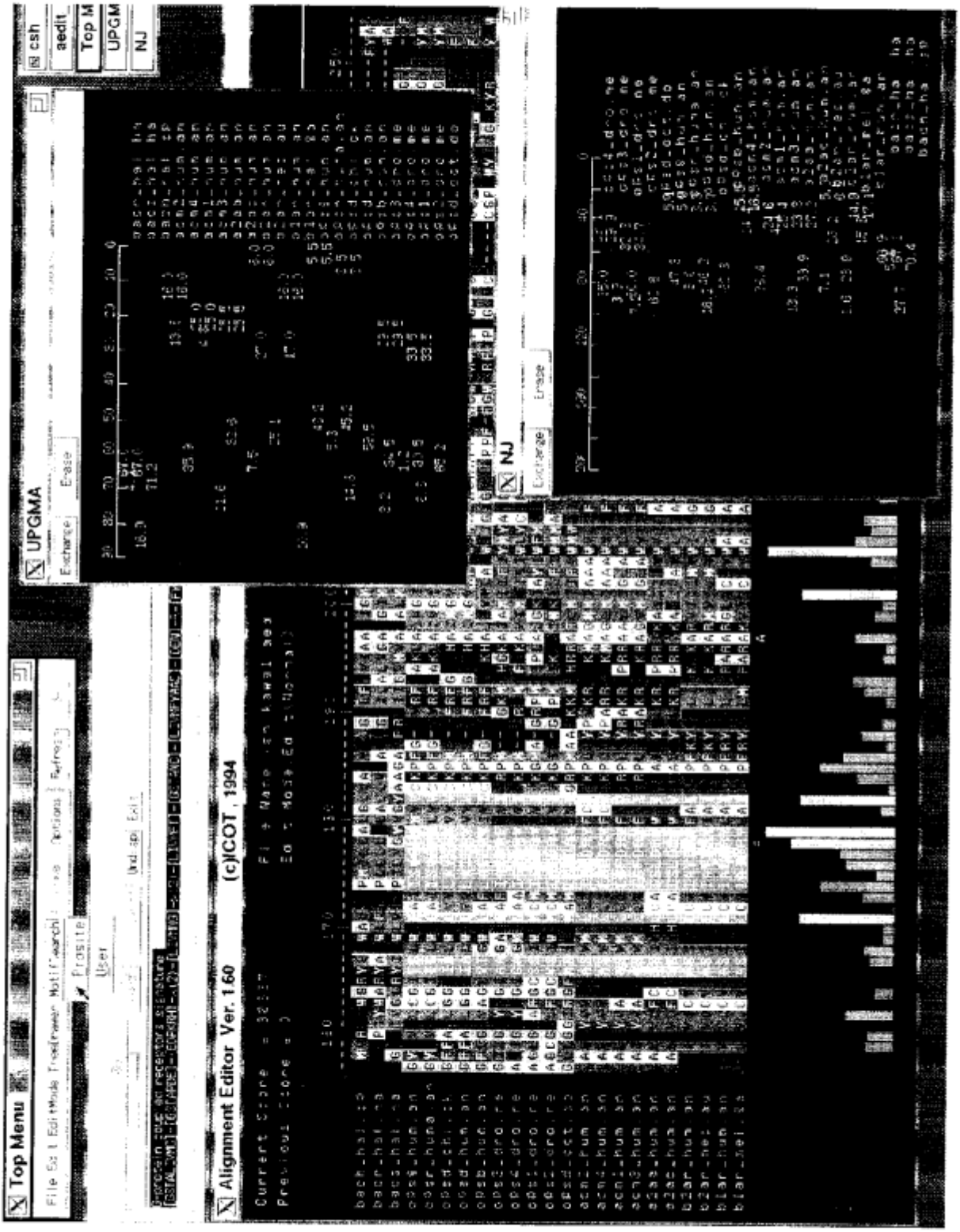


図 60: アライメント結果の解析画面

バクテリオロドプシンは、立体構造がX線結晶解析で判明しており、7本のヘリックスが、ほぼ膜面に垂直に並んでいる膜タンパク質である[102]。そして、一番C末端に近いヘリックスの中央にレチナールが結合している。また、アドレナリンなどのホルモン受容体も、レチナールは持っていないものの、同様な立体構造をしていることが知られている[103]。

図59の上部のウィンドウは、ロドプシンスーパーファミリーの配列23本（上からバクテリオロドプシンファミリー3本、ロドプシンファミリー10本、シグナル受容体ファミリー10本）をツリーベース組合せ法で暫定的なアライメントをした結果を、ワークベンチの画面で表示したものである。アライメントの長さは1,000カラム近くになり一度に表示できないが、スクロールの機能があり、図では、C末端側のヘリックスに相当する配列位置のアライメントが表示されている。このアライメントを見ると、バクテリオロドプシン類が、他の配列に比べホモロジー（相同性）が低く、十分にアライメントされてないことがわかる。

ここで、既知の知識を導入することができる。バクテリオロドプシンファミリー、ロドプシンファミリーは、レチナールを持っており、それがC末端側のヘリックス中央のリシン（K）に結合していることが知られている。すなわち、Kは機能上重要な役割をしているので、進化上強く保存されていることが容易に想像できる。となると、アライメントでもKが揃ってなくてはならない。そこでKに注目してみると、ロドプシンファミリーではKがしっかり揃っているが、バクテリオロドプシンファミリーでは揃っていない。一方、レチナールを持たないシグナル受容体ファミリーにはKがない。こうしたKに制約をかけることにより、アライメントが改善される可能性が高い。

上部のウィンドウで、本来の青色から暗色に変わっているKがいくつかあるが、マウスのクリックにより、制約がかかったことを表している。また、カラム番号部が白色になっている領域は、マウスの操作で、再アライメントの範囲が指定されたことを示している。

下部のウィンドウは、そうしたKの制約のもとで、並列反復改善法で再アライメントした結果を示している。この再アライメントでは、ギャップコストを高めにして実行している。膜内部のヘリックス部分にはギャップが入りにくいという知識に基づいたからである。アライメントを見ると、制約をかけたKが揃っており、ギャップも少なく、まとまった感じのアライメントとなっている。ユーザーは、この部分アライメントの状態を吟味し、良いアライメントと判断できる時は、Embedのボタンをクリックして、上部のメインのアライメントへ組み込む。再アライメントしても気に入らない部分は、マニュアルで手直しすることも可能である。

5.3.3 その他の解析機能

本ワークベンチは、アライメントの結果を解析する2つの機能も、合わせて持っている。それは、モチーフ抽出と系統樹解析である。

モチーフ抽出 モチーフ抽出とは、アライメントのなかから、既知のモチーフを探し出す機能で、すでに知られているモチーフを集めて登録した、Prositeデータベース[104]を検索し、アライメントとマッチングをとる。図60の左側は、先のロドプシンスーパーファミリーのアライメントから、Gタンパク質結合モチーフが、80%以上の配列から見つかったということを示している。見つかったモチーフの部分はアライメントのなかで、灰色に表示されている。ロドプシンスーパーファミリーでは、外部からの刺激を、Gタンパク質[105]が細胞内に媒介伝達、増幅しているので、こうしたモチーフが見つかるのも納得がいく。

モチーフ抽出は配列1本でも可能であるが、モチーフ部分といえども多少の突然変異があるのが普通で、配列1本では見つからないモチーフも、マルチプルアライメントでマッチングすることにより、確実に捕えることができる。未知のタンパク質配列のアライメントでは、このモチーフ抽出により見つかったモチーフの特徴から、構造や機能の推定が行えることがある。また逆に、モチーフを手がかりに、アライメントの質を向上させることもできる。大部分の配列にモチーフが見つかった場合には、モチーフが見つからない配列のアライメントが誤っていると疑ってみると、生産的なことが多い。

我々は、Prositeデータベースの情報を階層的に管理し、「あるモチーフが見つかったならば、ほかにどのようなモチーフが見つかる可能性が高いか」などの、簡単な質問に応答できるシステムも作成した[106]。

系統樹解析 系統樹描画には、平均距離法（UPGMA[34]）と近隣結合法（NJ[107]）を実装している。アライメントにおいて、各配列ペア間の置換、挿入、欠失の総イベント数を数え上げ、イベント数を距離に換算して、系統樹を描く。平均距離法では、進化速度は一定という仮定のもとで計算するため、系統樹の右端が揃うが、近隣結合法は、各枝の進化速度は異なるとするので、系統樹の右端は揃わない。近隣結合法では、系統樹の根元は決定できないのだが、見やすさを考慮して適当に決めて表示している。

図60の右側には、ロドプシンスーパーファミリーのアライメントをもとに、これら2つの方法によって描かれた系統樹が表示されている。どちらの系統樹においても、バクテリオロドプシンファミリー、ロドプシンファミリー、シグナル受容体ファミリーが、それぞれファミリーごとに分離されている。2つの方法によって描かれた系統樹のトポロジーが同

じであると、系統樹の信頼性が高いとされている [11]。また、ウィンドウの上部にある Exchange のボタンをクリックすると、系統樹に従った配列の順番に、アライメントされた配列の順番が変更される。

系統樹解析は、アライメント途中の状態を吟味するのにも有効である。アライメントしている配列群のうち、ある配列同士が仲間であるという事前知識があったならば、系統樹を描かせることで、そういった事前知識を反映したアライメントであるかどうか、チェックできる。

5.4 考察

我々は、ユーザーのインタラクティブな操作で、並列計算機を使いながらマルチプルアライメントの問題を解くワークベンチを開発した。本ワークベンチでは、ウィンドウ上に表示されているアライメント途中の部分配列を指定し、並列反復改善法を用いて、高速で高品質なアライメント処理を行える。

また、本ワークベンチは、アライメント途中に、ユーザー指定のアミノ酸を揃えるという制約を課したうえでのアライメント実行や、手作業による修正機能を有している。そのため、ユーザーが持つ事前知識の盛り込みや、配列の特徴を捉えながらのインタラクティブなアライメントが可能で、生物学の研究に有用なツールとなっている。

本ワークベンチは、すでに多くの生物分野の研究者に使用してもらっているが、本節では、本ワークベンチが特徴とする、自動アライメントと制約の機能について、現状の改善すべき点をまとめる。そして、その他の今後の課題を、最後に述べる。

5.4.1 自動アライメントの機能

現在のアライメントのスコア体系は、必ずしも生物学的な知見が十分に反映されているわけではないことが、こうしたインタラクティブなツールが必要となる大きな理由のひとつであった。並列反復改善法は、アライメントスコアの最適化を、指定した部分アライメントを初期状態にして強力に最適化を行うため、結果には初期状態のアライメントのかたちがほとんど残ってないことが多い。ユーザーとしては、初期状態のアライメントにある程度満足しているところに、少しの修正をかける意味で自動アライメントを起動することがよくある。そうした場合には、あまり厳密にスコアの最適化をせずに、ユーザーが良いと思うアライメントの傾向を残した解を提供したほうが望ましい。

幸い反復改善法は、類似性の高いところから改善サイクルごとに、徐々に揃っていくとい

う特徴をもっているので、改善サイクル数を調整することで、厳密な最適化を防げる。ユーザーが改善サイクル数を指定できるようにするか、あるいは、各改善サイクルごとの解を見比べて、ユーザーが選択できるようにするのがよいだろう。

