ICOT Technical Report: TR-565

TR-565

時間的一様な並列アニーリングアルゴリズム

木村宏一，瀧　和男

May, 1990

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03) 456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

時間的一様な並列アニーリングアルゴリズム
On a Time-homogeneous Parallel Annealing Algorithm

木村 宏一　　瀧 和男
Kouichi KIMURA　　Kazuo TAKI

(財) 新世代コンピュータ技術開発機構
Institute for New Generation Computer Technology (ICOT)

概要

シミュレーテッドアニーリング法 (SA 法) を並列化した新しいアルゴリズムを提案する。このアルゴリズムでは、各プロセッサに解を一つずつ与え、それぞれ互いに相異なる一定温度でアニーリングを同時並列的に行い、更に、プロセッサ間で解の交換を確率的に行う。これにより、通常の SA 法のように温度を時間とともに注意深く低下させることが不要となり、より安定した最適化能力が得られると期待される。本報告では、プロセッサ間の解の交換をどのように行なうべきかを決定し、それについて確率論的な考察を行なう。また、グラフの分割問題を例題として実験を行い、本アルゴリズムで得られた解を、通常の SA 法及び Kernighan-Lin アルゴリズムで得られた解と比較評価する。

1　シミュレーテッドアニーリング法

シミュレーテッドアニーリング (SA) 法 [Kirkpatrick 83] は、広範囲の組み合わせ最適化問題に適用可能な、汎用的な最適化手法の一つである。SA 法は、確率的アルゴリズムの一種であり、擬似乱数を用いて確率的な動作を行う。そのアルゴリズムを以下に示す。

X を解空間、$E : X \to R$ を最小化すべき目的関数とする。SA 法は、一つの初期解 x_0 から出発して、解の系列 $\{x_n\}_{n=0,1,2,\ldots}$ を、次のように順に生成して、最適解に近づいていく。まず、解 x_n をランダムに微小変形した解 x'_n を一つ生成し、目的関数の変化 $\Delta E = E(x'_n) - E(x_n)$ を計算する。$\Delta E \leq 0$ ならば、$x_{n+1} = x'_n$ とし、$\Delta E > 0$ ならば、$p = \exp(-\Delta E/T_n)$ として、確率 p で $x_{n+1} = x'_n$, 確率 $1-p$ で $x_{n+1} = x_n$ とする。ここで、T_n は正のパラメータであり、n と共に単調に減少させる。以下では、x_n から x_{n+1} を生成することを、アニーリングの 1 ステップと数えることにする。

SA 法は、$T_n \equiv +\infty$ のとき単なるランダムな探索、$T_n \equiv +0$ のときは手近の局所最適解に収束する所謂反復改善法となり、$0 < T < +\infty$ のときはそれらの中間的な性格を持つ。

SA 法のアイディアは、統計力学とのアナロジーに基づいている。目的関数 E をエネルギー、パラメータ T_n を温度、$\{T_n\}_{n=0,1,\ldots}$ を温度スケジュールと呼ぶ。$T_n \equiv T$ が一定の時、SA 法は、温度 T の下で熱平衡状態にある熱力学的系のシミュレーションを行なっていることに他ならない。従って、このとき得られる解の系列 $\{x_n\}$ は所謂 Boltzmann 分布に従う。この Boltzmann 分布は、温度 $T \to +0$ のとき、エネルギー最小の解、即ち、最適解に収束する。従って、原理的には SA 法で真の最適解が得られると期待できる。しかし、そのためには、温度を、

$$T_n \geq A/\log n \tag{1.1}$$

(A は関数 E の凹凸の程度を表す或る定数)

を満たすように極めてゆっくり下げねばならず、そのためには非現実的なほど長い計算時間が必要となる [Hajek 86]。従って、真の最適解を得ることは事実上不可能であり、限られた時間内に可能な限り最適解に近い解を求めることが目標となる。しかし、そのためには、温度スケジュールの最適化という、一種の確率制御の問題を解かなければならない。そのような理想的な温度スケジュールを求めることは、少なくとも、もとの組合せ最適化問題を解くことと同様に難しい。一方、全く不適切な温度スケジュールの下では、不満足な解しか得られない [White 84]。そこで、通常は、例えば、

$$T_n = \alpha^{\lfloor n/K \rfloor} \cdot T_0, \quad n = 1, 2, \ldots \tag{1.2}$$
$$0 < \alpha < 1, \qquad K \gg 1$$

の形の温度スケジュールが用いられる。これは最も単純で、かつ比較的良い結果をもたらすことが経験的に知られている [Kirkpatrick 83]。しかし、温度スケジュールの選び方は、アルゴリズムの性能を大きく左右するので、更に、より複雑で巧妙な種々のスケジュールが提案され、研究されている [Sechen 86][Aarts 85][Laarhoven 87]。温度スケジュール設定の問題は、SA 法に常につきまとう問題である。

