記 号 処 理 69－1
（1993．6．18）

並列シミュレーテッドアニーリングによるアミノ酸配列解析

戸谷 智之　　石川 幹人
荒木 均*　　星田 昌紀*

（財）新世代コンピュータ技術開発機構 研究所 第2研究部
松下電器産業株式会社 東京情報システム研究所*

組み合わせ最適化手法であるシミュレーテッドアニーリングを並列化して、タンパク質アミノ酸配列の解析システムに実現した。複数のプロセッサがそれぞれ解と異なる温度パラメータを所有し、独立に探索を行い、確率的にプロセッサ間で解の交換を行った結果、適当な過程を経た解が低温のプロセッサに現れる“温度並列SA”と、各プロセッサはそれぞれ解と単純な温度スケジュールを持ち、プロセッサ間で順位検定を行うことにより、スケジュールの短縮を行う“並列検定SA”の2種類を、並列計算機PIM/m上に実現した。さらに、従来手法を用いたシステムを加えて、比較、評価を行い、いずれの並列システムも質の高い解の獲得を早期に可能にしていることを確認した。

Parallel Simulated Annealing Systems for Protein Sequence Analysis

Tomoyuki Toya, Masato Ishikawa,
Hitoshi Araki*, Masaki Hoshida*

Institute for New Generation Computer Technology
and Matsushita Electric Industrial CO., LTD*

Using the KL1, a parallel programming language, we developed two parallel systems for multiple sequence alignment on a loosely-connected parallel computer consisting of 64 processors, PIM/m. Multiple sequence alignment is a very typical method of protein sequence similarity analysis. Both systems employ simulated annealing, a stochastic method used to solve complex combinatorial optimization problems. By applying the first system, we can get good solutions without designing a cooling schedule. The second system has two features, the detection of the equilibrium and the passing solution. Both systems performed better than one of the most popular methods of multiple sequence alignment in a much shorter time than sequential simulated annealing systems.

1 はじめに

シミュレーテッドアニーリング（以下、SA と略す）は、組み合わせ最適化問題の有効な解決手法のひとつである [1]。我々は、この手法を用いて、タンパク質アミノ酸配列の類似性を検出するマルチプルアライメントの問題の解決を図ってきた。一般に、SA を用いて高品質の解を発見するためには、温度スケジュールと呼ばれるパラメータ列の最適化が重要である。今回我々が実装したふたつの新しい SA の並列アルゴリズムは、温度スケジュール設計の困難さからの回避を目指している。これにより、実用的なスケールの問題にも比較的手軽に SA を適用でき、かつ従来手法に比べてもより質の高い解が得られる。

今回は、そのふたつの並列 SA システムを紹介すると共に、マルチプルアライメント問題においての両システムの評価を行う。

2 シミュレーテッドアニーリング

SA は、組み合わせ最適化問題でローカルミニマム（局所的にはコスト最小であるが、大局的にはそうでない点）に捕まらずに、グローバルミニマムを探索することを可能にするアルゴリズムである。元来、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、ある物質を高温の状態から徐々に温度を下げることにより、非常に安定な物質が得られる過程のことである。SA とは、この焼きなまし過程を摸擬したアルゴリズムで、温度パラメータと呼ばれるパラメータの変遷に従って探索の範囲を変化させる探索手法である (図 1)。探索初期は、高温に相当する温度パラメータに設定する。高温とは、コスト（評価値）の悪化する方向への探索をも許すような温度パラメータを意味する。実行につれて、徐々に温度パラメータの値を小さくしていくことで、探索範囲を絞り込む。つまり徐々にコストの良くなるような方向への探索の度合いが大きくなる。

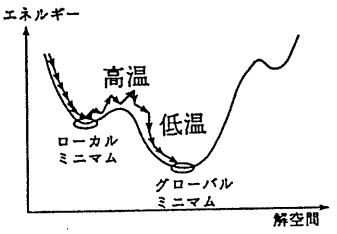

図 1: シミュレーテッドアニーリング

以下に、一般的な SA のアルゴリズムを説明する。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。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 Y_n を作る。最小化を目的とする評価関数（SA ではエネルギー関数とも呼ばれる）を E とすると、評価値の変化は $\Delta E = \mathrm{E}(Y_n) - \mathrm{E}(X_n)$ となる。$\Delta E \leq 0$ ならば、無条件に $X_{n+1} = Y_n$ とし、また、$\Delta E > 0$ のような場合には、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、$X_{n+1} = Y_n$ 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多数回繰り返す。ここで温度パラメータ列 $\{T_n\}$ を適切に設定することにより、エネルギー最小の解を求めることができる。