強力な最適化は、モチーフなどの局所的なパターンを見つけるには不向きなことがある。アライメントした配列のほとんどに、あるパターンが見られるときに、残りの配列にも同様なパターンがないかと調べたいことがよくでてくる。こうしたパターンの探索には、グローバルマッチングでは不十分で、1.2.1 で述べたローカルマッチングの技術が必要になる。

本論文の研究では、主にグローバルマッチングを取り上げたが、より実用的なアライメントワークベンチを構築するには、ローカルマッチングの機能も備えておく必要性が感じられた。

5.4.2 制約の機能

本ワークベンチでは、事前知識が制約のかたちでアライメントに反映でき、ユーザーが所望とするアライメントを容易に作成できた。しかし、ユーザーとしても、いつも事前知識に絶対的な確信があるわけではない。あるファミリーの仲間と思っていた配列が、実は別のファミリーの仲間であったなどはよくあることである。これには、確信が持てない事前知識も、制約とできる機能があれば便利である。具体的には、制約設定時にその制約の確信度も一緒に入力できるようにする。そして、最適化のときに、制約の確信度もスコアに含めて最適化すると良い。これは、制約つき文字のマッチングスコアを確信度に応じた値（制約なしのスコアに比べてどの程度大きくするかの任意性はあるが）とすれば、DP の枠組で実現できる。

確信度つき制約があれば、これまでにも増した柔軟なアライメントワークベンチが構築できる。たとえば、図59でKを揃えようとした時に、ロドプシンファミリーの該当部分にKが２～３個点在していたらどうだろうか。どれが正しいレチナール結合部分か確信が持てないだろう。こうした場合、現状ではいくつかの組合せで制約をかけてみて、それをアライメントさせ、一番いいスコアの解を選ぶなどとするしかない。確信度つき制約があれば、２～３個のKに対して、それぞれ弱い制約をかけておき、スコアの最適化のなかで、結果的に最も妥当な制約が採用され、便利である。その他にも、現れてきたモチーフを弱く固定する際や、仲間と推測できる配列群のアライメントを、ある程度保存する時に役に立つ。

5.4.3 今後の課題

本ワークベンチは、たくさんの生物分野の研究者に使用してもらい、総じて好評なのではあるが、数々の要求が寄せられた。多くはちょっとした操作性の不満や、表示形式のクレームであるが、時々、機能の根本的な追加になる要望がある。一番、強かった（そしてまだ実装されてない）要望は、タンパク質の高次構造の考慮であった。たとえば、図58のロドプシンスーパーファミリーは膜タンパク質であり、膜を貫通するヘリックス部分と膜の外のループの部分では、大きく立体構造が異なる。こうした構造の違いが配列のアライメント時に良くわかると、立体構造を考慮してアライメントが行え、より価値のあるワークベンチになる。

膜タンパク質の膜貫通部分は、膜が油脂であることから、疎水性のアミノ酸が集中して存在することが知られている。もちろん、現状のワークベンチでも、アミノ酸が色分けされており、赤っぽい文字の存在で、膜貫通部分をおおよそ推定できる。しかし、もっと直接的に、疎水性尺度 [108, 109, 110, 111] で、文字を段階的に色分けし、カラムごとに尺度の合計がグラフに出るようにするとなお良い。球状タンパク質を扱う場合は、疎水性尺度は内外性（アミノ酸がタンパク質の中心に位置するか表面に位置するか）にだいたい相当するが、それより、二次構造（ヘリックス／ストランド／ターン）を予測する数値 [112, 113] を用いた方がアライメントには利用価値が高い。

さらに、部分配列から二次構造を予測しながらアライメントする機能などあれば、非常に有用なようだが、もはや、ワークベンチの機能の範囲を越えている。次章では、アライメントの際に高次の立体構造を考慮する試みの一端を議論する。

6 立体構造を考慮した並列アライメントシステム

本章では、RNAのステム（茎）構造を考慮しながらマルチプルアライメントの問題を解決するシステム[114]について、報告する。本システムは、並列反復改善法で暫定的なRNA配列のアライメントをした後に、並列シミュレーテッドアニーリングの方式で、それらのステム構造を考慮しながらアライメントの精緻化をする。立体構造の観点から信頼性の高いアライメント結果が得られるうえに、配列群が共通に持つステム構造を推定できる。

本章の構成は次の通り。まず、立体構造を考慮したアライメントについて、簡単に解説する。次に、我々が開発した、RNAのステム構造を考慮したアライメント法の紹介を行う。続いて、その実装法、実験結果、および考察を示す。

6.1 立体構造とアライメント

本節では、立体構造とアライメントとの関係について、タンパク質の場合と、RNAの場合とに分けて解説する。そして、本論文の主要テーマは、タンパク質の配列解析であるが、本章に限ってRNA配列を例題に採用した理由を明らかにする。

6.1.1 タンパク質立体構造とアライメント

アライメントの代表的な利用目的に立体構造の予測がある（1.1.3参照）。アライメントされた配列群のうち、一部の配列の立体構造が既知のとき、残りも同様な立体構造を持つだろうと推測するのである。しかし、もし複数の配列にわたって立体構造が既知ならば、逆にその情報をアライメントの際に使うと、立体構造の観点からアライメントの信頼性が上げられる。つまり、共通な部分構造を持つ配列部分は、同じカラムにアライメントされる傾向を導入すれば良い[115, 116]。

また、立体構造が既知でなくても、各配列の立体構造予測の結果に基づいてアライメントすることは有効である。アライメントの品質も向上するし、構造予測の精度も上がる。タンパク質の場合、二次構造（ヘリックス／ストランド／ターン）の予測の際に、類似配列を集めて、それらをアライメントしたうえで予測をすると、精度が良いことが知られている[117, 118, 119]。

二次構造とアライメントの研究は盛んに行われているが、高次の立体構造情報をアライメントに反映させるのは難しい課題である。なぜなら、高次の構造では、配列上遠く離れたアミノ酸同士が相互作用を行っているうえ、その相互作用の形態も多様である。我々は、高次

の構造を抽象化し、階層的な文字表記に表現する研究を行い[120, 121, 122]、高次の構造情報をアライメントへ導入する方法を探った。

その他にも、3次元座標そのものを類似性マッチング（空間的な距離を計算する）して、アライメントする研究もなされている。グルーバルマッチングを行う研究[123]もあるが、ほとんどはローカルマッチングを行っている（たとえば[124]）。

6.1.2 RNA立体構造とアライメント

本章の研究でRNAの立体構造に注目したのは、配列上離れた部分の相互作用を考えるうえで、単純化された問題であるからである。配列上離れた部分の相互作用は、タンパク質の場合、ストランド（直線状の鎖）やヘリックス（螺旋状の鎖）といった構造のレベルには含まれないが、それより、高次の構造には含まれる。かといって、タンパク質の高次の構造を相手にすると、複雑になり過ぎる。階層的な文字表記を使用しても、高次の立体構造情報のアライメントへの導入の道筋は十分に確立できていない。

一方、RNAの場合はステム構造が存在し、配列上離れた相補的配列部分が水素結合している。この構造は、タンパク質では、配列上離れたストランドが集まって、平面的なシートを形成する構造や、ヘリックスを会合させたロイシンジッパー[125]を形成する構造、そしてシステイン同上のイオウ2重化結合の構造に似ている。どれも、配列上離れた部分の相互作用である。ステム構造をアライメントに反映させる研究は、タンパク質の立体構造情報を考慮するなかの、ひとつの研究要素に相当していると考えてよいだろう。

RNAは4種類の塩基（A, U, C, G）からなる配列で、AとUが、そして、CとGが水素結合する（UとGも弱く結合する）ことで、ところどころステム構造を形成している。図61は、tRNA(transfer RNA)のステム構造で、5つのステムからなっている。tRNAはタンパク質合成の際に、アミノ酸をを運ぶ役割を担うRNA断片で、ひとつの細胞内に数十種があるが、どれもほぼ同様なクローバー型（Extra Armは失われていることも多い）をしている。3次元の立体構造は、ステム同士がさらに高次の相互作用をすることで、もっとコンパクトな形状となっている。

RNAの配列1本だけから、ステム構造を予測する試みは古くからなされており、配列断片の結合力の強弱を計算する手法が主流である[126, 127, 128]。既知の知識を制約に表現し、ステム構造予測を制約充足問題として解く方法[129]も提案されている。たくさんの配列からステム構造予測をする研究には、ローカルマッチングをするもの[130]と、文脈自由文法を用いるもの[131]がある。最近の研究には、ステム構造を与えてRNA配列のアライ

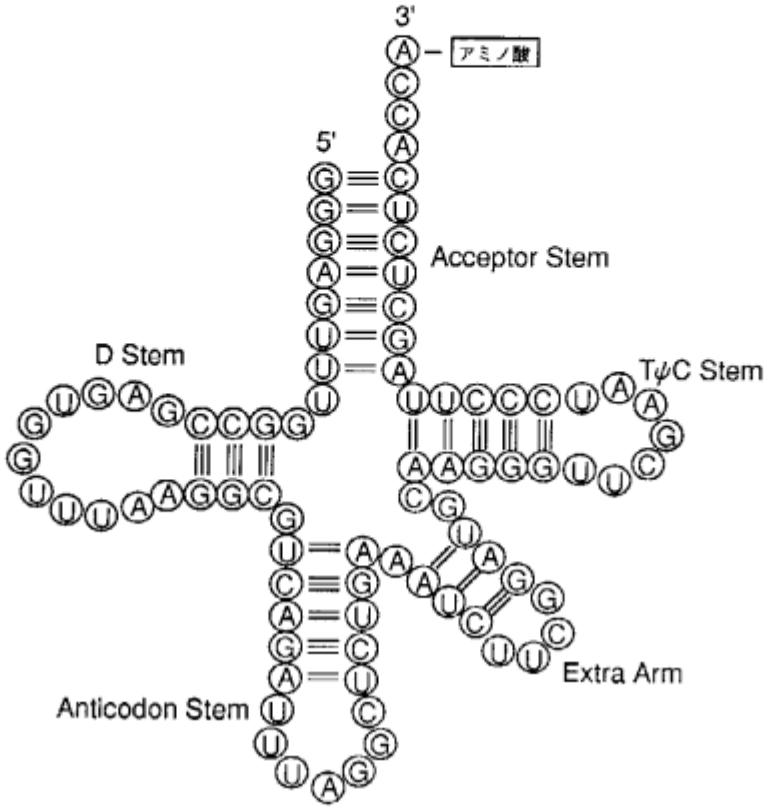


図 61: tRNA のステム構造

メントを向上させるものも出てきている [132]。我々のシステムは、ステム構造を与えることなしに高品質の RNA 配列のアライメントを実現し、結果的にステム構造の推測をするものである。

6.2 ステム構造を考慮したアライメント法

我々のステム構造を考慮したアライメントシステムは、RNA 配列のペアリングを評価する複雑なスコアの最適化を、並列シミュレーテッドアニーリングを用いて行う。しかし、通常シミュレーテッドアニーリング (SA) には、膨大な計算時間がかかるので、前処理を行う。ステム構造を考慮しないアライメントスコアを、並列反復改善法で最適化し、暫定的なアライメントを生成して SA の初期状態とする（図 62）。