SA 法は、従来の他のアルゴリズムと比較して、多大な計算時間を要するが、より良い解が得られるということか

ら注目された [Sechen 85]。そこで、SA 法を並列化して高速化しようという研究も行われている。特に、解を表現するデータを分割して各プロセッサに割り当て、各プロセッサ上で同時並列的に解の各部の改善を行う方法が、最も良く研究されている [Darema 87][Casotto 87]。このとき、解の微小変形に伴う目的関数値の変化を正確に計算できなくなる等の問題が生じるが、それがアルゴリズムの最適化能力にどのような影響を及ぼすかについて検討されている。

本稿では、上とは異なる、SA 法の新たな一つの並列化法を提案する。この並列化では、処理を高速化することはできないが、温度スケジュールを不要にする。これにより、SA 法に常につきまとう温度スケジュール設定の問題に対し、一つの解答を与えた。

2 SA 法の並列化アルゴリズム

2.1 基本的なアイディア

通常の逐次の SA 法では、温度 T を温度スケジュールに従って減少させる。効率的な最適化を行なうためには、温度スケジュールをうまく設定することが重要な課題である。そこで、各プロセッサに解を一つずつ与え、互いに相異なる一定温度で同時並列的にアニーリングを行うことを考える。このとき、逐次アルゴリズムで温度 T を温度 T' に減少させることに対応して、この並列アルゴリズムでは温度 T のプロセッサから温度 T' のプロセッサへ解を渡すことを考える。更に、対称性の観点から、プロセッサ間の解の受け渡しは、互いの解の交換によって行うものとする。

このとき、逐次アルゴリズムで温度スケジュールを設定することは、並列アルゴリズムではプロセッサ間の解の交換をいつ行うかを指定することに相当する。そこで、プロセッサ間の解の交換を確率的に行わせて自動化し、温度スケジュールを不要にすることが考えられる。即ち、確率的な解の交換が、良い温度スケジュールを選び出してくれることを期待するのである。

そこで、次のような用語の定義を行なう。二つのプロセッサの間で解の確率的交換を行なうとは、何らかの基準により実数 $0 \leq p \leq 1$ を定めて、確率 p で互いの解を交換し、確率 $1-p$ で何もしないことと定義する。ここで、p を解の交換確率とよぶ。p を定める基準については、次節で論ずる。

この用語を用いて、本並列アルゴリズムを次のように定める。各プロセッサに解を一つずつ与え、互いに相異なる一定温度で同時並列的にアニーリングを行う。このときアニーリングを一定ステップ数 k だけ行なう毎に、互いに隣接する温度をもつプロセッサどうしの間で一斉に解の確率的交換を行なう。こうして、最終的に最低温度のプロセッサで得られる解を答とする。ここで、k の逆数を解の確率的交換の頻度と定義する。

2.2 プロセッサ間の解の交換確率の定め方

前節で述べたように各プロセッサ毎に互いに相異なる温度でアニーリングを行った場合、高温のプロセッサでは比較的自由に解空間が探索されて最適化はあまり進まないのに対し、低温のプロセッサでは E の減少する方向ばかりに探索が進むので逆に局所最適解に陥り易い。従って、単に各プロセッサ毎に独立にアニーリングを行っただけでは、最低温度のプロセッサで最良の解が得られるとは限らない。そこで、このような温度とエネルギーとの逆転現象を解消するため、まず、次のように解の交換確率を定める。

$$(T-T')(E-E') < 0 \quad \Rightarrow \quad p(T,E,T',E') = 1$$

ここで、$p(T,E,T',E')$ は、温度 T のプロセッサがエネルギー E の解を有し、温度 T' のプロセッサがエネルギー E' の解を有するとき、それらのプロセッサの間で解の交換を行う確率を表す。

次に、$(T-T')(E-E') \geq 0$ のときの $p(T,E,T',E')$ の値を定めるために、以下の考察を行う。今、仮に、無限の時間が費やされ、各プロセッサ上で Boltzmann 分布に従う平衡状態が実現されたとする。プロセッサ間の解の確率的交換は、この平衡状態を崩してはならないから、詳細釣合の原理から、

$$\frac{1}{Z(T)}\exp(-\frac{E}{T})\cdot\frac{1}{Z(T')}\exp(-\frac{E'}{T'})\cdot p(T,E,T',E')$$

$$= \frac{1}{Z(T)}\exp(-\frac{E'}{T})\cdot\frac{1}{Z(T')}\exp(-\frac{E}{T'})$$

$$\therefore \quad p(T,E,T',E') = \exp(-\frac{(T-T')(E-E')}{TT'})$$

が成り立たなければならない。ここで、$Z(T)$ は分配関数である。以上より、プロセッサ間の解の交換確率は次のように定めるべきであることが示された。

(2.2.1)