この温度パラメータ列 $\{T_n\}$ は、温度スケジュールまたは、クーリングスケジュールと呼ばれる。これを適切に設計することはなかなか困難な問題で、研究課題とされている。

理論的には、各温度でその温度における平衡状態に十分近付くだけ変換を繰り返して、温度無限大から徐々に温度を 0 に近付けていくと、コスト最小の解に収束する。実際には、ランダムウォークに近い探索を行う高温域では、一様分布に近い平衡状態に比較的短時間のうちに収束する。少しずつ温度を下げていくと、局所的にコストの低い解に落ち着く傾向が徐々に大きくなってくる。ここで平衡状態に達するだけの変換を繰り返したならば、大域的なコスト最小の解に達することが出来るはずである。しかし、本当に平衡状態に達したかどうかを検知することは困難である。従来から、各温度で一定回数微小変形を繰り返した後、T をある比率で段階的に下げていく温度スケジュールが単純でかつ比較的良い結果をもたらすとされているが、十分ではなく、経験的に決定せざるをえない。温度スケジューリングについては、いくつか研究が行われているが、これといったものはまだないようである。

また、SA を用いて、実用的なシステムを構築するのが難しいとされている点に、解を得るまでの所用時間の長さがしばしば挙げられる。並列化による高速化の研究もなされてはいるが、オリジナルの SA 自体は逐次的イメージの強いアルゴリズムゆえ、容易ではない。今回は SA の実行を複数のプロセッサに分割実行させるという直接的な並列化イメージとは少し異なる、2 種類の並列化を実装し、性能評価を行う。

3 シミュレーテッドアニーリングの並列化

我々は、ふたつの SA の並列システムに注目している。ひとつは、事前に温度スケジュールを詳細設計せずとも、低エネルギーの解を発見できることが特徴の手法 [2] で、温度並列 SA と呼ぶ。もうひとつは、従来用いられてきた各温度で一定回数の変換を行うという温度スケジュールを参考にするが、順位検定の手法をもとに擬似平衡状態の検出を行って、温度スケジュールの短縮を可能にする手法 [3] で、並列検定 SA と呼ぶ。これらふたつの並列化手法について、以下で詳しく述べる。

3.1 温度並列シミュレーテッドアニーリング

限られた時間内に可能な限り最適解に近づくためには、温度スケジュールを適切に設計することが重要であるが、それは非常に難しい問題である。以下に述べる並列化を用いることにより、人手による詳細な温度スケジューリング設計をかなり解消できる。この手法を用いることにより、ユーザは低温プロセスが所有する解の状態を監視しておくだけで、手軽に良い解を得ることが可能になる。仕組みは以下の通りである (図 2)。

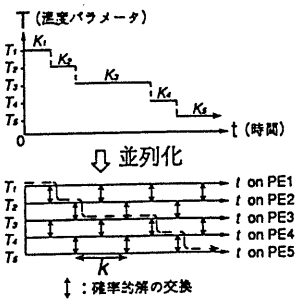

図 2: 温度並列 SA

最初に、各プロセッサごとに初期解とそのプロセッサが担当する温度パラメータを与える。プロセッサはそれぞれ、最初に与えられた温度パラメータに従った探索を一貫して行う。高温に相当する温度パラメータを与えられたプロセッサでは、ランダムサーチに近い探索を行い、温度パラメータが低温域に近い程、そのプロセッサでの探索は単純な降下法に近くなる。各プロセッサは並列に探索を行うことができるが、ある間隔ごとに隣接する温度パラメータ間で確率的な解の交換を試みる。温度パラメータ $T1$ において得られた解の評価値が $E1$、温度パラメータ $T2$ において得られた解の評価値が $E2$ の時、

$\Delta E = E1 - E2$、$\Delta T = T1 - T2$ とおく。この時、プロセス間の解の交換確率を、

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

として、定義する。

これにより、各プロセス上における Boltzmann 分布に従う平衡状態を崩さずに解の交換を行うことが可能になり、十分に時間をかければ最適解が得られることは保証される。この解交換の確率的制御により、実行以前に温度スケジューリングを詳細に設計する必要はなくなり、適当な温度プロセスの変遷を経た解が最終的に低温度プロセスに至ることが可能になる。それゆえ、低温を担当するプロセッサをモニターしていると、しかるべき時間の後に質の良い解が現れることが期待できるのである。