このように本システムは、暫定的なアライメントを生成するモジュールと、ステム構造を考慮した、アライメント精緻化モジュールとで構成される。次にそれぞれのモジュールについて、詳しく述べる。

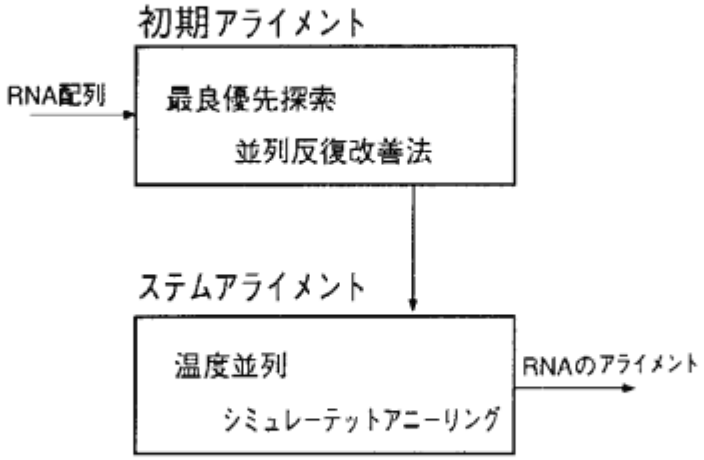


図 62: ステム構造を考慮したアライメントシステムの構成

6.2.1 暫定的アライメントの生成

暫定的アライメントには、ワークベンチで使用したものと同様な、反復改善法の最良優先探索並列、1本抜き限定分割を用いた。後のモジュールで精緻化するので、高速なアルゴリズムを選んだ結果である。

最適化すべきアライメントのスコアには、これまで通り SP 体系を用いた。この段階では、ステム構造は考慮されていない。塩基のマッチングの場合、Dayhoff マトリックスのような指標がないので、さまざまなパラメータの組合せの中から、妥当なアライメントを与えるマッチングスコアを選んだ。その算出式は、以下の通り。

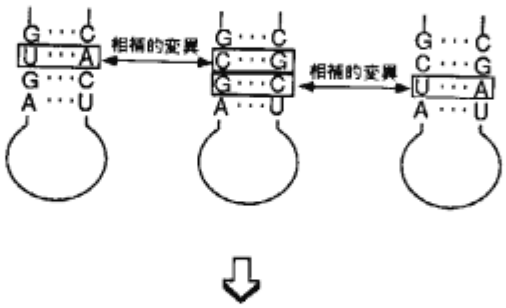
$$S = \sum_{i<j}^{seq.pair} \sum_{k}^{column} MatchScore(B_{ik}, B_{jk})$$

$$MatchScore(B_{ik}, B_{jk}) = \begin{cases} 4 & B \text{ がともに同じ塩基} \\ -2 & B \text{ が違う塩基} \\ 0 & B \text{ がともにギャップ} \\ 0 & B \text{ の一方がアウトギャップ} \\ -7 & B \text{ が塩基と opening gap} \\ -1 & B \text{ が塩基と extending gap} \end{cases}$$

6.2.2 ステム構造のアライメント

ステム構造を考慮したアライメントには、第2章で述べた温度並列 SA を用いた。ステム構造を反映した複雑な評価値の最適化を行うのに、妥当な手法である。

(a) RNAのステム構造



(b) RNAのアライメント

図 63: 相補的変異を利用したアライメント

図63に示したように、RNA のステム部分に対応する塩基には、相補的突然変異が見られる。これは、ステムで結合している塩基のペアの一方に突然変異が起こったならば、相手側の塩基にも、それを補うかたちで突然変異が起きるといったものである。同じステム構造を持つであろう複数の RNA 配列をアライメントする場合、この相補的突然変異を手がかりにすると、品質の良いアライメントができる。これを考慮し、最適化すべきエネルギー値は、以下の通りとした。

$$Energy = -\sum_{k+4<l}^{column} positive[Stem(k,l) + Stem(k+1,l-1) + Stem(k-1,l+1)]$$

$$Stem(k,l) = \sum_{i}^{seq} PairMatch(ik,il) + CovarianceMatch(k,l)$$

$$PairMatch(B_{ik}, B_{il}) = \begin{cases} 1 & B \text{ は A と U、または、G と C} \\ 0 & B \text{ は G と U} \\ -2 & \text{その他すべて} \end{cases}$$

$$\begin{aligned} CovarianceMatch(k,l) \;=\; & \#AU(k,l) + \#UA(k,l) + \#GC(k,l) + \#CG(k,l) \\ & -max[\#AU(k,l), \#UA(k,l), \#GC(k,l), \#CG(k,l)] \end{aligned}$$

(例、$\#AU(k,l)$: k カラムに A があり、l カラムに U がある配列の個数)

$Stem(k,l)$ では、k カラムと l カラムがステムを成している可能性を、単なる塩基ペアのマッチングと、相補的変異の度合から推測している。Energy では、3つ以上連続したカラム対について良い推測が得られる場合に限り、足し合わせる操作をして、全体のステム部分の量を評価している。なお、第2章と同様に、Energy は負の方向が良い値となっている。

このエネルギー値には、ステム領域以外の部分のマッチングスコアは、陽には含まれていないので、SA を適用した結果、ステム領域以外の部分が不満足なアライメントとなってしまうことがある。その場合、その部分のみを前のモジュールに戻し、反復改善法で再アライメントする必要がある。

6.3 実験と結果

前節で述べたシステムを、並列推論マシン PIM/p[133] 上に実装し、ステム構造の判明している tRNA の配列データに対してテストした。温度並列SA の部分に関しては、温度の数を変えて比較実験した。

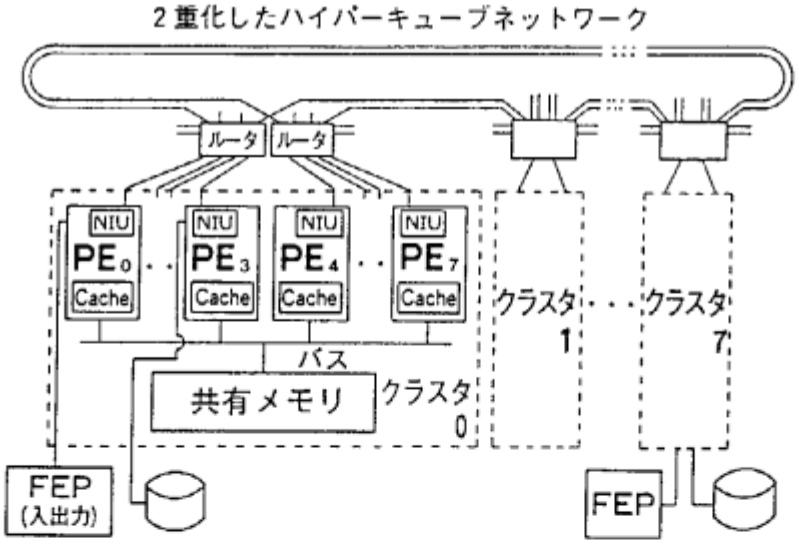


図 64: 並列推論マシン PIM/p の構成

6.3.1 並列計算機への実装

この実験で使用した PIM/p （図 64）は、PIM/m と同様に KL1 で動作する並列計算機であるが、要素プロセッサ (PE) の結合の仕方が異なっている。PIM/m の場合はメッシュ結合であったが、PIM/p の場合は、2重化ネットワークである。さらに、PIM/p の場合

は、PEが8つずつメモリを共有するクラスタを形成しており、クラスタ内では、忙しいPEで処理待ちしているプロセスが、自動的に他の空きPEに割り付けられる。いわゆる自動負荷分散がなされる。

プログラムは、これまでに使用したものが、ほとんどそのまま使える。暫定的アライメントを行う並列反復改善法は、第3章で使用したもののうち、アミノ酸用のマッチングスコアを塩基用のマッチングスコアに変えるだけであった。ステム構造を考慮した温度並列SAは、第2章で使用したプログラムのうち、評価値を配列ペアでなくて、カラムペアで計算する構造に書き換えるだけであった。

6.3.2 tRNAのステム構造の同定

Genbankから、配列が比較的多様である、ロイシン(leucine)を輸送するtRNAを検索し、テスト問題群を作った。図65の左上のウィンドウには、代表的な問題に対して暫定的アライメントを行った結果が示してある。並列反復改善法を用いて10分弱で求まった。22本のうち、上から8本はミトコンドリアで使われるtRNAであり、残りは核で使われるtRNAである。

我々は、アライメント結果からステム領域を抽出するツールも合わせて作成した。図65の右下のウィンドウでは、アライメントのなかの、どのカラムとどのカラムがステムを形成しやすいかを、円状の表現で示している。円の縁に表示されている数字は、アライメントのカラム番号を意味し、円内の直線は、該当カラム同士がステムを形成しやすいことを示唆する。表示にはパラメータがあり、図では、60%以上の配列について塩基がペアを成しており、かつ、3つ以上連続しているカラム対がその条件を満たしているものだけを表示させている。AとUが60%以上ペアを成しているカラム対を赤い直線、CとGのカラム対を紫の直線、両者が混在しているカラム対を緑の直線で表している。

円内を見ると、平行な直線群が5つ見られ、それぞれがステム領域である可能性を示している。前に述べたように、本当のステム領域には相補的突然変異が頻繁に起きているので、緑の直線群の方が正しいステムに対応している可能性が高い。ウィンドウの右側には、直線群を示すカラム番号のペアリストがあり、そこで指定したカラムが、アライメントのウィンドウに反映される。図では、緑の直線群に対応しているペアリストを指定し、アライメント内の該当カラムでペアを成している文字が、白く色が変わったところを示している。緑の直線群は、まさに、図61でいう、Acceptor Stem, D Stem, Anticodon Stemに対応している。(図61は、実はアライメントのうちの、最後の配列に対応する構造である。)