$$p(T,E,T',E') = \begin{cases} 1 & \Delta T\cdot\Delta E < 0 \\ \exp(-\frac{\Delta T\cdot\Delta E}{TT'}) & \text{otherwise} \end{cases}$$

$$\Delta T = T - T', \qquad \Delta E = E - E'$$

2.3 解の確率的交換の効果 — Kullback 情報量による検討

本節では、前節で一意に定められた解の確率的交換が、実際、最適化に貢献する処理であることを示す。

一般に、離散的な確率分布 $p = (p_i)$, $q = (q_i)$ に対して、Kullback 情報量、

(2.3.1) $$D(q\|p) = -\sum_i q_i \log\frac{p_i}{q_i} \geq 0$$

は、p と q とが互いにどの程度離れているかを表す尺度である [Amari 85]。

上の並列アルゴリズムにおいて、或る時点における全プロセッサ上の解の集まりの分布を p とし、平衡状態におけるその分布を π とする。π は、各プロセッサの温度に対する Boltzmann 分布の直積である。このとき、$D(\pi\|p)$ はその時点で系がどの程度平衡状態に近づいているかを表す。

今、各プロセッサ上でアニーリングを 1 ステップずつ進めると、解の集まりの確率分布は p から pA に変化する。ここで、A は或る推移確率行列であり、その具体的な形は付録の節で与える。また、全プロセッサを幾組かのペアに分け、各ペアで一斉に解の確率的交換を行うと、解の集まりの確率分布は p から pC に変化する。ここで、C も或る推移確率行列である。このとき、付録の節で述べるように、常に、

$$D(\pi\|p) \geq D(\pi\|pA) \tag{2.3.2}$$

$$D(\pi\|p) \geq D(\pi\|pC)$$

が成立することが示される。

従って、各プロセッサ上でアニーリングを 1 ステップずつ進めても、或いは、プロセッサ間で解の確率的交換を行っても、解の集まりの確率分布は、平衡状態に単調に近づくことが結論できる。

2.4 アルゴリズムの時間的一様性

上記の並列アルゴリズムは、通常の逐次 SA 法における温度スケジュールのような、処理の進行とともに変化させるべきパラメータをもたない。即ち、このアルゴリズムは時間的に一様である。

したがって、このアルゴリズムの確率的な動作は、時間的に一様なマルコフ連鎖として記述される。また、このマルコフ連鎖は、明らかに、既約かつ非周期的である。従って、確率論でよく知られているように [Feller 57]、このマルコフ連鎖は、その唯一の定常状態 (平衡状態) に法則収束することが判る。

また、アルゴリズムの時間的一様性により、或る時点で処理を打ち切って得られた解が不満足なものならば、処理をそのまま継続して更に最適化を図ることができる。これに対し、通常の逐次の SA 法では、得られた解が不満足なときは、温度を再び上げることが必要になる。しかも、このとき温度をどの程度上げるべきかが問題となる。

3 KL1/ マルチ PSI による実験評価

上記の確率的並列最適化アルゴリズムを、並列論理型言語 KL1[Chikayama 88] で記述し、並列推論マシンのプロトタイプ機マルチ PSI/V2[Nakajima 89] を用いて実験を行ない評価した。

例題として、次のようなグラフの分割問題を用いた。

問題 グラフ $G = (V, E)$ が与えられたとき、各頂点 $v \in V$ にラベル $\lambda(v) = \pm 1$ を与え、目的関数、

$$E = -\sum_{(u,v)\in E} \lambda(u)\lambda(v) + c \cdot \Big(\sum_{v\in V} \lambda(v)\Big)^2 \tag{3.1}$$

$$c > 0 : \text{定数}$$

を最小化せよ。

これは、直観的には、グラフの頂点を性能同数の 2 グループに分け、それらのグループ間にまたがる辺の数を少なくせよ、という問題である。これは、NP 困難な組み合わせ最適化問題の典型的な例である [Garey 79]。

マルチ PSI の 64 台のプロセッサのうち、1 台は全体の制御、入出力、実行結果のデータ整理等の専用とし、残りの 63 台に等比的に温度を割り当てた。温度の上限、下限は試行錯誤的に決めた。各プロセッサには、アニーリングを行う KL1 プロセスを置き、解を表現するストリングデータをプロセッサ間で交換させた。

以下では、頂点数 400、辺数約 2000 のランダムグラフに対して、各プロセッサ上で 20000 ステップずつアニーリングを行った結果を示す。実験は、乱数の種を変えて 3 回ずつ行った。

図 1 に、プロセッサ間の解の確率的交換の頻度を変えて得られた、解のエネルギー (目的関数) 値を示す。また、比較のために、式 (1.2) の温度スケジュールを用いた通常の逐次 SA 法、及び、Kernighan-Lin アルゴリズム [Kernighan 69] で得られた解のエネルギー値を示す。前者は、上で各プロセッサに割り当てたのと同じ温度でそれぞれ同ステップずつ全体で 20000 ステップのアニーリングを、乱数の種を変えて 63 通り行った結果の平均値である。後者は、63 通りのランダム初期分割に Kernighan-Lin アルゴリズムを収束するまで繰り返し適用した結果の平均値である。