各プロセッサに割付ける温度パラメータの決定は以下のように考えた。一般に、各プロセッサに与える温度パラメータは高温域から低温域まである程度網羅されていることが必要なことは明らかであろう、良くない局所解に陥らないためには、中温域での十分な探索が重要であるという SA の性格上、中温域を多くのプロセッサに担当させることは意味のあることである。また、従来、温度スケジュールを設計する際に、温度は等比的に降下させていくのが単純な割にそれほど悪くはないと考えられている点も考慮に入れ、中、低温域での温度パラメータ列は等比列になるように選択した。

3.2 並列検定シミュレーテッドアニーリング

3.2.1 平衡状態の検出

SA は温度一定の微小変形を繰り返している間は、エネルギー平衡状態に向かう統計力学的な系としてとらえることができる。したがって、各温度で平衡状態に十分近づいた後、温度を次の段階に下げるようにしていくと、エネルギー最小の状態に収束することができる。このような SA の性質から、高温部では微小変形回数をさほど多くする必要はないが、中、低温部で微小変形回数が少ないとローカルミニマムにつかまる可能性があり、この可能性は微小変形回数が少なくなればなるほど大きくなる。各温度で一定回数実行していたのでは、どうしても高温部で必要以上の微小変形を行うこととなり、全体の実行時間を無駄に伸ばしてしまいがちである。そこで、複数プロセッサにおける解のエネルギー分布から擬似的に平衡状態を検出することにより温度を下げるタイミングを判定し、得られる解の質を落とさずに、高温域の余剰な変換を削減することが可能となる。

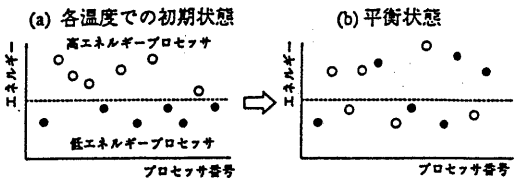

図 3: 平衡状態

最初に、各プロセッサには初期状態として、状態をひとつづつ与え、同じ高温パラメータからアニーリングを開始する。各温度で実行する微小変形の回数を決めて、温度スケジューリングを行うのが一般的であるが、このシステムでは、各温度で平衡検定を行い、それに適合すれば、決められた回数の微小変形を行う以前でも、次の段階に温度を下げる。平衡検定は以下のように行う (図 3)。各温度で探索を開始する時点で、解をエネルギー上位と下位の 2 グループにグループ分けし、マークを付けておく。その温度での実行が進み、変換が繰り返されるにつれ、各プロセッサの解はそれぞれ変化する。ふたつのグループ間でエネルギー分布に差がなくなった時点でこのシステムでは平衡状態に達したものと判断を下す。その後、温度を次の段階に下げ、再び、その時点でのエ

ネルギー上位、下位のふたつのグループに分け直し、同様な処理を行う。エネルギー分布の違いの検出には順位検定を用い、閾値はパラメータとして与えることとした。順位検定はふたつのグループ間で確率分布が等しいか、否かを判定する検定手法である。温度一定で変換を繰り返すと、平衡に達した時には初期状態に関係なく唯一の分布に収束するため、ある温度で最初にコストの上位と下位のふたつに分けたグループ間で、コスト分布に差がなくなれば、平衡状態に達したと考えられる。それゆえに、順位検定を用いた。

以下に、順位検定について、例を用いて、簡単に述べる。ふたつのグループを、

```
GroupX = {12,27,50,61,79,91,110}
GroupY = {10,18,25,30,39,85,120}
```

とする。これらのグループの要素をソートして、GroupXの要素はx、GroupYの要素はyで表すと、

```
{y,x,y,y,x,y,y,x,x,x,y,x,x,y}
```

となる。これをもとに、下図で、xを右に1マス、yを上に1マス進める規則で、左下から右上への経路を書いた時、それと左下と右上を結ぶ対角線で囲まれる面積を検定統計量として用いたものが順位検定である(図4)。

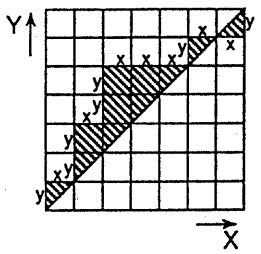

図 4: 順位検定

各温度で、最初に上位(X)、下位(Y)のふたつのグループに分けると、ソート結果は、

```
{x,x,x,x,x,x,x,y,y,y,y,y,y,y}
```

といった感じになり、得られる統計量は最大値である。実行が進むと、それぞれのエネルギー値は変化し、ソート結果は当然変わり、平衡に近づくと検定統計量はゼロに近い値になる。