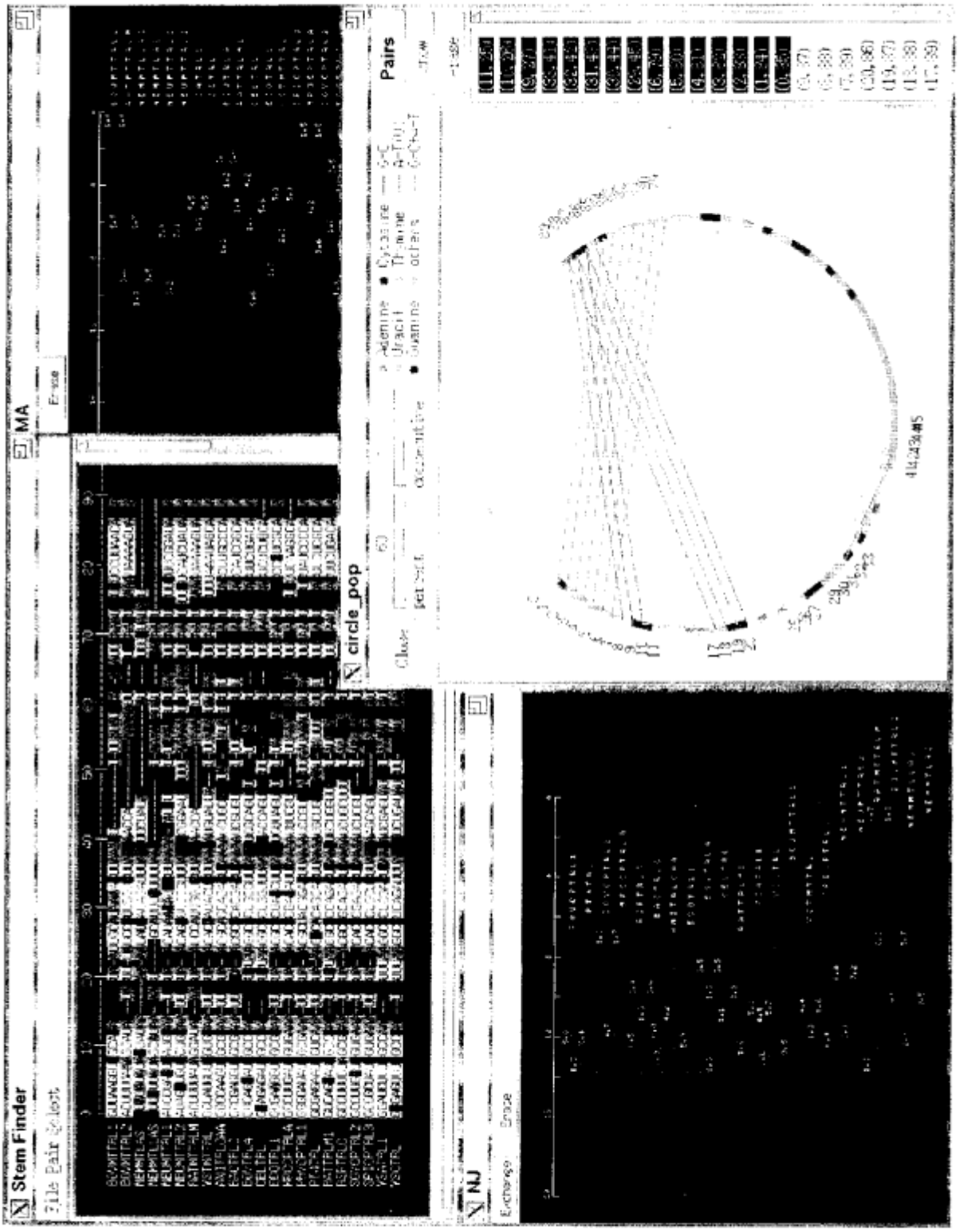


図 65: 暫定的アライメントからのステム抽出

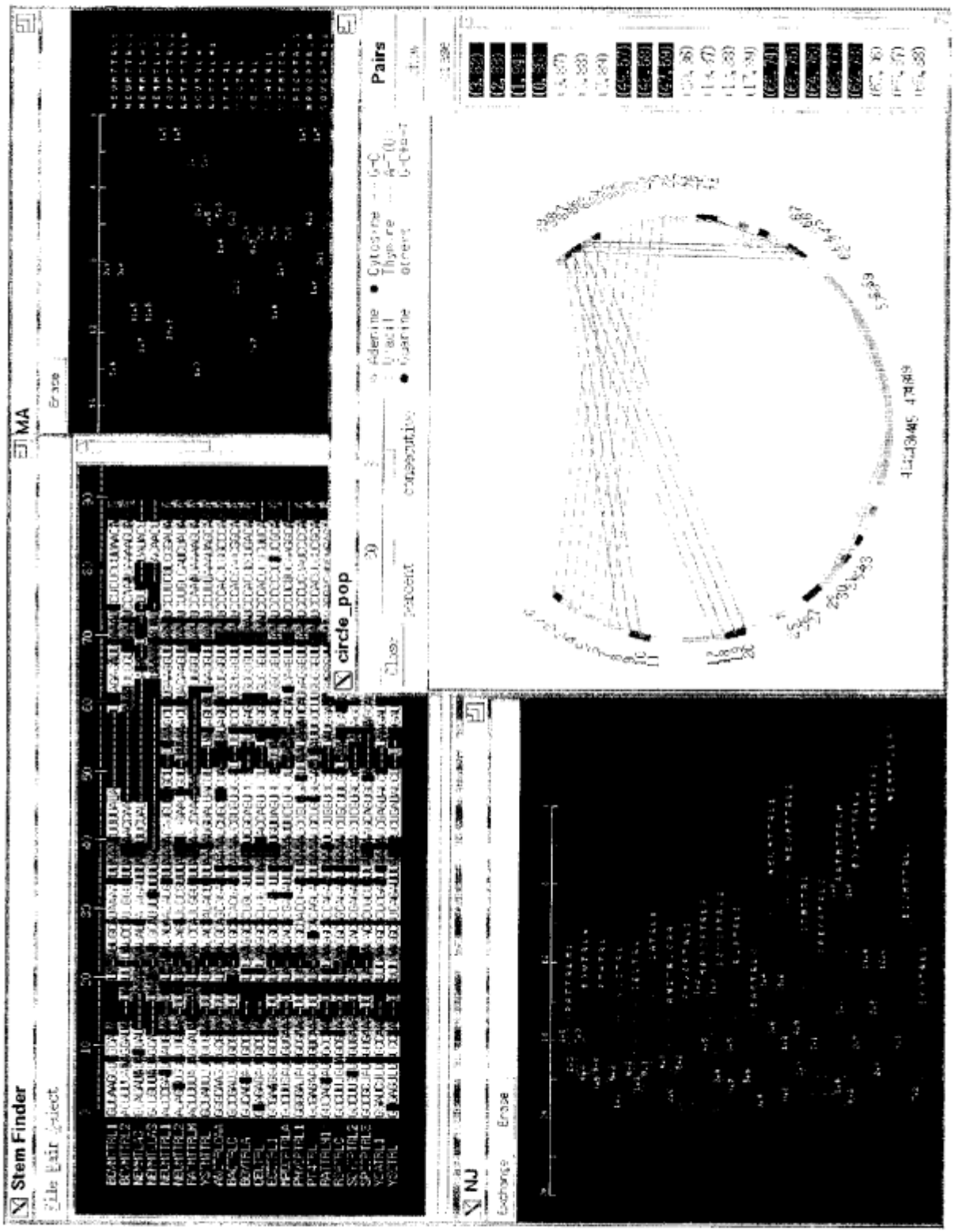


図 66: 最終アライメントからのステム抽出

暫定アライメントからは、tRNA が持つ５つのステムのうち、２つのステムは抽出できなかったが、アライメントの精緻化を行うとそれが可能である。図 66の左上のウィンドウには、図 65のアライメントを初期状態として、温度並列 SA で 12 時間アニーリングした結果が示されている。右下の円内を見ると、こんどは平行な直線群が８つ現れている。前と同様に緑の直線群だけを指定し、アライメントに反映させると、それらは、首尾良く tRNA の５つのステムに対応していることがわかる。

ステム領域の抽出ツールは、前章で述べたワークベンチに組み込んであり、ワークベンチが持つ他の機能も連動して使える。図 66の左下には、NJ 法で書いた系統樹が表示されているが、ミトコンドリアの tRNA が、５本と３本の系統に分かれている。一方、図 65の NJ 法の系統樹では、５本と２本と１本に分離しており、同系統と思われる配列が分散してしまった奇妙な系統樹であった。系統樹の観点からも、図 66のアライメントは正しいものに近付いていると推測できる。

このように、tRNA という、RNA のなかでは比較的小さな問題例ではあったが、良く知られているステム構造が、本システムで得られたアライメントから同定できた。

6.3.3 温度の個数を変えた実行比較

ここでは、並列計算の観点からの実験を報告する。図 65のアライメントを初期状態とし、温度の数を変えたいろいろな温度並列 SA を実行し、比較した。実験に使用した PIM/p はクラスタが８個であるが、１クラスタをモニタ用にとっておき、残りの７クラスタに、それぞれ１～８個の温度を割り当てた。クラスタには８つの PE があるので、８個の温度を割り当てた時に、１ PE 当り１つの解のアニーリングを行うこととなり、８個未満の場合は、複数の PE で１つの解のアニーリングを、協力して行うことになる。

図 67は、温度を 7 ～ 56 個に変えて 7.5 時間並列 SA を行った時の、最良解のエネルギーを示している。温度の値は、中高温とみられる 50 度と、最低温の 0.1 度の間を、必要な個数だけ等比的に分けて求めた。負荷分散が十分効率良くなされているとすると、温度が少ないと、温度当りの微小変形回数が多く、逆に、温度が多いと、微小変形回数が少なくなる。つまり、同一資源を用いた時の、少数点探索的アニーリングと、多点探索的アニーリングとのトレードオフ点を見つけることに相当する。

グラフを見ると、この時間内では、温度が 28 個の場合が最も良い解を与えたことがわかる。あまり温度数が多いと、時間当りの微小変形回数が少なくなるうえ、高温から低温へ良い解が降りてくるのに、いくつもの温度を経なければならず時間を要する。一方、温度数が

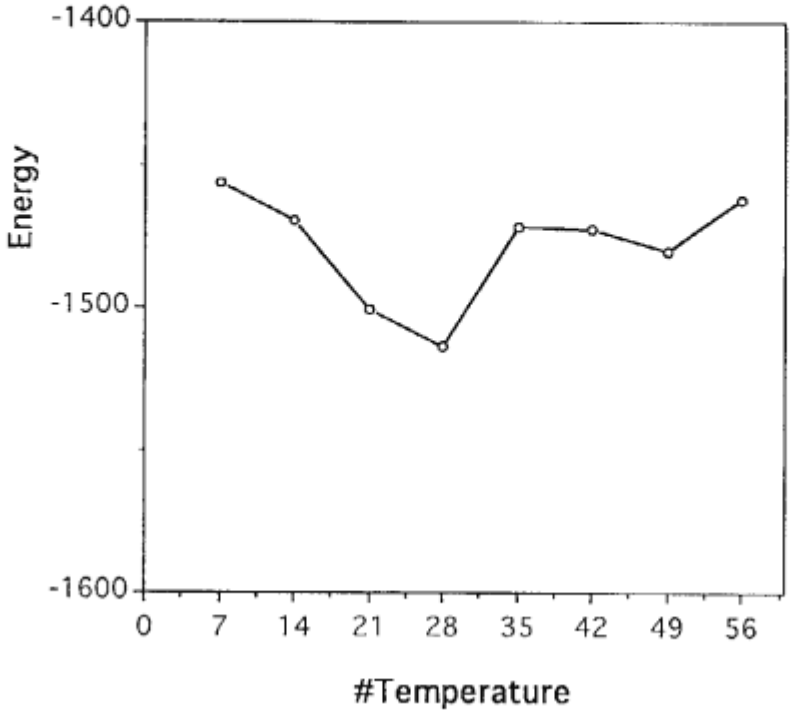


図 67: 温度個数による並列SA の比較

少ないと、荒っぽいアニーリングとなり、局所解に陥り易くなる。7.5 時間の実行で、温度 28 個あたりにトレードオフがあるのもうなずける。もっと長時間実行すると、温度数の多い方が有利になるだろうが、計算機の長時間占有に制限があり、実験ができなかった。