ここで、逐次 SA 法の結果があまり良くないのは、分割すべきグラフの頂点数、辺数に対して、アニーリングのステップ数が小さいからである。このとき、式 (1.2) のように各温度に平等にステップ数を割り当てると、最適化が最も効果的に行われる温度領域で相対的にステップ数が不足すると考えられる。一方、本手法のような並列化を行うと、より効果的な温度スケジュールが確率的に選択されると考えられる。また、本手法では、解の交換の頻度が極端に少ない場合を除き、Kernighan-Lin 法より良い解が得られている。本例題に関しては、解の交換頻度が 1/400 程度以上であれば、十分な最適化能力が得られることが判る。

ところで、プロセッサ間の解の確率的交換の際には、プロセッサ間で逐次性のある一連の処理を行う。このとき、一方のプロセッサの計算結果を他方のプロセッサが待つことにより、プロセッサがアイドル状態になることがある。従って、解の交換頻度を f とすると、リダクション数、処理時間、プロセッサのアイドル時間は全て f の一次関数として増加すると考えられる。よって、プロセッサの稼働率は f の一次分数関数で低下することが予想できる。表 1 に、プロセッサ間の解の確率的交換の頻度 f を変えたときの、

リダクション数、処理時間の実測結果を示す。上で述べた $f = 1/400$ の場合では、解の確率的交換を1回しか行なわなかった $f = 1/20000$ の場合と比較することにより、解の交換処理の占める計算コストはごく僅かであることが判る。表1をグラフに描くと、上で予想した線形性が確認できる(図2)。このグラフで表される一次関数の係数を読みとって、プロセッサの稼働率を次のように推定できる。

$$(3.2) \qquad (\text{稼働率}) = \frac{1+\rho f}{1+\tau f} \times 100(\%),$$

$$\rho = 0.516, \qquad \tau = 9.40,$$

$$f = (\text{解の確率的交換の頻度})$$

表2に、この推定式に基づく計算結果と、パフォーマンス・メータでの測定結果との比較を示す。パフォーマンスメータは、マルチPSIの各プロセッサの稼働率を10%の精度でリアルタイムにフロントエンドプロセッサ画面に表示するツールである。これらの計算結果と測定結果は概ね一致している。解の交換頻度が1/200程度以下ならば、9割程度の稼働率が得られることが判る。

表3に、本手法、逐次SA法、Kernighan-Lin法で得られた解の比較評価結果を示す。これは、図1の解の交換頻度 $f = 1/400$ での比較に対応する。この表より、並列化による最適化能力の向上が確認できる。

4 結言・今後の課題

従来のSA法では、効率的な最適化のためには、温度スケジュールの微妙な調整が必要であった。しかも、温度スケジュール自身を最適化することは、易しい問題ではなかった。本稿では、SA法を並列化して、温度スケジュールを用いない新しい並列確率的最適化アルゴリズムを提案した。即ち、このアルゴリズムは、処理の進行とともに変化させるべき処理制御用のパラメータを持たず、時間的に一様である。従って、応用上の観点からはより容易に効率的な最適化が実現でき、また、理論的観点からは分析が容易になるという利点をもつ。本稿では、確率論の議論により、本アルゴリズムの正当化を行なった。また、簡単な例題を用いて計算機実験を行ない、本アルゴリズムの最適化能力を確認した。

しかし、本アルゴリズムを実際の問題に容易に適用し得るためには、次の課題を検討する必要がある。

- 一般に、どの範囲の温度をどのような間隔で、各プロセッサに割り当てれば良いか。
- 一般に、アルゴリズムの最適化能力を損なわない範囲での、プロセッサ間の解の確率的交換の頻度の下限値を、評価できないか。

本稿では、一例題についてのみ、試行錯誤によって、これらの課題を解いた。

また、逐次アルゴリズムと比較して、本手法での並列化により最適化能力が"常に"向上するか否かに関しては、理論的な結果を導くことができなかった。本稿では、一例題について、本手法と、式(1.2)の温度スケジュールによる逐次SA法との実験比較を行った。そこで、次のことが一般的に成立するかどうかの理論上の課題が残る。

- $\{T_1, \ldots, T_N\}$ 上に値をとる任意の温度スケジュールの逐次SA法と比較して、N プロセッサ上で本手法を行えば、同じアニーリング・ステップ(時間)内で、常に平均的に同等以上の質の解が得られるか。