3.2.2 局所解からの回避

先に述べたようにして、平衡状態の検出を行いつつ、SAを並列実行する。しかし、あくまでも、擬似的に検出された平衡状態であるから、必ずしも全プロセッサで平衡状態に達していたかどうかは確かではない、それゆえ、プロセッサによっては実行途中に局所的に比較的コストの低いところに陥るであろう。そういったプロセッサは残りの実行で意味のない探索を行ってるに過ぎない。そこで、そのプロセッサをさらに有効に活用する機能を付加する。ある期間、全く解に変動がなくなったプロセッサは、マスタープロセッサとの通信の際に、上位の筋の良さそうな解のコピーを受けとり、今まで自分が持っていたものを捨て去って、受けとったコピーの状態から探索を続行することとした(図5)。この機能を付加することにより、さらに低コストの解を発見できる可能性が増加すると期待できる。

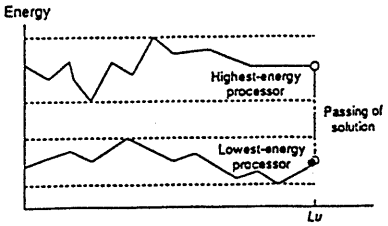

図 5: 局所解からの回避

4 アミノ酸配列解析への適用

我々は、遺伝子情報処理の分野のなかのタンパク質の配列解析に、ふたつの並列SAを適用し、実験システムを構築した。このシステムは、並列プログラミング技術の面からも、生物学的な実用性の面からも、多大な意義をもつ。

4.1 タンパク質と配列解析

よく知られているように、遺伝情報は細胞の内部にあるDNAに格納されている。タンパク質は、このDNAの情報から翻訳生成されるアミノ酸配列が、空間的に折れ畳まって特異的な形状になったものである。タンパク質は生物の体を形成し、生命の代謝反応を司る重要な物質であるので、その性質や形状などを詳しく知ることは非常に意義がある。

タンパク質の構成要素であるアミノ酸は20種類あり、それぞれ異なるアルファベットが割り当てられ表現される。それぞれのアミノ酸には、大きさ、水との親和性、酸性/塩基性、極性などの性質に違いがあり、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。タンパク質には大小さまざまなものがあり、短いのは数十個から、長いのは数百個まで、平均すると200から300個のアミノ酸が連なって出来ている。

タンパク質の構造や機能は、実験を積み重ねて初めてわかるものとされている。事実、タンパク質の正確な構造は、まだ300種類程度しか知られていない。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20,000種類以上のタンパク質について、その

配列が決定されている。この数字は近年、ますます増大している。

このように多くのタンパク質の配列データが集まるにつれて、新たな可能性が見えてきた。類縁のタンパク質は類似したアミノ酸配列をもつ傾向が明らかになったのである。すると、構造や機能が未知のタンパク質であっても、それと類似の配列をもつタンパク質の構造や機能が既知であれば、それから未知の構造や機能を推測することが可能になる。そこで必要になるのは、配列間の類似性を解析する技術である。この技術は、将来的に遺伝病の解明や新薬の開発に貢献する重要技術として位置づけることができる。

4.2 マルチプルアライメント

最も基本的な配列解析は、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。たとえば、以下のようなタンパク質の配列が4本あったとする。

```
GDVEKGKIFIMKCSQCHTVEKGGKHKTGPNLHGLFG
PYAPGDEKKGASLFKTAQCHTVEKGGANKVGPNLHGVFG
ASFAEAPAGTTGAKIFKTKCAQCHTVKGHKQGNGLFG
PPKARAPLPPGDAARGEKLRAAQCHTANQGGANGVGYGLVG
```

ここで、最上段左からGDVEKは、それぞれグリシン、アスパラギン酸、バリン、グルタミン酸、リシンを意味している。これをアライメントすると、次のようになる。

```
----------GDVEKGK-IFIMKCSQCHTVEKGGKHKTGPNLHGLFG
------PYAPGDEKKGASLFKT--AQCHTVEKGGANKVGPNLHGVFG
--ASFAEAPAG--TTGAKIFKTKCAQCHTV-KG--HKQG---NGLFG
PPKARAPLPPGDAARGEKL---RAAQCHTANQGGANGVG---YGLVG
```