6.4 考察

我々は、RNA のステム（茎）構造を考慮しながらマルチプルアライメントを行うシステムを、並列反復改善法と並列シミュレーテッドアニーリング法とを適用して構築した。このシステムでは、RNA にみられる相補的突然変異を考慮に入れた、ステム領域を評価するスコアを最適化する。本システムにより、tRNA のクローバー型のステム構造を同定できるアライメントが生成できた。本節では、この結果をふまえて、アライメントと立体構造との関係、タンパク質への応用を検討する。最後に、今後の課題を述べる。

6.4.1 アライメントが先か、立体構造予測が先か

1.2.2 では、マルチプルアライメントと系統樹が、鶏と卵の関係になっていると述べたが、マルチプルアライメントと立体構造も、同様の関係になっている。すなわち、アライメント

の結果から立体構造予測が精度良く行える一方で、立体構造が正しく予測されているとアライメントの結果も向上する。理想的には、双方を同時に行えると良い。RNA のステム構造とそのアライメントでは、それらを同時に行う理論的な検討[134]がなされているが、やはり、n 次元 DP 以上の膨大な計算量が必要となる。実用的には、どちらかを先に、あるいは、交互に行わざるを得ない。

今回のシステムは、どちらを先に行ったものと考えられるだろうか。SA を用いたアライメントの処理を見ると、アライメントと構造予測を同時に行っているシステムと言えなくもない。しかし、その処理が要する計算量は大きく、最終解にかなり近い暫定的アライメントから出発しなければ、現実的時間内に処理ができない（これは第２章の考察から明らかである）。すなわち、本システムでは、暫定的アライメントが重要であることから、アライメントを先に行うシステムと考えられる。

では、ステム構造予測を先に行うシステムが考えられないだろうか。我々は、今回開発した、相補的突然変異を考慮に入れたステム領域の評価スコアを利用し、次のような、ステム構造予測を先に行うシステムを立案した。

1. 与えられた RNA 配列を１本ずつ、従来法の１つでステム構造予測する。この際に、予測結果を１つだけでなく、第２、第３候補と、複数生成する。

2. RNA 配列に第１候補のステム予測結果を付加し、反復改善法でアライメントする。この際に、通常の塩基マッチングスコアだけでなく、ステム構造部分のマッチングスコアも導入して、合わせて最適化する。

3. 完成したマルチプルアライメントを、ステム領域の評価スコアで評価する。この際に、各配列が、この評価スコアに寄与する度合を調べておく。

4. 最も寄与の少ない配列（予測が誤っている可能性が高い）について、構造予測結果を第２候補に入れ替え、再びアライメントする。

5. これを予測候補がなくなるまで繰り返し、求まったなかで、ステム領域の評価スコアが最も良いアライメントを最終解とする。

このシステムについて予備的な実験を行ったところ、アライメントを先に行う、SA を用いたシステムよりも、効率が良さそうな感触を得ている。

6.4.2 タンパク質立体構造への応用

RNA のステム構造を考慮したアライメントの枠組を、タンパク質へ応用する場合の問題点を考える。この枠組のキーポイントは、相補的突然変異の考慮であったが、タンパク質のアミノ酸の突然変異でも、同様なことがないだろうか。タンパク質で、ストランド同士がシートを形成するところは、主鎖の相互作用であり、アミノ酸の差異による影響は低い。ロイシンジッパーやシステイン結合は、それぞれ、ロイシンやシステインでないと起きないので、相補的突然変異は考えにくい。

現在のところ、タンパク質におけるアミノ酸の相補的突然変異は、確立された概念としては存在しないようである。しかし、最近盛んになっている、タンパク質のパッキング研究（たとえば[135]）には、その端緒が見い出せる。東大の河野氏の研究（私信）によると、パッキングしているコア部分のアミノ酸配列に突然変異が起きた場合に、空間的に隣接しているアミノ酸が、体積の点でパッキングを保持するように突然変異している例が、よく見つかるそうである。

また、図58で代表される膜タンパク質の構造安定性が、極性の相互作用で維持されている[136]とすれば、極性が反転しているような相補的突然変異が見つかっても不思議ではない。いずれにせよ、アミノ酸においても相補的突然変異がないだろうかという観点は、今後の興味深い研究テーマを提供している。

6.4.3 今後の課題

本システムの実証評価の面で、今後の一番の課題は、大きなアライメント問題でのテストである。RNA には、数百塩基、なかには千塩基を越える長いものがある。それらについても、本システムにより、ステム構造を的確に同定したアライメントが得られることが期待される。しかし、長い RNA に関しては、正しいステム構造がほとんど判明していない。そのため、アライメントのテスト用に、共通なステム構造を持ち、かつ、多様で長い配列データを集めるのが、現時点ではかなり難しい。今後の RNA に関する生物学的研究の進展を待たねばならない。

テストデータが準備できない点を除いても、別な問題がある。本システムの現状では、SA 処理に時間がかかり過ぎ、大きなアライメント問題の解決は困難である。2.4.3 で述べたような、SA 処理の効率化を図る必要がある。あるいは、6.4.1 で議論したような、SA を使わない他の方法を考案する必要があるだろう。

7 結論

本章では、本論文による研究成果の要約と、その成果を踏まえた将来への展望とを述べ、本論文のまとめとする。最後に、研究成果の公表方法、謝辞を述べる。

7.1 研究成果の要約

本論文は、タンパク質配列の典型的な解析課題であるマルチプルアライメントの問題に対し、実用規模の高品質な処理結果を与える技術についての、研究報告であった。従来の計算機によるアライメント問題の解法は、得られる解の精度が十分でなく、難しいところは、熟練した生物学者が手作業で取り組んでいた。我々は、並列計算機を応用し、高品質な解を実用的な時間内に提供できるシステムの開発に成功した。これにより、生物学者が行う手作業が大幅に軽減できた。システム開発のポイントは、並列反復改善法における限定分割手法の考案であった。また、遺伝的アルゴリズムにおける独自のマルチ個体群方式を開発し、効率の良い並列探索を実現した。

並列最適化手法	実行時間	評価スコア	解の一定性	最適化性能
シミュレーテッド アニーリング	長大にかかる	何でも可能	乱数に依存	収束性が悪い
最良優先反復改善法	実用的な範囲	DP可能なもの	いつも一定	局所解に陥る
マルチ個体群 遺伝的アルゴリズム	実用的な範囲	DP可能なもの	乱数に依存	性能が高い

図 68: 並列最適化手法の特徴比較

第2～4章では、並列シミュレーテッドアニーリング、並列反復改善法、並列遺伝的アルゴリズムの3つの最適化手法を、それぞれマルチプルアライメントの問題に適用した。図68には、それらの特徴の比較を示した。第2章の並列シミュレーテッドアニーリングのシステムは、複雑な評価スコア体系に対応できるものの、膨大な実行時間がかかり、最適解への収束速度が非常に遅い。第3章に記述した、最良優先探索を用いた並列反復改善法システムは、ダイナミックプログラミング (DP) をもとにした部分的最適化を繰り返すので、DP 可能な評価スコア体系ならば、高速に解を得られる。第4章の遺伝的アルゴリズムのシステム

は、マルチ個体群の方式を並列実装しており、最良優先探索で発生する局所最適解に陥る問題点を緩和している。ただ、乱数を使うので、一定した解がつねに得られるわけではないのが難点である。

第5章では、ユーザーのインタラクティブな操作でアライメントの問題を解く、ワークベンチについて報告した。現在のアライメントのスコア体系は、それを最適化しても、必ずしも生物学的に意味のあるアライメントになるとは限らない。そのため、現在では、あるスコア体系における最適化のほかに、こうしたツールの開発も必要となる。本ワークベンチでは、画面に表示されているアライメント途中の配列を指定し、制約を課したうえでの、再アライメントが可能である。アライメントには、乱数に依存せずに高速な解が得られる、最良優先探索の並列反復改善法を用いた。この方法は、局所解に陥る危険性をもつが、局所解であってもユーザーに有益な情報をもたらすことが多いので、解の一定性と速度を重視して採用した。将来的に、アライメントの良さが完全に盛り込まれた評価が、DP可能なスコア体系として決定されたならば、最も最適化性能が高い、遺伝的アルゴリズムの方式を採用するのが良い。

第6章では、RNAのステム（茎）構造を考慮しながらアライメント問題を解決するシステムについて報告した。本システムは、ステム構造を評価する複雑なスコア体系を、並列シミュレーテッドアニーリングの方式で最適化した。アニーリングの膨大な実行時間は、暫定的なアライメントから始めることで、ある程度低減できた。暫定的なアライメントの作成には、やはり速度を重視して、最良優先探索の並列反復改善法を採用した。

以下では、生物学的側面と計算機工学的側面とに分けて、研究成果の要約を記述する。

7.1.1 生物学的側面

従来の計算機によるマルチプルアライメントは、ダイナミックプログラミングで求めたペアワイズアライメントをツリー状に組み合わせていたため、得られる解の品質が十分でなかった。そのため、アライメントを行いたい生物学者は、多くの部分を手作業に頼らざるを得ない状態であった。

本研究では、高品質なアライメント解を生成できるシステムの開発を目指し、まず、並列シミュレーテッドアニーリングを応用した。その結果、小さなアライメント問題に限れば、従来手法を上回る解を得た（第2章）。次に、処理の高速化を図り、独自の限定分割法を搭載した並列反復改善法を考案した。それにより、実用規模の問題にまで、従来手法を上回る解が得られるシステムが開発できた（第3章）。さらに、遺伝的アルゴリズムを適用し、並

列反復改善法がもつ局所解に陥る問題点を緩和した（第４章）。

また我々は、生物学者の利用のため、並列反復改善法を搭載したアライメントのワークベンチを開発した。このワークベンチは、高品質な自動アライメントを提供すると同時に、制約に基づく、ユーザーのインタラクティブな操作が可能になっている。そのため、ユーザーが持つ事前知識の盛り込みや、配列の特徴を捉えながらのアライメント操作が実現でき、生物学の研究に有用なツールとなっている（第５章）。最後にRNAのステム構造を考慮したアライメントを、相補的突然変異を評価するスコア体系を開発することで可能とした（第６章）。

本研究により、アライメントに熟練していない生物学者でも、生物学的な知見を反映したアライメントが手軽に行える計算機環境が、確立された。

7.1.2 計算機工学的側面

本研究では、マルチプルアライメントの問題を組合せ最適化問題と捉え、並列計算機による解決を図った。始めに、並列シミュレーテッドアニーリングを適用し、アライメントのギャップ移動を微小変形に対応させることで、問題が解決できた。また、ギャップのブロック移動などの効率的な変形操作を導入することで、シミュレーテッドアニーリングの実行時間を大幅に短縮できた。それでも、最適化に必要な変形操作数は多く、小規模な問題への適用に限られた（第２章）。

実用規模のマルチプルアライメントの解決には、ギャップ移動のようなミクロな変形操作でなく、もっとマクロな変形操作を基にする必要があると認識し、ダイナミックプログラミングを配列群間のアライメントの最適化に利用する、反復改善法に着目した。反復改善法に限定分割手法のアイデアを導入し、最良優先探索並列とマルチ山登り並列を開発、実装したところ、実用規模の問題解決に成功した。最良優先探索並列を用いると、高速に準最適解が、マルチ山登り並列を用いると、少し時間がかかるが、より良い準最適解が得られる（第３章）。