謝辞

本稿を読んでコメントを下さり、また、御討論いただいた、六沢一昭氏、市吉伸行氏に感謝します。また、KL1プログラミングに関して助言を頂いた杉野栄二氏に感謝します。

A 付録

本節では、SA法を並列化して得られた上述の並列アルゴリズムをマルコフ連鎖として表現して、その推移確率行列を求め、2.3節で与えた式(2.3.2)の証明を与える。

X を探索すべき解空間とし、

$$E : X \longrightarrow R \qquad i \longmapsto E_i$$

を最小化すべき目的関数、即ち、エネルギーとする。第1節のSA法のアルゴリズムの中で述べたような、解 x_n から解 x'_n をへのをランダムな微小変形は、確率行列、

$$S = (s_{ij})_{i,j \in X}, \qquad 0 \le s_{ij} \le 1, \qquad \sum_j s_{ij} = 1$$

で表現される。即ち、各 s_{ij} は、条件付き確率、

$$s_{ij} = \Pr(\, x'_n = j \mid x_n = i \,)$$

を表す。ここで、SA法が正しく機能するためには、S は対称かつ既約でなければならない。このとき、一定温度 T の下での逐次のSAアルゴリズムの挙動は、次の推移確率行列 A をもつマルコフ連鎖として表現される。

$$(A.1) \qquad A = A(\beta) = (a_{ij}(\beta)),$$

$$a_{ij} = a_{ij}(\beta) = \begin{cases} s_{ij} e^{-\beta(E_j - E_i)_+} & (j \neq i) \\ 1 - \sum_{k \neq i} e^{-\beta(E_k - E_i)_+} & (j = i) \end{cases}$$

$$(E_j - E_i)_+ = \max(0, E_j - E_i), \qquad \beta = 1/T$$

即ち、或る初期解 $i_0 \in X$ から出発して一定温度 $T = 1/\beta$ で逐次SA法を行なったとき、t ステップ後の解の確率分布を行ベクトル $p_t \in [0,1]^X$ で表すと、

$$p_0 = (\delta_{ii_0})_{i \in X}, \qquad p_{t+1} = p_t A$$

が成り立つ。$t \to +\infty$ の極限では、p_t は次の Boltzmann 分布 π に収束する。

(A.2) $$\pi = \pi(\beta) = (\frac{1}{Z(\beta)} e^{-\beta E_i})_{i \in X},$$
$$Z(\beta) = \sum_{i \in X} e^{-\beta E_i}$$

X 上の確率分布 $p = (p_i)$ が平衡状態に対する分布 π からどれだけ離れているかは、2.3 節で述べたように、次の Kullback 情報量によって表される。

(A.3)
$$D(\pi \| p) = -\frac{1}{Z} \sum_i e^{-\beta E_i} \log p_i - \frac{\beta}{Z} \sum_i e^{-\beta E_i} E_i - \log Z$$

これに対して次の補題が成立することから、$D(\pi \| p_t)$ は、t について単調に減少することが判る。即ち、p_t は π に単調に近づいていく。

補題 A.1 X 上の任意の確率分布 p に対して、次式が成立する。

(A.4) $$D(\pi \| p) \geq D(\pi \| pA)$$

証明 a_{ij} の定義より、

(A.5) $$e^{-\beta E_i} a_{ij} = e^{-\beta E_j} a_{ji}$$

が成り立つことに注意すれば、

$$\begin{aligned} & \sum_i e^{-\beta E_i} \log(\sum_j p_j a_{ji}) \\ \geq & \sum_i e^{-\beta E_i} \sum_j a_{ij} \{\log p_j - \beta(E_j - E_i)\} \\ = & \sum_{i,j} e^{-\beta E_j} a_{ji} \log p_j - \beta \sum_{i,j} (e^{-\beta E_j} E_j a_{ji} - e^{-\beta E_i} E_i a_{ij}) \\ = & \sum_j e^{-\beta E_j} \log p_j \end{aligned}$$

よって、式 (A.3) により、上の不等式 (A.4) が成り立つ。
■

次に、SA 法を並列化したアルゴリズムについて考える。プロセッサの台数を N として、各プロセッサに一定温度 $T_1, T_2, \ldots, T_N$ を与える。ここで、$T_1 > T_2 > \ldots > T_N$ とし、$\beta_n = 1/T_n$ とする。各プロセッサは各々 1 つの解を保持し、全体では N 個の解が同時並列的に探索される。従って、この並列アルゴリズムの挙動は、X^N 上のマルコフ連鎖として表現できる。各プロセッサ上で、それぞれ、アニーリングを 1 ステップずつ進めることは、次の推移確率行列 $\tilde{A}$ で表現される。$\otimes$ はテンソル積を表す。

$$\tilde{A} = A(\beta_1) \otimes A(\beta_2) \otimes \cdots \otimes A(\beta_N)$$

また、このアルゴリズムで無限の時間をかけた時の、平衡状態の確率分布は、次の $\tilde{\pi}$ で与えられる。

$$\tilde{\pi} = \tilde{\pi}(\beta_1) \otimes \tilde{\pi}(\beta_2) \otimes \cdots \otimes \tilde{\pi}(\beta_N)$$