配列のところどころにギャップ"-"を入れることで、QCHTなどの共通文字が同じ列に並んでいるのがわかる。QCHTのように複数の文字が、複数の配列で共通になっている文字の組を配列モチーフと呼び、タンパク質の構造や機能のうえで重要な部分を指し示していると判断される。

これは生物の進化の過程の仮説に基づいている。簡単に説明しておくと、このような遺伝情報が書き込まれている配列において、各文字は同じ確率で置き換わったり、失われてしまったりする。しかし、この配列上、変化が生じてはまずい部分があり、そのような部分に変化が起こった場合、機能が失われてしまったり、作り出される形状が変わってしまったりして、淘汰されてしまう。よって、生き残っている生物に共通に保存され続けている個所というのは必然的になんらかの重要な意味を持っていると考えることができる。そのような部分を抽出する意味で、マルチプルアライメントは重要である。

一般のマルチプルアライメントでは、ひとつの列に同じ文字が揃うことは少なく、異なる文字でもそれらが表すアミノ酸の性質が似ていれば、同じ列に置くことを許容して処理を行う。

4.3 マルチプルアライメント問題へのSAの適用

マルチプルアライメントにSAを適用するためには、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、各状態における評価尺度にあたるエネルギー関数を定義する必要がある。

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難となる。そこで、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[4]。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、その間にあった文字列を、選ばれたギャップのあった方へ1カラム分移動させる(図6参照)。ただし、両サイドのギャップは配列中にギャップが入り過ぎることを考慮して、ギャップがいくつあってもひとつとみなして確率的に選択する。

```
(微小変形前)
 -----LLDFLHQLTHLSFSKMKALLERSHSPYYMLNRDRTL-----
 -----VLQLSPAELHSFTHCGQTALTLQGATTTEASNILRS----- ←
 -----AYPLREAKDLHTALHIGPRALSKACNISMQQAREVV-----
(微小変形後)
 -----LLDFLHQLTHLSFSKMKALLERSHSPYYMLNRDRTL-----
 ----VLQLSPAELHSFTHCGQTALT-LQGATTTEASNILRS-----
 -----AYPLREAKDLHTALHIGPRALSKACNISMQQAREVV-----
```

図6: 基本微小変形例

この微小変形は、相同性の高い配列群のマルチプルアライメントは比較的うまく行えるのに対し、ギャップが固まりで入るような相同性の低い配列間のマルチプルアライメントは、苦手である。それは、ギャップの固まりが配列内部に形成されにくいためである。その欠点を補うために、ギャップを長方形の固まりで動かすブロックオペレーション[4]を導入した。ブロックオペレーションは、あるギャップをランダムに選んだならば、そのギャップの横方向や縦方向にギャップの連なりを探し、矩形ブロックの単位でギャップ群を移動させる微小変形である。

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る。現在エネルギー関数として、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[5]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。Dayhoffマトリックスとは、20種類のアミノ酸の各ペアの相同性を表す数値表である。Dayhoffマトリックスには、ギャップ対アミノ酸の評価値としての指定はないので、パラメータとして取り扱えるようにしている。ギャップを含むペアのコストについては、長さkのギャップに対して$a+bk$のような一次式が与えられる。aとbは可変であるが、上記のアミノ酸のDayhoff値とのかねあいから、通常は、$a=7$, $b=1$としているが確かなものではなく、実験的に決めざるを得ない現状である。

以上のように定式化して、SAを用いてマルチプルアライメントの問題に取り組んだ。

5 並列論理型言語KL1によるふたつの並列SAの実装とその評価

我々は、このふたつの並列SAを用いたマルチプルアライメントシステムを、並列論理型言語KL1で記述し、64個の要素プロセッサ（以下、PEと呼ぶ）を持つ並列マシンPIM/m上で実行、性能評価を行った。

5.1 並列論理型言語KL1

KL1という言語の特徴を一言でいうなら、記号処理を並列に行うための言語ということになる。記号処理という側面では、データ構造についてのメモリの割付けや解放に気を取られずに済む自動メモリ管理機能や一意的名前付けを可能にする機能が特徴で、ほぼLispなどと同様な機能を持っている。そのうえ、並列処理を容易に行えるように設計されている。KL1では、プログラムは並列に実行されることが前提になっており、その際に複雑になる同期の問題にはデータフロー同期機構により同期が自動化され、プログラマの同期の誤りによるバグの混入を防ぐ特徴を持つ。ユーザによる物理的並列実行の指定は、プラグマと呼ばれる記述の追加により可能で、プログラム自身の正当性を変更することなく、並列処理の仕方を変更することが可能になっている。以下に述べるふたつの並列SAの実装においては、複雑な動的付加分散を行っているわけではなく、静的に各プロセッサにプロセスの割付けを行っているだけである。しかし、これにおいても、プロセス間での通信などにおいて、KL1の特徴からくる同期バグの追放はシステム実装を非常に容易なものとしている。