最良優先探索は、比較的悪い局所最適解に陥る可能性があった。そこで、マルチ山登りのように局所最適解をなるべく避け、かつ、最良優先探索の収束性を保つことを目標に、並列遺伝的アルゴリズムの適用を試みた。遺伝的アルゴリズムのマルチ個体群の枠組のなかで、最良優先探索とマルチ山登りの、それぞれの特徴を生かした、独自の方式が実現でき、効果的な並列探索に成功した（第４章）。

アライメントのワークベンチでは、制約つきのマルチプルアライメントを実装したが、制

約によって分離される処理を並列に実行して高速化を行った（第5章）。ステム構造を考慮したアライメントでは、ダイナミックプログラミングでは扱えない複雑なスコア体系を、並列シミュレーテッドアニーリングで最適化することができた。この際、並列反復改善法で求めた暫定的な解を初期状態とし、シミュレーテッドアニーリングを実行することで、実行時間を低減できた（第6章）。

本研究により得られた最も基本的教訓は、次の通り。最適化手法を実用規模の大きな問題に適用する時には、問題の性質をよく調べ、最適解に至る効率の良い、なるべくマクロな変形操作を見つけることである。

7.2 将来への展望

本研究の成果をもとにして、将来行うべき研究の展望を、これも生物学的側面と計算機工学的側面とで分けて、それぞれ述べる。

7.2.1 生物学的側面

我々が開発したアライメントのワークベンチは、国内外の十数か所の研究機関で使われているが、ユーザーである生物学者から寄せられる意見や要望から、将来の研究課題が抽出できる。一番多い要望は、アライメント時の立体構造の考慮であった。本論文でも、第6章でRNAのケースを試みたが、要望の多いタンパク質の構造の考慮は、第6章でも議論したように、まだまだ解決すべき問題が山積している。

ときどき寄せられる課題に、ローカルマッチングがある。アライメントしたい配列のなかに部分的にしか類似性がないことが、しばしばあるようである。ローカルマッチングと、本研究が扱ったグローバルマッチングとを、使い分けてマルチプルアライメントできると、一層良いワークベンチとなろう。また、熟練した生物学者が良いアライメントだと見なすのには、ギャップの入り方などに一定のパターンがありそうである。そうした規則を、経験的変形操作として表現し、計算機へ盛り込むのも、実用的で面白い課題である。

最近では、PCR法の利用技術の進歩から、cDNA（mRNAから逆転写したDNA）解析が盛んに行われるようになった。つまり、タンパク質に翻訳される部分のDNA配列が次々に集まって来ている。それに伴って、DNA配列と、それに対応するタンパク質配列とを相互参照的にマルチプルアライメントできる環境の需要が高まってきた。配列の類似性が低いと、DNA配列のレベルより、タンパク質配列レベルでアライメントした方が効果的である。その一方で、DNA配列が、読みとりミスで1塩基欠けていていたりすると、対応する

タンパク質配列は全く誤りになってしまっているので、やはり、DNA 配列のアライメントも必要なのである。こうした、DNA 配列と、対応するタンパク質配列を同時平行的にアライメントするシステムの開発は、興味深いテーマである。

7.2.2 計算機工学的側面

最適化問題の解法のうち、遺伝的アルゴリズムは、変異、交配、淘汰を工夫でき、シミュレーテッドアニーリングに比べて柔軟で使いやすい枠組であった。将来、アライメントの専門家の経験的変形操作を盛り込む時にも、定向進化 [6] といったかたちで、遺伝的アルゴリズムの枠組のなかで対処するのが良いように思う。また、遺伝的アルゴリズムのマルチ個体群方式は、多様な探索方式を実現できるうえ、並列計算機と相性がいいので、アライメントの問題に限らずいろいろな問題に適用を進めたい。

最近は、隠れマルコフモデルを始めとした、確率的文法表現技術が進んでおり、アライメントの問題にも適用されてきている [75, 76, 77, 131]。こうした技術の導入や、詳細な比較検討は、将来に残された課題である。

7.3 研究成果の公表

本論文の研究成果は、研究論文および無償公開ソフトウェアのかたちで公表されている。

7.3.1 研究論文として

各章の内容と公表研究論文との関係は以下の通り。

第１章の内容

石川幹人，金久實：配列データの多重アラインメント法，新生化学実験講座第１６巻，日本生化学会編，東京化学同人，pp.323-342 (1993).

第２章の内容

Ishikawa,M., Toya,T., Hoshida,M., Nitta,K., Ogiwara,A. and Kanehisa,M., "Multiple sequence alignment by parallel simulated annealing," *Computer Applications in the Biosciences*, Vol.9, No.3, pp.267-273 (1993).

第３章の前半の内容

Ishikawa,M., Hoshida,M., Hirosawa,M., Toya,T., Onizuka,K. and Nitta,K., "Protein

Sequence Analysis by Parallel Inference Machine," *Proceedings of International Conference on Fifth Generation Computer Systems '92,* Vol.1, pp.294-299 (1992).

第3章の後半の内容

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).

第4章の前半の内容

Ishikawa,M., Toya,T., Totoki,Y. and Konagaya,A., "Parallel Iterative Aligner with Genetic Algorithm," *Proceedings of AI and Genome Workshop in the 13th International Joint Conference on Artificial Intelligence,* pp.13-22 (1993).

第4章の後半の内容

Ishikawa,M., Toya,T. and Totoki,Y., "Parallel Application Systems in Genetic Information Processing," *Proceedings of International Symposium on Fifth Generation Computer Systems '94,* pp.129-138 (1994).

第5章の内容

Ishikawa,M., Totoki,Y., Tanaka,R. and Hirosawa,M., "Multiple Sequence Alignment Editor Featured by Constraint-based Parallel Iterative Aligner," *Proceedings of the 3rd International Conference on Bioinformatics and Genome Research* (近刊).

第6章の内容

Ishikawa,M., Toya,T., Totoki,Y. and Tanaka,R., "Multiple RNA-Sequence Alignment Considering Stem Regions," *Proceedings of Genome Informatics Workshop V,* Universal Academy Press, pp.110-119 (1994).

7.3.2 無償公開ソフトウェアとして

各章で開発したプログラムは、ICOT 無償公開ソフトウェア（IFS）に登録されており、Internet を介して ftp.icot.or.jp (192.26.9.33) にアクセスすることで誰でも入手できる。登録番号とこれまでのアクセス回数の実績は、以下の通り。

- 第2章のソフトウェア： IFS No.56 アクセス実績 67 回（1992 年 6 月公開）
- 第3章のソフトウェア： IFS No.75 アクセス実績 40 回（1993 年 6 月公開）

- 第4章のソフトウェア： IFS No.94 （1994年12月公開）
- 第5章のソフトウェア： IFS No.76 アクセス実績70回（1993年6月公開）
- 第6章のソフトウェア： IFS No.95 （1994年12月公開）

7.4 謝辞

本研究は、著者が（財）新世代コンピュータ技術開発機構 (ICOT) の遺伝子情報処理応用グループに在籍中に行ったものである。同グループのメンバーであった、星田昌紀氏、広沢誠氏、戸谷智之氏、十時泰氏、田中秀俊氏、田中令子氏、鬼塚健太郎氏、矢田哲士氏、赤星正幸氏、浅井潔氏には研究に協力していただいた。同じくICOTに在籍されていた木村宏一氏、市吉伸行氏には計算機科学の面からの議論に参加していただいた。ICOTが主催していた遺伝子情報処理ワーキンググループを通して、金久實氏、美宅成樹氏、小長谷明彦氏、高木利久氏、秋山泰氏、中井謙太氏、荻原淳氏、藤博幸氏、渡辺日出海氏、松田秀雄氏、内藤公敏氏、榊原康文氏、阿久津達也氏の方々には、とくにご助言を賜わった。生物学の利用者の立場から、京都大学宮田研究室の岩部直之氏、隈啓一氏からは、たいへん貴重なご意見をいただいた。古くからアライメントの研究を手掛けられている後藤修氏には、技術的な面をご指導いただいた。海外からは、Kaoru Yoshida氏、Richard Feldmann氏、George Michaels氏、Ann Barber氏、Ross Overbeek氏、Mike Lynch氏、John Conery氏に、ご意見をお寄せいただいた。また、本論文の執筆にあたって、北方まゆみ嬢、田沢ゆか嬢には図の作成をお手伝い願った。以上の諸氏に心から謝意を表する。最後に、研究の機会を作って下さった、ICOTの内田俊一所長、新田克己部長、松下電器産業（株）の山崎正人部長にお礼を申し上げて、本論文の結びとしたい。

参考文献

[1] Bernstein,F.C. *et al.*, "The Protein Data Bank: A Computer-based Archival File for Macromolecular Structures," *J. Mol. Biol.*, Vol.112, pp.535-542 (1977).

[2] Barker,W.C. *et al.*, "The PIR-International Protein Sequence Database," *Nucleic Acids Res.*, Vol.20 (Supplement), pp.2023-2026 (1992).

[3] 五條堀孝，森山悦子，内藤公敏，河合正人：大量ＤＮＡデータを対象とした遺伝情報のコンピュータ解析，情報処理 Vol.31, pp.878-886 (1990).

[4] Staden,R., "Methods to Define and Locate Patterns of Motifs in Sequences," *Comput. Applic. Biosci.*, Vol.4, pp.53-60 (1988).

[5] Rhodes,D. and Klug,A.：ジンクフィンガーによる遺伝子の発現制御，日経サイエンス，Vol.23, No.4, pp.96-106 (1993).

[6] 木村資生：分子進化の中立説，紀伊国屋書店 (1986).

[7] 星田昌紀，石川幹人：進化の謎を解く鍵－アミノ酸配列解析，**bit**，Vol.25，No.1，pp.51-60 (1993).

[8] Pearson,W.G. and Lipman,D.J., "Improved Tools for Biological Sequence Comparison," *Proc. Natl. Acad. Sci.*, Vol.85, pp.2444-2448 (1988).

[9] Mullis,K.B.：遺伝子を自動的に複製するPCR法の発見，日経サイエンス，Vol.20, No.6, pp.16-25 (1990).

[10] 宮田隆，藤博幸，林田秀宣：コンピューターによる逆転写酵素遺伝子の探査，日経サイエンス，Vol.16, No.2, pp.86-97 (1986).

[11] 根井正利：分子進化遺伝学，培風館 (1990).

[12] 石川幹人，金久實：配列データの多重アラインメント法，新生化学実験講座第１６巻，日本生化学会編，東京化学同人，pp.323-342 (1993).

[13] 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).

[14] Smith,T.F. and Waterman,M.F., "Identification of common molecular subsequences," *J. Mol. Biol.*, Vol.147, pp.195-197 (1981).

[15] Gotoh,O., "An improved algorithm for matching biological sequences," *J. Mol. Biol.*, Vol.162, pp.705-708 (1982).

[16] 後藤修：核酸・蛋白質一次構造の計算機による解析，日本物理学会誌，Vol.38, No.6, pp.477-480 (1983).

[17] 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).

[18] Gonnet,G.H., Cohen,M.A. and Benner,S.A., "Exhaustive Matching of the Entire Protein Sequence Database," *Science,* Vol.256, pp.1443-1445 (1992).

[19] Jones,D.T., Taylor,W.R. and Thornton,J.M., "The rapid generation of mutation data matrices from protein sequences," *Comput. Applic. Biosci.,* Vol.8, pp.275-282 (1992).