このとき、(2.3.2) の第 1 式に対応する次の補題が成立する。

補題 A.2 X^N 上の任意の確率分布 $\tilde{p}$ に対して、次式が成立する。

(A.6) $$D(\tilde{\pi} \| \tilde{p}) \geq D(\tilde{\pi} \| \tilde{p}\tilde{A})$$

証明 I_X を X 上の恒等変換を表す単位行列として、

$$\tilde{A}_n = I_X^{\otimes(n-1)} \otimes A(\beta_n) \otimes I_X^{\otimes(N-n-1)}$$

とおくと、補題 A.1 の証明と同様にして、

$$D(\tilde{\pi} \| \tilde{p}) \geq D(\tilde{\pi} \| \tilde{p}\tilde{A}_n)$$

が成立することが判る。一方、

$$\tilde{A} = \tilde{A}_1 \cdot \tilde{A}_2 \cdot \cdots \cdot \tilde{A}_N$$

であるから、帰納法により (A.6) 式が成り立つことが判る。
■

次に、プロセッサ間での解の確率的交換について考える。先ず、$N = 2$ で、$T_1 = T$, $T_2 = T'$ の場合について考える。2.2 節で述べたことから、これらのプロセッサ間での解の確率的交換は、次の推移確率行列で表される。

(A.7) $$C = C(\beta, \beta') = (c_{(ij)(kl)}(\beta, \beta'))$$

$$\begin{aligned} c_{(ij)(kl)} &= c_{(ij)(kl)}(\beta, \beta') \\ &= \begin{cases} c_{ij} & (i,j) = (l,k) \\ 1 - c_{ij} & (i,j) = (k,l) \neq (j,i) \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

$$c_{ij} = c_{ij}(\beta, \beta') = \exp(\min\{0, (\beta - \beta')(E_i - E_j)\})$$
$$\beta = 1/T, \quad \beta' = 1/T'$$

これに対して、次の補題が成立する。

補題 A.3 $N = 2$ のとき、$X^N = X^2$ 上の任意の確率分布 $\tilde{p}$ に対して、次式が成立する。

(A.8) $$D(\tilde{\pi} \| \tilde{p}) \geq D(\tilde{\pi} \| \tilde{p}C)$$

証明 $\tilde{p} = (p_{ij})$ とすると、(2.3.1) 式より、

(A.9)
$$\begin{aligned} & D(\tilde{\pi} \| \tilde{p}) \\ = & -\frac{1}{Z(\beta)Z(\beta')} \sum_{i,j} e^{-(\beta E_i + \beta' E_j)} \log p_{ij} \\ & -\frac{\beta}{Z(\beta)} \sum_i e^{-\beta E_i} E_i - \frac{\beta'}{Z(\beta')} \sum_j e^{-\beta' E_j} E_j \\ & - \log Z(\beta) - \log Z(\beta') \end{aligned}$$

よって、(A.8) 式を示すためには、(A.7),(A.9) より、

$$\sum_{i,j} e^{-(\beta E_i + \beta' E_j)} \log\{(1 - c_{ij}) p_{ij} + c_{ji} p_{ji}\}$$
$$\geq \sum_{i,j} e^{-(\beta E_i + \beta' E_j)} \log p_{ij}$$

を示せば良い。c_{ij} の定義より、

(A.10) $$e^{-(\beta E_i+\beta' E_j)}c_{ij} = e^{-(\beta E_j+\beta' E_i)}c_{ji}$$

が成り立つことに注意すれば、

$$\begin{aligned}
&\sum_{i,j} e^{-(\beta E_i+\beta' E_j)} \log\{(1-c_{ij})p_{ij} + c_{ji}p_{ji}\} \\
\geq\ &\sum_{i,j} e^{-(\beta E_i+\beta' E_j)}\{-\beta E_i - \beta\, E_j \\
&+(1-c_{ij})\log(e^{\beta E_i+\beta' E_j}p_{ij}) + c_{ij}\log(e^{\beta E_j+\beta' E_i}p_{ji})\} \\
=\ &\sum_{i,j} e^{-(\beta E_i+\beta' E_j)}(1-c_{ij})\log p_{ij} \\
&+\sum_{i,j} e^{-(\beta E_j+\beta' E_i)}c_{ji}\log p_{ji} \\
&-\sum_{i,j} e^{-(\beta E_i+\beta' E_j)}c_{ij}(\beta-\beta')(E_i-E_j) \\
=\ &\sum_{i,j} e^{-(\beta E_i+\beta' E_j)}c_{ij}\log p_{ij} \quad \blacksquare
\end{aligned}$$