5.2 温度並列SAの実装

本システムにおいては、適当な数の温度パラメータ分の一定温度での探索プロセスを定義する。温度パラメータの数はさほど多くの数は必要ないため、1プロセッサに1温度プロセスを割りつけた。最初に、問題に従い、解が十分ランダムな状態になる高温域の温度パラメータを実験的にみつけ、それを最高温度とし、一定比率0.9をかけあわせていくことで、温度パラメータ列を決定した。マスタープロセスが最初に、各プロセッサに決定された温度パラメータ列の要素をひとつずつ与え、各プロセッサは与えられた温度パラメータを用いて探索を行う。プロセス定義時に、隣接するプロセスとの間に通信路（KL1のストリームにより容易に記述可能）を設けておく。そして、あるインターバルごとにその通信路を介して、メッセージのやりとりを行い、確率的解交換を実現した。また、解の状況を実行とリアルタイムに監視できるように、マスターから監視ストリームを指定したプロセスに張っておく。このストリームは、リアルタイムに他のプロセスへ張り替えることも可能なようにしてあり、いろんなプロセスの状況をチェックできる。このストリームの先のプロセスはあるインターバルごとに解の情報をマスターに送り、マスターはその情報をもとにディスプレイに解を表示する仕組みである。

5.3 検定並列SAの実装

本システムも、アニーリングプロセスとマネージャプロセスの2種類のプロセスから構成されている。アニーリングプロセスは初期状態、初期温度、減温比率、1温度での上限となる微小変形回数が与えられており、初期温度からアニーリングを行うものである。アニーリングプロセスは各プロセッサに割り当てられ、それぞれ並列に実行される。各アニーリングプロセスはマネージャープロセスと通信路を保持しており、指定したインターバルごとに各アニーリングプロセスからエネルギー値が報告される。マネージャは全てのPEのアニーリングプロセスから情報が届いた時点で、温度を下げるか否かを順位検定により決定し、終了チェックをした後、各アニーリングプロセスに、決定された命令を与える。これらのプロセス間の通信はやはりKL1のストリームを用いることで容易に実装できるのである。各アニーリングプロセスは、マネージャプロセスからの命令に従い、温度を維持、もしくは減温比率に従い温度を下げてアニーリングを続けるか、または収束条件に達した場合にはアニーリングを終了させるかする。

また、アニーリングプロセスは一定回温度が下げられる間、解に変動がなくなった場合、独自に局所解に捕まったと判断する。そして、その解は諦め、他のプロセスが持つ評価値の良い解のコピーが欲しい旨をマネージャに伝える。マネージャは所持している最も良い解のコピーを送り返し、それを受けとったプロセスは、その解をもとに探索を続行する。終了条件としては、最も良い評価値を持つプロセスで一定時間、変化が起きなくなった時点で全体の実行を終了する。

5.4 評価

６０個のアミノ酸から構成されるタンパク質の前後にあらかじめギャップを１５個前後ずつ付け加えた、全長９０の文字列６本をサンプル問題として実験を行った。実行には、格子上に要素プロセッサがメッシュ結合された並列計算機PIM/m [6] で、64 の要素プロセッサから構成されたものを用いた。

比較には、これまで計算機上で用いられてきているマルチプルアライメント手法 (Tree-based Method) により得られる解の評価値を基準にした。オリジナルの SA の１プロセッサでの実行（逐次 SA）、それの微小変形などを決める乱数列のみを変更して、複数プロセッサで並列に実行させたもの（単純並列 SA）を加えて、今回のふたつの並列 SA を比較、考察してみた。なお、各システムで用いた最高温度は、微小変形の 90% 受理を目安に実験的に設定した。また、温度スケジュールを必要とするシステム（逐次 SA および単純並列 SA）には、$r = 0.9$ の等比列を用い、各温度でのインターバルは微小変形回数 Lu をパラメータとして用いた。並列検定 SA でも同じ温度スケジュールをもとにしており、Lu は最大微小変形回数に対応する。温度並列 SA では、先程設定した最高温度をもとに、やはり $r = 0.9$ の等比列から各プロセッサの持つ温度パラメータを決めた。