[20] Fitch,W.M. and Smith,T.F., "Optimal sequence alignments," *Proc. Natl. Acad. Sci.,* Vol.80, pp.1382-1386 (1983).

[21] 宮田隆，林田秀宣，菊野玲子，安永照雄：コンピューターによる遺伝子のホモロジー解析，続生化学実験講座第1巻遺伝子研究法Ⅰ，日本生化学会編，東京化学同人，pp.381-423 (1986).

[22] Goad,W.B. and Kanehisa,M.I., "Pattern recognition in nucleic acid sequences. I. A general method for finding local homologies and symmetries," *Nucleic Acids Res.,* Vol.10, pp.247-263 (1982).

[23] Murata,M., Richardson,J.S. and Sussman,J.L., "Simultaneous comparison of three protein sequences," *Proc. Natl. Acad. Sci.,* Vol.82, pp.3073-3077 (1985).

[24] Altschul,S.F., "Gap Costs for Multiple Sequence Alignment, *J. Theor. Biol.,* Vol.138, pp.297-309 (1989).

[25] Ukkonen,E., "Algorithms for Approximate String Matching," *Inf. Control,* Vol.64, pp.100-118 (1985).

[26] Carrillo,H. and Lipman,D., "The multiple sequence alignment problem in biology," *SIAM J. Appl. Math.* Vol.48, pp.1073-1082 (1988).

[27] Vingron,M. and Argos,P., "A fast and sensitive multiple sequence alignment algorithm," *Comput. Applic. Biosci.,* Vol.5, pp.115-121 (1989).

[28] Bacon,D.J. and Anderson,W.F., "Multiple Sequence Alignment," *J. Mol. Biol.*, Vol.191, pp.153-161 (1986).

[29] Alexandrov,N.N., "Local multiple alignment by consensus matrix," *Comput. Applic. Biosci.*, Vol.8, pp.339-345 (1992).

[30] Taylor,W.R., "Multiple sequence alignment by a pairwise algorithm," *Comput. Applic. Biosci.*, Vol.3, pp.81-87 (1987).

[31] Barton,J.G. and Sternberg,M.J.E., "A Strategy for Rapid Multiple Alignment of Protein Sequences," *J. Mol. Biol.*, Vol.198, pp.327-337 (1987).

[32] Gotoh,O., "Optimal Alignment between Groups of Sequences and its Application to Multiple Sequence Alignment," *Comput. Applic. Biosci.*, Vol.9, pp.361-370 (1993).

[33] Gribskov,M., McLachlan,A.D. and Eisenberg,D., "Profile analysis: Detection of distantly related proteins," *Proc. Natl. Acad. Sci.*, Vol.84, pp.4355-4358 (1987).

[34] Sneath,P.H.A and Sokal,R.R., "Numerical Taxonomy," Freeman and Company (1973).

[35] 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).

[36] 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).

[37] Altschul, S.F. and Lipman, D.J.: Trees, Stars, and Multiple Biological Sequence Alignment, *SIAM J. Appl. Math.*, Vol.49, pp.197-209 (1989).

[38] Hirosawa,M., Hoshida,M., Ishikawa,M. and Toya,T., "MASCOT: multiple alignment system for protein sequence based on three-way dynamic programming," *Comput. Applic. Biosci.*, Vol.9, pp.161-167 (1993).

[39] Sankoff,D., "Minimal mutation trees of sequences," *SIAM J. Appl. Math.*, Vol.28, pp.35-42 (1975).

[40] Hogeweg,P. and Hesper,B., "The Alignment of Sets of Sequences and the Construction of Phyletic Trees: An Integrated Method," *J. Mol. Evol.,* VVol.20, pp.175-186 (1984).

[41] Hein,J., "Unified Approach to Alignment and Phylogenies," *Methods in Enzymology,* Vol.183, Academic Press, pp.626-644 (1990).

[42] Corpet,F., "Multiple sequence alignment with hierarchical clustering," *Nucleic Acids Res.,* Vol.16, pp.10881-10890 (1988).

[43] Coulson,A.F.W., Collins,J.F. and Lyall,A., "Protein and nucleic acid sequence database searching: a suitable case for parallel processing," *Comput. J.,* Vol.30, pp.420-424 (1987).

[44] Miller,P.L., Nadkarni,P.M. and Carriero,N.M., "Parallel computation and FASTA: confronting the problem of parallel database search for a fast sequence comparison algorithm," *Comput. Applic. Biosci.,* Vol.7, pp.71-78 (1991).

[45] Sittig,D.F. *et al.*, "A Parallel Computing Approach to Genetic Sequence Comparison: The Master-Worker Paradigm with Interworker Communication," *Comput. Biomed. Res.,* Vol.24, pp.152-169 (1991).

[46] Kawai,M., Kishino,A. and Naito,K., "Rapid Analysis Methodology for Gene Sequences Using a Parallel Processor," *FUJITSU Scientific and Technical Journal,* Vol.27, pp.270-277 (1991).

[47] Istvanick,W. *et al.*, "Dynamic Methods for Fragment Assembly in Large Scale Genome Sequencing Projects," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.534-543 (1993).

[48] Iyengar,A.K., "Parallel DNA Sequence Analysis," *MIT/LCS/TR-428,* (1988).

[49] Araki,S. *et al.*, "Application of Parallelized DP and A* Algorithm to Multiple Sequence Alignment," *Proc. Genome Informatics Workshop IV,* Universal Academy Press, pp.94-102 (1993).

[50] 戸谷智之，星田昌紀，石川幹人，新田克己：並列3次元ダイナミックプログラミング法によるタンパクの配列解析，*Proc. KL1 Programming Workshop '91,* pp.83-92 (1991).

[51] Ukiyama,N. and Imai,H., "Parallel Multiple Alignments and Their Implementation on CM5," *Proc. Genome Informatics Workshop IV,* Universal Academy Press, pp.103-108 (1993).

[52] Butler,R. *et al.*, "Aligning Genetic Sequences," *Strand: New Concepts in Parallel Programming,* Prentice-Hall, New Jersey (1990).

[53] White,C.T. *et al.*, "BioSCAN: A VLSI-Based System for Biosequence Analysis," *Proc. IEEE Int'l. Conf. Comput. Design,* pp.504-509 (1991).

[54] Archambaud,D., Faudemay,P. and Greiner,A., "RAPID-2, An Object-Oriented Associative Memory Applicable to Genomic Data Processing," *Proc. 27th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.5, pp.150-159 (1994).

[55] Benson,D. Lipman,D.J. and Ostell,J., "Genbank," *Nucleic Acids Res.,* Vol.21, pp.2963-2965 (1993).

[56] 金久實，新田克己，小長谷明彦，田中秀俊：知識情報処理技術とヒトゲノム計画，人工知能学会誌, Vol.6, pp.630-640 (1991).

[57] Nitta,K. and Ishikawa,M., "AI Technologies for Molecular Biology Analysis," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.744-745 (1993).

[58] Ishikawa,M., Toya,T., Hoshida,M., Nitta,K., Ogiwara,A. and Kanehisa,M., "Multiple sequence alignment by parallel simulated annealing," *Comput. Applic. Biosci.,* Vol.9, pp.267-273 (1993).

[59] 石川幹人，金久實：文字列を比較し並べる，日本物理学会誌，Vol.48，pp.341-343 (1993).

[60] Metropolis,N. *et al.*, "Equation of state calculations by fast computing machines," *J. Chemical Physics,* Vol.21, pp.1087-1092 (1953).

[61] Kirkpatrick,S., Gelatt,C.D. and Vecci,M.P., "Optimization by Simulated Annealing," *Science,* vol.220, pp.671-680 (1983).

[62] Cuticchia,A.J., Arnold,J. and Timberlake,W.E., "The use of simulated annealing in chromosome reconstruction experiments based on binary scoring," *Genetics,* Vol.132, pp.591-601 (1992).

[63] Lukashin,A.V., Engelbrecht,J. and Brunak,S., "Multiple alignment using simulated annealing: branch point definition in human mRNA splicing," *Nucleic Acids Res.,* Vol.20, pp.2511-2516 (1992).

[64] Ohya,M., Miyazaki,S. and Ogata,K., "On Multiple Alignment of Genome Sequences," *IEICE Trans. Commun. E75-B,* pp.453-457 (1992).

[65] Bouzida,D., Kumar,S. and Swendsen,R.H., "A Simulated Annealing Approach for Probing Biomolecular Structures," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.736-742 (1993).

[66] Kimura,K. and Taki,K., "Time-homogeneous Parallel Annealing Algorithm," *Proc. Comput. Appl. Math. 13 (IMACS'91),* pp.827-828 (1991).

[67] Taki,K. "The Parallel Software Research and Development Tool: Multi-PSI System," *Programming of Future Generation Computers,* Elsevier Science Publishers (1988).

[68] Nakajima,K. *et al.*, "Distributed Implementation of KL1 on the Multi-PSI/V2," *Proc. 6th Int'l. Conf. Logic Programming,* MIT press, pp.436-451 (1989).

[69] Hirosawa,M., Hoshida,M. and Ishikawa,M., "Protein Multiple Sequence Alignment using Knowledge," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.803-812 (1993).

[70] Toya,T., Ishikawa,M., Hoshida,M. and Araki,H., "Parallel Simulated Annealing and its Application to Protein Sequence Matching," *Poster Summaries 5th Australian Joint Conf. Artifi. Intelli.,* pp.35-36 (1992).

[71] Hirosawa,M., Feldmann,R.J., Rawn,D., Ishikawa,M., Hoshida,M. and Michaels,G., "Folding Simulation using Temperature Parallel Simulated Annealing," *Proc. Fifth Generation Comput. Syst. '92,* pp.300-306 (1992).

[72] Ishikawa,M., Hoshida,M., Hirosawa,M., Toya,T., Onizuka,K. and Nitta,K., "Protein Sequence Analysis by Parallel Inference Machine," *Proc. Fifth Generation Comput. Syst. '92,* pp.294-299 (1992).

[73] 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).

[74] 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).

[75] Haussler,D., Krogh,A., Mian,I.S. and Sjoelander,K., "Protein Modeling using Hidden Markov Models: Analysis of Globins," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.792-802 (1993).

[76] Baldi,P., Chauvin,Y., Hunkapillar,T. and McClure,M., "Hidden markov models of biological primary sequence information," *Proc. Natl. Acad. Sci.,* Vol.91, pp.1059-1063 (1994).

[77] Tanaka,H., Asai,K., Ishikawa,M. and Konagaya,A., "Hidden Markov Models and Iterative Aligners: Study of their Equivalence and Possibilities," *Proc. 1st Int'l. Conf. Intelli. Syst. Mol. Biol.,* pp.395-401 (1993).

[78] Hanks,K.S., Quinn,A.M. and Hunter,T., "The Protein Kinase Family," *Science,* Vol.241, pp.42-52 (1988).

[79] Hirata,K. *et al.*, "Parallel and Distributed Implementation of Concurrent Logic Programming Language KL1," *Proc. Fifth Generation Comput. Syst. '92,* pp.436-459 (1992).

[80] Nakashima,H. *et al.*, "Architecture and Implementation of PIM/m," *Proc. Fifth Generation Comput. Syst. '92,* pp.425-435 (1992).