次に、一般の $N>2$ の場合について考える。このとき、互いに隣合う温度のプロセッサ間で解の交換を一斉に行なう方法は2通りあり、それぞれ、次の遷移確率行列で表現される。

$$\tilde{C}_{\rm even} = \begin{cases} C_1 \otimes C_3 \otimes \cdots \otimes C_{N-1} & (N:\text{even}) \\ C_1 \otimes C_3 \otimes \cdots \otimes C_{N-2} \otimes I_X & (N:\text{odd}) \end{cases}$$

$$\tilde{C}_{\rm odd} = \begin{cases} I_X \otimes C_2 \otimes C_4 \otimes \cdots \otimes C_{N-2} \otimes I_X & (N:\text{even}) \\ I_X \otimes C_2 \otimes C_4 \otimes \cdots \otimes C_{N-1} & (N:\text{odd}) \end{cases}$$

$$C_n = C(\beta_n, \beta_{n+1})$$

このとき、(2.3.2) の第2式に対応する次の補題が成立する。

補題 A.4　X^N 上の任意の確率分布 $\tilde{p}$ に対して、次式が成立する。

(A.11) $$D(\tilde{\pi}\|\tilde{p}) \geq D(\tilde{\pi}\|\tilde{p}\tilde{C}_{\rm even}),$$
$$D(\tilde{\pi}\|\tilde{p}) \geq D(\tilde{\pi}\|\tilde{p}\tilde{C}_{\rm odd})$$

証明

$$\tilde{C}_n = I_X^{\otimes(n-1)} \otimes C_n \otimes I_X^{\otimes(N-n-2)}$$

とおくと、補題 A.3 の証明と同様にして、

$$D(\tilde{\pi}\|\tilde{p}) \geq D(\tilde{\pi}\|\tilde{p}\tilde{C}_n)$$

が成立することが判る。一方、$\tilde{C}_{\rm even}, \tilde{C}_{\rm odd}$ はそれぞれ $\tilde{C}_n$ の積に分解できるから、帰納法により上式が成り立つことが判る。$\blacksquare$

参考文献

[Kernighan 69] B.W. Kernighan and S. Lin, "An Efficient Heristic Procedure for Partitioning Graphs," *Bell. sys. tech. J.*, vol.49, pp.291-307 (1969).

[Kirkpatrick 83] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecci, "Optimization by Simulated Annealing," *Science*, vol.220, no.4598, pp.671-681 (1983).

[Hajek 86] B. Hajek, "Cooling Schedule for Optimal Simulated Annealing," submitted to *Mathematics of Operations Research* (1985), *revised version* (1986).

[White 84] S. R. White, "Concepts of Scales in Simulated Annealing," *Proc. IEEE ICCD* pp.646-651 (1984).

[Sechen 86] C. Sechen, and A. Sangiovanni-Vincentelli, "TimberWolf3.2: A New Standard Cell Placement and Global Routing Package", *Proc. 23rd Design Automation Conf.* pp.432-439 (1986).

[Aarts 85] E.H.L. Aarts, and P.J.M. van Laarhoven, "Statistical Cooling: A General Approach to Combinatorial Optimization Problems", *Philips J. Res.* vol.40, no.4, pp.193-226 (1985).

[Laarhoven 87] P.J.M. van Laarhoven, and E.H.L. Aarts, "Simulated Annealing: Theory and Applications", (Reidel, Dordrecht, Holland, 1987).

[Sechen 85] C. Sechen, and A. Sangiovanni-Vincentelli, "The TimberWolf Placement and Routing Package", *IEEE Journal of Solid-State Circuits* vol. SC-20, no.2, pp.510-522 (1985).

[Darema 87] F. Darema, S. Kirkpatrick and V.A. Norton, "Parallel Algorithms for Chip Placement by Simulated Annealing ," *IBM J. Res. Dev.* vol.31, no.3, pp.391-402 (1987).

[Casotto 87] A. Casotto, and A. Sangiovanni-Vincentelli, "Placement of Standard Cells Using Simulated Annealing on the Connection Machine," *Proc. IEEE ICCAD* pp.350-353 (1987).

[Amari 85] S. Amari, "Differential Geometric Methods in Statitics," Lecture Note in Statistics 28 (Springer, Verlag, 1985).

[Feller 57] W. Feller, "An Introduction to Probability Theory and Its Applications," vol. 1 (John Wiley & Sons, 1957).

[Chikayama 88] T. Chikayama, H. Sato, and T. Miyazaki, "Overview of the Parallel Inference Machine Operating System (PIMOS)", *Proc. Int. Conf. FGCS* pp.230–251 (1988).

[Nakajima 89] K. Nakajima, Y. Inamura, N. Ichiyoshi, K. Rokusawa, and T. Chikayama, "Distributed Implementation of KL1 on the Multi-PSI/V2", *Proc. 6th Int. Conf. on Logic Programming* pp.436–451 (1989).