従来手法による結果と逐次SA、単純並列SAの結果 図7は、従来手法により得られる評価値、逐次 SA を Lu ごとに 63 回実行した結果の平均値と単純並列 SA を 4 回実行した結果の平均値をプロットしたグラフである。逐次 SA では、数千 sec の実行時間では従来手法のレベルにまで至らないことがほとんどであった。実行時間をどんどん伸ばせば、さらに良い解が得られそうではあるが、このように実用に近い規模の問題では現実的な実行とは言えない。従来から言われている SA の実行時間がかかり過ぎるという問題そのものであろう。単純並

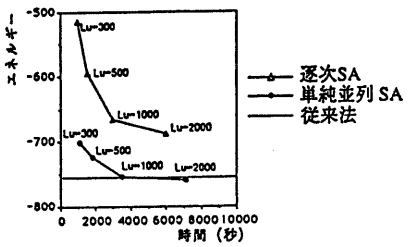

図 7: 従来手法の結果と逐次 SA、単純並列 SA の能力

列 SA とは、逐次 SA を各プロセッサで独立に並列実行させたもので、それにより得られた最良解を解としている。この結果から分かるように、SA も何回も実行してみると、中には比較的短時間（数千 sec）のうちに、いい解が得られることもあることがわかる。Lu を変えていくつかデータを集めている。Lu を大きくするほど当然実行時間は伸びるが、解の質も向上し、Lu = 1000 から 2000 で、従来手法を上回る評価の解を発見できている。ただ、単純であるがゆえ、やはり偶然性に頼る側面が大きすぎ、信頼性という点で少し心細い。

温度並列SAの実行結果 次に、図8として、図7のグラフの上に温度並列 SA の最低温度プロセスに表れた解の評価値の履歴をプロットしてみた。

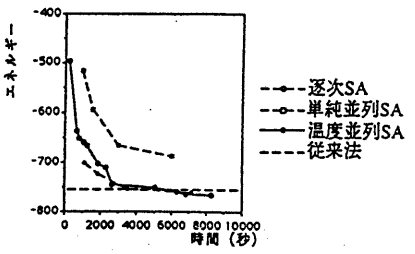

図 8: 温度並列 SA の性能

温度並列 SA は、特に温度スケジュールの設計を行っていないにも関わらずかなり短時間のうちに従来手法に優る解をみつけている。単純並列と比べてみても、従来手法よりよい解が得られる実行時間になると温度並列 SA の方に優位性が見られる。さらに、通常の SA では、折角実行しても満足な解が得られなかった場合、再度アニーリングをしなおすしかないのだが、この温度並列 SA なら、実行を延長することで、さらなる解の改善が期待できる。ただし、どこで実行を打ち切るのが適当かを正確に判断することが難しいとも言えなくもない。

並列検定SAの実行結果 今度は、図9は、やはり図7のグラフの上に、並列検定 SA の結果を Lu ごとにプロットしてみた。

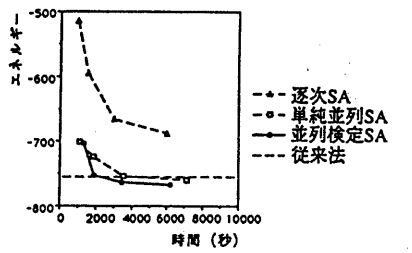

図 9: 並列検定 SA の性能

単純並列同様、Lu = 1000 程度、実行時間にして約 1 時間程度で。従来手法の評価値を更新している。但し、このシステムでの Lu はインターバルの最大値を意味する。また、単純並列 SA と比較しても、検定機能の付加はそれほど実行時間には影響を与えてないようであり、それでいて、解の質は各 Lu ごとに同等以上であ