[81] Allison,L., Wallace,C.S. and Yee,C.N., "Minimum message length encoding, evolutionary trees and multiple-alignment," *Proc. 25th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.4, pp.663-674 (1992).

[82] Subbiah,S. and Harrison,S.C., "A Method for Multiple Sequence Alignment with Gaps," *J. Mol. Biol.,* Vol.209, pp.539-548 (1989).

[83] Hirosawa,M., Totoki,Y., Hoshida,M. and Ishikawa,M., "Comprehensive Study on Iterative Algorithms of Multiple Sequence Alignment," *Comput. Applic. Biosci.,* (印刷中) .

[84] Ishikawa,M., Toya,T., Totoki,Y. and Konagaya,A., "Parallel Iterative Aligner with Genetic Algorithm," *Proc. AI and Genome Workshop in 13th Int'l. Joint Conf. Artifi. Intelli.,* pp.13-22 (1993).

[85] Holland,J.H., "Adaptation in Natural and Artificial Systems," Univ. Michigan Press (1975).

[86] Goldberg,D.E., "Genetic Algorithms in Search, Optimization, and Machine Learning," Addison-Wesley (1989).

[87] Konagaya,A. and Kondou,H., "Stochastic Motif Extraction using a Genetic Algorithm with the MDL Principle," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.746-755 (1993).

[88] Parsons,R., Forrest,S. and Burks,C., "Genetic Algorithms for Sequence Assembly", *Proc. 1st Int'l. Conf. Intelli. Syst. Mol. Biol.,* pp.310-318 (1993).

[89] Platt,D. and Dix,T.I., "Construction of Restriction Maps Using a Genetic Algorithm," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.756-762 (1993).

[90] Unger,R. and Moult,J., "On the Applicability of Genetic Algorithms to Protein Folding," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.715-725 (1993).

[91] Muehlenbein,H. and Schlierkamp-Voosen,D., "Predictive Models for the Breeder Genetic Algorithm: Continuous Parameter Optimization," *Evolutionary Computation,* Vol.1, pp.25-49 (1993).

[92] 筒井茂義，藤本好司：個体群探索分岐型遺伝的アルゴリズム（Forking GA）の提案，人工知能学会誌, Vol.9, pp.741-747 (1994).

[93] Tajima,K., "Multiple Sequence Alignment Using Parallel Genetic Algorithms," *Proc. Genome Informatics Workshop IV,* Universal Academy Press, pp.183-187 (1993).

[94] Ishikawa,M., Totoki,Y., Tanaka,R. and Hirosawa,M., "Multiple Sequence Alignment Editor Featured by Constraint-based Parallel Iterative Aligner," *Proc. 3rd Int'l. Conf. Bioinformatics and Genome Research* (印刷中).

[95] 谷村隆次，梅山秀明：エキスパートシステムを用いたタンパク質の立体構造の推定と病気治療への応用，情報処理 Vol.31, pp.897-903 (1990).

[96] Smithers,T., Tang,M.X. and Tomes,N., "An Intelligent Drug Design Support," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.1, pp.634-645 (1993).

[97] Akahoshi,M., Onizuka,K., Ishikawa,M. and Asai,K., "A Three-Dimensional Animation System for Protein Folding Simulation," *Proc. 27th Annu. Hawaii Int'l. Conf. Syst. Sci.,* Vol.5, pp.173-182 (1994).

[98] Barber,A.M. and Maizel Jr,J.V., "SequenceEditingAligner: A multiple sequence editor and aligner," *Gene Anal. Tech.,* Vol.7, pp.39-45 (1990).

[99] Schuler,G.D., Altschul,S.F. and Lipman,D.J., "A Workbench for Multiple Alignment Construction and Analysis," *PROTEINS,* Vol.9, pp.180-190 (1991).

[100] 隈啓一：アライメントの最適化と自動化への問題点，日本生物物理学会誌，Vol.32, pp.62-64 (1992).

[101] Chikayama,T., Fujise,T. and Sekita,D., "A portable and efficient implementation of KL1," *Lecture Notes Comput. Sci.,* Vol.844, pp.25-39 (1994).

[102] Blanck,A. and Oesterhelt,D., "The halo-opsin gene. II. Sequence, primary structure of halorhodopsin and comparison with bacteriorhodopsin," *EMBO J.,* Vol.6, pp.265-273 (1987).

[103] Hargrave,P.A., "Seven-helix receptors," *Current Opinion Struc. Biol.*, Vol.1, pp.575-581 (1991).

[104] Bairoch,A., "PROSITE: a dictionary of sites and patterns in proteins," *Nucleic Acids Res.*, Vol.19, pp.2241-2245 (1992).

[105] Gilman,A.G., "G Proteins: Transducers of Receptor-Generated Signals," *Annu. Rev. Biochem.*, Vol.56, pp.615-649 (1987).

[106] Hirosawa,M., Tanaka,R. and Ishikawa,M., "Application of Deductive Object- Oriented Knowledge Base to Genetic Information Processing," *Proc. Int'l. Symp. Next Generation Database Syst. Applic.*, pp.116-122 (1993).

[107] Saitou,N., and Nei,M., "The neighbor-joining method: A new method for reconstructing phylogenetic trees," *Mol. Biol. Evol.*, Vol.4, pp.406-425 (1987).

[108] Nozaki,Y. and Tanfford,C., "The solubility of amino acids and two glycine peptides in aqueous ethanol and dioxane solutions," *J. Biol. Chem.*, Vol.246, pp.2211-2217 (1971).

[109] Janin,J., "Surface and inside volumes in globular proteins," *Nature*, Vol.277, pp.491-492 (1979).

[110] Kyte,J. and Doolittle,R.F., "A simple method for displaying the hydropathic character of a protein," *J. Mol. Biol.*, Vol.157, pp.105-132 (1982).

[111] Miyazawa,S. and Jernigan,R.L., "Estimation of effective interresidue contact energies from protein crystal structures: Quasi-chemical approximation," *Macromolecules*, Vol.18, pp.534-552 (1985).

[112] Chou,P.Y. and Fasman,G.D., "Empirical predictions of protein conformations," *Annu. Rev. Biochem.*, Vol.47, pp.251-276 (1978).

[113] Levitt,M., "Comformational preferences of amino acids in globular proteins," *Biochemistry*, Vol.17, pp.4277-4285 (1978).

[114] Ishikawa,M., Toya,T., Totoki,Y. and Tanaka,R., "Multiple RNA-Sequence Alignment Considering Stem Regions," *Proc. Genome Informatics Workshop V*, Universal Academy Press, pp.110-119 (1994).

[115] Henneke,C.M., "A multiple sequence alignment algorithm for homologous proteins using secondary structure information and optionally keying alignments to functionally important sites," *Comput. Applic. Biosci.*, Vol.5, pp.141-150 (1989).

[116] Smith,R.F. and Smith,T.F., "Pattern-induced multi-sequence alignment (PIMA) algorithm employing secondary structure-dependent gap penalties for use in comparative modelling," *Protein Engneering*, Vol.5, pp.35-41 (1992).

[117] 西川建：タンパク質の二次構造予測，情報処理 Vol.31, pp.887-896 (1990).

[118] Zvelebil,M.J., Barton,G.J., Taylor,W.R. and Sternberg,J.E., "Prediction of protein secondary structure and active sites using the alignment of homologous sequences," *J. Mol. Biol.*, Vol.195, pp.957-961 (1987).

[119] Rost,B., Sander,C. and Schneider,R., "Evolution and Neural Networks—Protein Secondary Structure Prediction Above 71% Accuracy," *Proc. 27th Annu. Hawaii Int'l. Conf. Syst. Sci.*, Vol.5, pp.385-394 (1994).

[120] Onizuka,K., Asai,K., Ishikawa,M. and Wong,S.T.C., "A multi-Level Description Scheme of Protein Conformation," *Proc. 1st Int'l. Conf. Intelli. Syst. Mol. Biol.*, pp.301-309 (1993).

[121] Onizuka,K., Tsuda,H., Ishikawa,M., Aiba,R., Asai,K. and Ito,K., "Protein Structure Prediction Based on Multi-Level Description," *Proc. 27th Annu. Hawaii Int'l. Conf. Syst. Sci.*, Vol.5, pp.355-364 (1994).

[122] Onizuka,K., Akahoshi,M., Ishikawa,M. and Asai,K., "The Multi-Scale 3D-1D Compatibility Scoring for Inverse Protein Folding Problem," *Proc. 2nd Int'l. Conf. Intelli. Syst. Mol. Biol.*, pp.314-321 (1994).

[123] Taylor,W.R. and Orengo,C.A., "Protein Structure Alignment," *J. Mol. Biol.*, Vol.208, pp.1-22 (1989).

[124] Akutsu,T., Onizuka,K. and Ishikawa,M., "Hashing techniques and its applications to protein structure databases," *Proc. 28th Annu. Hawaii Int'l. Conf. Syst. Sci.* (印刷中).

[125] Landshulz,W.H., Johnson,P.F. and McKnight,S.L., "The Leucine Zipper: a Hypothetical Structure Common to a New Class of DNA Binding Proteins," *Science,* Vol.240, pp.1759-1764 (1988).

[126] Pipas,J.M. and McMahon,J.E., "Method for Predicting RNA Secondary Structure," *Proc. Natl. Acad. Sci.,* Vol.72, pp.2017-2021 (1975).

[127] Zuker,M. and Stiegler,P., "Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information," *Nucleic Acids Res.,* Vol.9, pp.133-148 (1981).

[128] Yamamoto,K. and Yoshikura,H., "An improved algorithm for the prediction of optimum and suboptimum folding structures of long single-stranded RNA," *Comput. Applic. Biosci.,* Vol.3, pp.31-35 (1987).

[129] Major,F. *et al.,* "The Combination of Symbolic and Numerical Computation for Three-Dimensional Modeling of RNA," *Science,* Vol.253, pp.1255-1260 (1991).

[130] Waterman,M.S., "Mathematical Methods for DNA Sequences," CRC Press, pp.185-224 (1989).

[131] Sakakibara,Y. *et al.,* "Stochastic context-free grammars in computational biology: application to modeling RNA," *Proc. Genome Informatics Workshop IV,* pp.36-45 (1993).

[132] Corpet,F. and Michot,B., "RNAlign program: alignment of RNA sequences using both primary and secondary structures," *Comput. Applic. Biosci.,* Vol.10, pp.389-399 (1994).

[133] Kumon,K. *et al.,* "Architecture and Implementation of PIM/p," *Proc. Fifth Generation Comput. Syst. '92,* pp.414-424 (1992).

[134] Sankoff,D., "Simultaneous Solution of the RNA Folding, Alignment and Protosequence Problems," *SIAM J. Appl. Math.*, Vol.45, pp.810-825 (1985).

[135] Lathrop,R.H. and Smith,T., "A Branch-and-Bound Algorithm for Optimal Protein Threading with Pairwise Amino Acid Interactions," *Proc. 27th Annu. Hawaii Int'l. Conf. Syst. Sci.*, Vol.5, pp.365-374 (1994).

[136] Suwa,M., Hirokawa,T. and Mitaku,S., "A Theoretical Method for the Determination of Helix Configuration in Membrane Proteins," *Proc. Genome Informatics Workshop IV*, pp.157-166 (1993).