[Garey 79] M. Garey and D. Johnson, "Computers and Intractability, A Guide to the Theory of NP-Completeness," (Freeman, New York, 1979).

表 1: プロセッサ間の解の確率的交換頻度 vs リダクション数、処理時間

解の交換頻度	1/20000	1/2000	1/1000	1/400	1/200	1/100
リダクション数	64445599	64457551	64476702	64521305	64608180	64776242
処理時間 (秒)	38.178	37.569	38.500	38.394	39.095	41.319
解の交換頻度	1/50	1/32	1/16	1/8	1/4	1/2
リダクション数	65111892	65479469	66520114	68596721	72748891	81060005
処理時間 (秒)	44.439	48.611	59.843	82.022	126.460	214.664

表 2: プロセッサ間の解の確率的交換頻度 vs プロセッサ稼働率

解の交換頻度	1/20000	1/2000	1/1000	1/400	1/200	1/100
稼働率実測値 (%)	90 ～ 100	90 ～ 100	90 ～ 100	80 ～ 100	80 ～ 100	70 ～ 100
稼働率推定値 (%)	100.0	99.6	99.1	97.8	95.8	91.9
解の交換頻度	1/50	1/32	1/16	1/8	1/4	1/2
稼働率実測値 (%)	70 ～ 90	60 ～ 80	50 ～ 70	30 ～ 50	20 ～ 40	10 ～ 30
稼働率推定値 (%)	85.0	78.5	65.0	48.9	33.7	22.1

表 3: 得られた解の比較評価結果

手法	E	C	B
並列 SA 法 ($f = 1/400$)	−1736	568	0
	−1748	564	−2
	−1740	567	0
逐次 SA 法	−1640.3	591.9	−0.032
Kernighan-Lin 法	−1700.8	576.8	0.000

$E = -\sum\lambda(u)\lambda(v) + (\sum\lambda(v))^2 = 2C + B^2 - L$: 目的関数 (エネルギー)

$C = \#\{(u,v)|\lambda(u) \neq \lambda(v),\ u < v\}$: 2 分割された頂点集合間にまたがる辺の数

$B = \sum\lambda(v)$: 2 分割された頂点集合のサイズの差　　$L = 2004$: グラフの辺の総数

Fig.1. energy vs frequency of exchanges

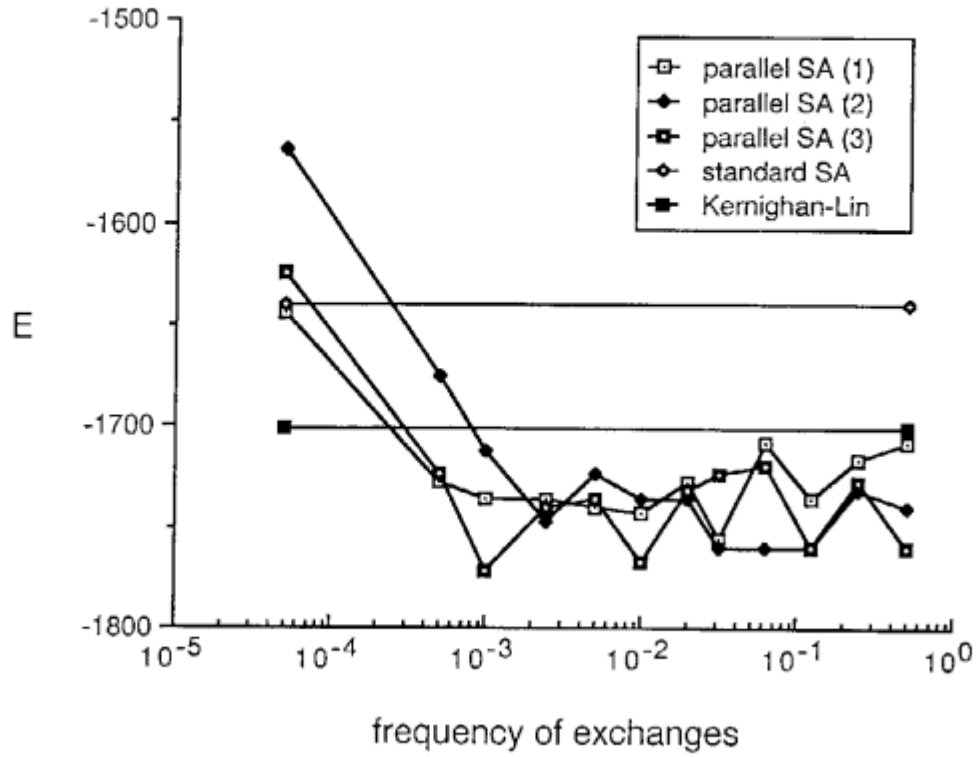

Fig.2. reductions, time vs frequency of exchanges