る。Luの値の調整が必要という点は同じであるが、単純並列よりは期待できる。

並列検定SAにおける局所解判定機能の効果　並列検定SAにおいて、局所解に陥ったプロセッサを検出して、別の探索に割り振る機能の有効性をチェックしてみた。

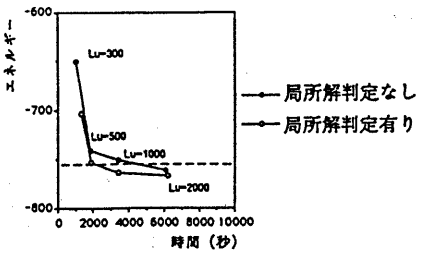

図10: 並列検定SAにおける局所解判定機能の効果

図10には、局所解判定の有無別に、実行結果をプロットしてみた。局所解判定を組み込んだ結果、ほぼ同等の時間でより評価値の良い解を発見できており、この機能の付加は有意義であったと結論出来る。つまり、完全な温度スケジュールの実行ではないのだから、ある程度探索が進んだ段階で局所解に陥っているわけであり、そこで諦めて、他の解を複数のPEで探索する意義が認められている。特に、実行時間が短めな場合ほど、効果は顕著なようである。それは、局所解に陥り易い実行だからと言えるであろう。

温度並列SAと並列検定SA　最後に、今回のテーマである、このふたつの並列SAの結果を同時にグラフ上にプロットしてみた（図11）。

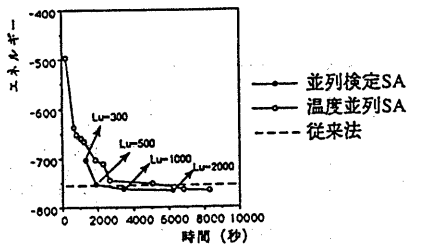

図11: 温度並列SAと並列検定SA

いずれもの手法も逐次SAより飛躍的に短時間のうちに、従来手法以上の良い評価値の解を見つけることに成功しているとみなせる。温度並列SAの方が若干実行時間的に不利ではあるが、特に温度スケジュールを与えずに、確率的な温度の変遷によるだけでこれだけのものを得られるのは大きなメリットであるとも考える。また、解の質の点でも、温度並列SAの場合、実行を続行させることで、新たに良い解が得られる可能性を残しているのは先に延べたとおりである。並列検定SAの場合、全体の実行時間はパラメータLuで決まるため、解が不十分な場合、アニーリングをやり直すなどの必要が生じるのは、従来のSAと同じで、温度並列SAに劣る部分ではある。しかし、いずれも今回の実行に関しては、従来手法に優る解を安定してそれなりの時間のうちに獲得できており、十分実用的だと結論できる。

6　おわりに

今回、SAの新しい並列方式を2種類とりあげ、その有効性をタンパク質の配列解析の問題を例に、逐次的手法も交えて、比較、検証を行った。逐次にSAを行った場合に最大の問題となる実行時間の点で、これらの並列化には、かなりの効果がみられている。単純にSAを何回も実行を行えば、偶然にいい解を得られることもありそうであるが、今回のふたつの新しい並列SAでは、実行時間や得られる解の質の安定性のうえで明らかなアドバンテージが見られている。SAを用いるべき問題というのは解空間の性質がよく分かっていないような問題であるべきであり、そういった問題には十分適用してみるに足る手法であろう。

また、今回は、いずれも複数の探索プロセスを各プロセッサにひとつずつ配置し、それらを並行に実行するという大粒度の並列化アイデアに絞って研究を行い、探索プロセス内の並列性には触れていない。SAの探索プロセスは比較的逐次的な処理が主であるが、それでも並列化により数倍程度の高速化の余地を残していると思われる。

謝辞　本研究にあたっては、生物学応用のご指導をいただいた金久實京都大学教授に感謝します。また、SAに関して助言を頂いた木村宏一氏に感謝します。

参考文献

[1] S.Kirkpatrick, C.D.Gelatt and M.P.Vecci : "Optimization by Simulated Annealing" Science, vol.220, no.4598, pp.671-681, 1983.

[2] 木村、瀧： 時間的一様な並列アニーリングアルゴリズム 電子情報通信学会 NC90-1,1990.

[3] 荒木、舘野、加藤、間藤： 疎結合並列計算機上でのシミュレーテッドアニーリング 情処研報, 91-AI-77-2, 1991, pp.7-14.

[4] M.Ishikawa, T.Toya, M.Hoshida. K.Nitta, A.Ogiwara, and M.Kanehisa : Multiple sequence alignment by parallel simulated annealing, Comput. Applic. Biosci. Vol.9 No.3, 1993.

[5] Dayhoff, Hunt and Hurst-Calderone : "Composition of Protein" Atlas of Protein Sequence and Structure 5:3, Nat. Biomed. Res. Found., Washington, D.C., 1978, pp.363-373.

[6] Uchida : "Summary of the Parallel Inference Machine and its Basic Software" Proc. Int. Conf. on Fifth Generation Computer Systems, 1992. pp.33-49.